man pages section 1M: System Administration Commands

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 816–0211–10
May 2002

Copyright 2002 Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 U.S.A. All rights reserved.

This product or document is protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this product or document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any. Third-party software, including font technology, is copyrighted and licensed from Sun suppliers.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the U.S. and other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, docs.sun.com, AnswerBook, AnswerBook2, and Solaris are trademarks, registered trademarks, or service marks of Sun Microsystems, Inc. in the U.S. and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the U.S. and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The OPEN LOOK and Sun™ Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

Federal Acquisitions: Commercial Software–Government Users Subject to Standard License Terms and Conditions.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Copyright 2002 Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 U.S.A. Tous droits réservés

Ce produit ou document est protégé par un copyright et distribué avec des licences qui en restreignent l'utilisation, la copie, la distribution, et la décompilation. Aucune partie de ce produit ou document ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a. Le logiciel détenu par des tiers, et qui comprend la technologie relative aux polices de caractères, est protégé par un copyright et licencié par des fournisseurs de Sun.

Des parties de ce produit pourront être dérivées du système Berkeley BSD licenciés par l'Université de Californie. UNIX est une marque déposée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company, Ltd.

Sun, Sun Microsystems, le logo Sun, docs.sun.com, AnswerBook, AnswerBook2, et Solaris sont des marques de fabrique ou des marques déposées, ou marques de service, de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays. Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc.

L'interface d'utilisation graphique OPEN LOOK et Sun™ a été développée par Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox pour la recherche et le développement du concept des interfaces d'utilisation visuelle ou graphique pour l'industrie de l'informatique. Sun détient une licence non exclusive de Xerox sur l'interface d'utilisation graphique Xerox, cette licence couvrant également les licenciés de Sun qui mettent en place l'interface d'utilisation graphique OPEN LOOK et qui en outre se conforment aux licences écrites de Sun.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" ET AUCUNE GARANTIE, EXPRESSE OU IMPLICITE, N'EST ACCORDEE, Y COMPRIS DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DE LA PUBLICATION A REPONDRE A UNE UTILISATION PARTICULIERE, OU LE FAIT QU'ELLE NE SOIT PAS CONTREFAISANTE DE PRODUIT DE TIERS. CE DENI DE GARANTIE NE S'APPLIQUERAIT PAS, DANS LA MESURE OU IL SERAIT TENU JURIDIQUEMENT NUL ET NON AVENU.

020313@3332

Contents

Preface 19

Introduction 25
Intro(1M) 26

System Administration Commands 27
accept(1M) 28
acct(1M) 30
acctadm(1M) 33
acctcms(1M) 36
acctcon(1M) 38
acctmerg(1M) 40
acctprc(1M) 41
acctsh(1M) 43
adbgen(1M) 46
addbadsec(1M) 49
add_drv(1M) 51
admintool(1M) 55
afbconfig(1M) 57
aliasadm(1M) 65
answerbook2_admin(1M) 67
apache(1M) 68
arp(1M) 70
aset(1M) 72
aset.restore(1M) 78

audit(1M) 79
auditconfig(1M) 80
auditd(1M) 86
auditreduce(1M) 88
audit_startup(1M) 96
auditstat(1M) 97
audit_warn(1M) 99
automount(1M) 101
automountd(1M) 108
autopush(1M) 109
bdconfig(1M) 111
boot(1M) 113
bsmconv(1M) 127
bsmrecord(1M) 129
busstat(1M) 132
cachefsd(1M) 136
cachefslog(1M) 137
cachefspack(1M) 139
cachefsstat(1M) 141
cachefswssize(1M) 143
captoinfo(1M) 145
catman(1M) 146
cfgadm(1M) 150
cfgadm_ac(1M) 161
cfgadm_pci(1M) 165
cfgadm_sbd(1M) 170
cfgadm_scsi(1M) 184
cfgadm_sysctrl(1M) 190
cfgadm_usb(1M) 194
cfsadmin(1M) 204
cg14config(1M) 208
chat(1M) 210
check-hostname(1M) 218
check-permissions(1M) 219
chown(1M) 220
chroot(1M) 221
cimworkshop(1M) 222
clear_locks(1M) 224

clinfo(1M) 225
clri(1M) 226
consadm(1m) 227
conv_lp(1M) 229
conv_lpd(1M) 230
coreadm(1M) 232
cpustat(1M) 236
cron(1M) 239
cvcd(1M) 241
dcs(1M) 242
dd(1M) 243
devattr(1M) 249
devfree(1M) 250
devfsadm(1M) 251
devinfo(1M) 253
devlinks(1M) 254
devnm(1M) 258
devreserv(1M) 259
df(1M) 261
dfmounts(1M) 265
dfmounts_nfs(1M) 267
dfshares(1M) 268
dfshares_nfs(1M) 269
df_ufs(1M) 271
dhcpagent(1M) 272
dhcpconfig(1M) 276
dhcpmgr(1M) 283
dhtadm(1M) 285
dig(1M) 291
directoryserver(1M) 298
disks(1M) 317
diskscan(1M) 321
dispadmin(1M) 322
dmesg(1M) 325
dmi_cmd(1M) 326
dmiget(1M) 329
dminfo(1M) 330
dmispd(1M) 332

dnskeygen(1M) 333
domainname(1M) 335
drvconfig(1M) 336
dsvclockd(1M) 338
dumpadm(1M) 339
editmap(1M) 344
edquota(1M) 346
eeprom(1M) 348
efdaemon(1M) 355
etrn(1M) 356
fbconfig(1M) 358
fdetach(1M) 360
fdisk(1M) 361
ff(1M) 366
ffbconfig(1M) 368
ff_ufs(1M) 376
flar(1M) 377
flarcreate(1M) 380
fmthard(1M) 383
fncheck(1M) 386
fncopy(1M) 387
fncreate(1M) 389
fncreate_fs(1M) 396
fncreate_printer(1M) 401
fndestroy(1M) 404
fnselect(1M) 405
fnsypd(1M) 407
format(1M) 408
fsck(1M) 412
fsck_cachefs(1M) 416
fsck_pcfs(1M) 417
fsck_udfs(1M) 419
fsck_ufs(1M) 422
fsdb(1M) 426
fsdb_udfs(1M) 427
fsdb_ufs(1M) 435
fsirand(1M) 445
fssnap(1M) 446

fssnap_ufs(1M) 448
fstyp(1M) 453
ftpaddhost(1M) 454
ftpconfig(1M) 456
ftprestart(1M) 457
ftpshut(1M) 458
fuser(1M) 460
fwtmp(1M) 462
getdev(1M) 463
getdgrp(1M) 465
getent(1M) 467
gettable(1M) 469
getty(1M) 470
getvol(1M) 472
gkadmin(1M) 474
groupadd(1M) 476
groupdel(1M) 478
groupmod(1M) 479
growfs(1M) 481
gsscred(1M) 484
gssd(1M) 486
halt(1M) 487
hostconfig(1M) 488
htable(1M) 490
id(1M) 491
idsconfig(1M) 494
ifconfig(1M) 496
if_mpadm(1M) 511
ifparse(1M) 513
ikeadm(1M) 515
ikecert(1M) 522
in.comsat(1M) 527
in.dhcpd(1M) 528
inetd(1M) 533
in.fingerd(1M) 536
infocmp(1M) 537
in.ftpd(1M) 541
in.iked(1M) 548

init(1M) 550
init.wbem(1M) 555
in.lpd(1M) 557
in.mpathd(1M) 558
in.named(1M) 562
in.ndpd(1M) 568
in.rarpd(1M) 570
in.rdisc(1M) 572
in.rexecd(1M) 574
in.ripngd(1M) 576
in.rlogind(1M) 579
in.routed(1M) 582
in.rshd(1M) 586
in.rwhod(1M) 589
install(1M) 591
installboot(1M) 593
installer(1M) 595
installf(1M) 596
install_scripts(1M) 600
in.talkd(1M) 607
in.telnetd(1M) 608
in.tftpd(1M) 611
in.tnamed(1M) 613
in.uucpd(1M) 614
iostat(1M) 616
ipsecconf(1M) 620
ipseckey(1M) 636
kadb(1M) 645
kadmin(1M) 651
kadmind(1M) 663
kdb5_util(1M) 665
kdmconfig(1M) 667
kernel(1M) 670
keyserv(1M) 673
killall(1M) 675
kprop(1M) 676
krb5kdc(1M) 678
kstat(1M) 680

ktkt_warnd(1M) 684
labelit(1M) 685
labelit_hsfs(1M) 687
labelit_udfs(1M) 688
labelit_ufs(1M) 690
ldapaddent(1M) 691
ldap_cachemgr(1M) 695
ldapclient(1M) 697
link(1M) 706
listdgrp(1M) 707
listen(1M) 708
llc2_loop(1M) 710
locator(1M) 712
lockd(1M) 714
lockfs(1M) 715
lockstat(1M) 718
lofiadm(1M) 726
logadm(1M) 731
logins(1M) 740
lpadmin(1M) 742
lpfilter(1M) 755
lpforms(1M) 761
lpget(1M) 768
lpmove(1M) 770
lpsched(1M) 772
lpset(1M) 773
lpshut(1M) 776
lpsystem(1M) 777
lpusers(1M) 778
lu(1M) 780
luactivate(1M) 783
lucancel(1M) 785
lucompare(1M) 786
lucreate(1M) 789
lucurr(1M) 797
ludelete(1M) 799
ludesc(1M) 801
lufslist(1M) 804

lumake(1M) 806
lumount(1M) 808
lurename(1M) 811
lustatus(1M) 813
luupgrade(1M) 815
luxadm(1M) 823
m64config(1M) 838
mail.local(1M) 843
makedbm(1M) 845
makemap(1M) 847
makeuuid(1M) 849
mdmonitord(1M) 851
medstat(1M) 852
metaclear(1M) 854
metadb(1M) 856
metadevadm(1M) 861
metahs(1M) 863
metainit(1M) 866
metaoffline(1M) 876
metaparam(1M) 878
metarecover(1M) 880
metarename(1M) 882
metareplace(1M) 886
metaroot(1M) 889
metaset(1M) 891
metastat(1M) 896
metasync(1M) 899
metattach(1M) 901
mib2mof(1M) 905
mibiisa(1M) 907
mipagent(1M) 930
mipagentconfig(1M) 933
mipagentstat(1M) 939
mkdevalloc(1M) 941
mkdevmaps(1M) 942
mkfifo(1M) 943
mkfile(1M) 944
mkfs(1M) 945

mkfs_pcfs(1M) 947
mkfs_udfs(1M) 951
mkfs_ufs(1M) 953
mknod(1M) 956
modinfo(1M) 957
modload(1M) 958
modunload(1M) 959
mofcomp(1M) 960
mofreg(1M) 963
monitor(1M) 966
mount(1M) 978
mountall(1M) 981
mount_cachefs(1M) 983
mountd(1M) 986
mount_hsfs(1M) 987
mount_nfs(1M) 989
mount_pcfs(1M) 997
mount_tmpfs(1M) 999
mount_udfs(1M) 1001
mount_ufs(1M) 1003
mount_xmemfs(1M) 1007
mpstat(1M) 1009
msgid(1M) 1011
mvdir(1M) 1012
named-bootconf(1M) 1013
named-xfer(1M) 1014
ncaconfd(1M) 1016
ncheck(1M) 1017
ncheck_ufs(1M) 1019
ndc(1M) 1020
ndd(1M) 1022
netstat(1M) 1024
newaliases(1M) 1031
newfs(1M) 1033
newkey(1M) 1037
nfsd(1M) 1039
nfslogd(1M) 1041
nfsstat(1M) 1044

nisaddcred(1M) 1049
nisaddent(1M) 1055
nisauthconf(1M) 1060
nisbackup(1M) 1062
nis_cachemgr(1M) 1065
nisclient(1M) 1067
nisinit(1M) 1072
nisldapmaptest(1M) 1076
nislog(1M) 1080
nisping(1M) 1081
nispopulate(1M) 1084
nisprefadm(1M) 1089
nisrestore(1M) 1093
nisserver(1M) 1096
nissetup(1M) 1099
nisshowcache(1M) 1100
nisstat(1M) 1101
nisupdkeys(1M) 1103
nlsadmin(1M) 1105
nscd(1M) 1111
nslookup(1M) 1113
nstest(1M) 1121
nsupdate(1M) 1125
ntpdate(1M) 1128
ntpq(1M) 1131
ntptrace(1M) 1138
obpsym(1M) 1140
ocfserv(1M) 1142
parse_dynamic_clustertoc(1M) 1143
passmgmt(1M) 1144
patchadd(1M) 1146
patchrm(1M) 1156
pbind(1M) 1162
pcmciad(1M) 1164
pfinstall(1M) 1165
pgxconfig(1M) 1169
picld(1M) 1174
ping(1M) 1176

pkgadd(1M) 1181
pkgask(1M) 1184
pkgchk(1M) 1186
pkgrm(1M) 1189
pmadm(1M) 1192
pmconfig(1M) 1197
pntadm(1M) 1199
pooladm(1M) 1206
poolbind(1M) 1208
poolcfg(1M) 1210
ports(1M) 1214
powerd(1M) 1218
pppd(1M) 1219
pppoec(1M) 1244
pppoed(1M) 1247
pppstats(1M) 1252
praudit(1M) 1255
printmgr(1M) 1256
privatepw(1M) 1258
prodreg(1M) 1260
projadd(1M) 1261
projdel(1M) 1263
projmod(1M) 1264
prstat(1M) 1266
prtconf(1M) 1271
prtdiag(1M) 1273
prtfru(1M) 1274
prtpicl(1M) 1275
prtvtoc(1M) 1276
psradm(1M) 1278
psrinfo(1M) 1281
psrset(1M) 1283
putdev(1M) 1287
putdgrp(1M) 1290
pwck(1M) 1292
pwconv(1M) 1293
quot(1M) 1295
quota(1M) 1297

quotacheck(1M) 1298
quotaon(1M) 1299
rctladm(1M) 1301
rdate(1M) 1303
reboot(1M) 1304
rem_drv(1M) 1306
removef(1M) 1307
repquota(1M) 1309
re-preinstall(1M) 1310
rmmount(1M) 1313
rmt(1M) 1316
roleadd(1M) 1318
roledel(1M) 1322
rolemod(1M) 1324
route(1M) 1327
rpcbind(1M) 1332
rpc.bootparamd(1M) 1334
rpcinfo(1M) 1335
rpc.metad(1M) 1339
rpc.metamedd(1M) 1340
rpc.metamhd(1M) 1341
rpc.nisd(1M) 1342
rpc.nisd_resolv(1M) 1347
rpc.nispasswdd(1M) 1348
rpc.rexd(1M) 1350
rpc.rstatd(1M) 1352
rpc.rusersd(1M) 1353
rpc.rwalld(1M) 1354
rpc.smserverd(1M) 1355
rpc.sprayd(1M) 1356
rpc.yppasswdd(1M) 1357
rpc.ypupdated(1M) 1359
rpld(1M) 1360
rquotad(1M) 1365
rsh(1M) 1366
rtc(1M) 1368
runacct(1M) 1369
rwall(1M) 1372

sac(1M) 1373
sacadm(1M) 1376
sadmind(1M) 1380
saf(1M) 1384
sar(1M) 1401
savecore(1M) 1403
sckmd(1M) 1405
sendmail(1M) 1407
setuname(1M) 1427
sf880drd(1M) 1428
sftp-server(1M) 1429
share(1M) 1430
shareall(1M) 1432
share_nfs(1M) 1433
showmount(1M) 1440
showrev(1M) 1441
shutdown(1M) 1443
slpd(1M) 1445
smartcard(1M) 1447
smattrpop(1M) 1455
smc(1M) 1460
smccompile(1M) 1464
smcconf(1M) 1468
smcregister(1M) 1475
smcron(1M) 1485
smdiskless(1M) 1492
smexec(1M) 1498
smgroup(1M) 1502
smlog(1M) 1506
smmaillist(1M) 1509
smmultiuser(1M) 1513
smosservice(1M) 1518
smpatch(1M) 1523
smprofile(1M) 1529
smrole(1M) 1535
smrsh(1M) 1542
smserialport(1M) 1543
smuser(1M) 1549

snmpdx(1M) 1557
snmpXdmid(1M) 1560
snmpXwbemd(1M) 1562
snoop(1M) 1564
soconfig(1M) 1575
soladdapp(1M) 1577
soldelapp(1M) 1578
solstice(1M) 1579
sppptun(1M) 1580
spray(1M) 1582
ssaadm(1M) 1583
sshd(1M) 1589
statd(1M) 1597
strace(1M) 1598
strclean(1M) 1600
strerr(1M) 1601
sttydefs(1M) 1603
su(1M) 1605
sulogin(1M) 1608
suninstall(1M) 1609
SUNWgfb_config(1M) 1610
SUNWifb_config(1M) 1618
swap(1M) 1626
swmtool(1M) 1629
sxconfig(1M) 1630
sync(1M) 1633
syncinit(1M) 1634
syncloop(1M) 1637
syncstat(1M) 1640
sysdef(1M) 1642
syseventadm(1M) 1644
syseventconfd(1M) 1648
syseventd(1M) 1649
sysidconfig(1M) 1651
sysidtool(1M) 1654
syslogd(1M) 1657
sys-unconfig(1M) 1660
tapes(1M) 1662

taskstat(1M) 1666
tcxconfig(1M) 1667
th_define(1M) 1668
th_manage(1M) 1677
tic(1M) 1679
traceroute(1M) 1680
trapstat(1M) 1687
ttyadm(1M) 1698
ttymon(1M) 1700
tunefs(1M) 1703
tzselect(1M) 1705
uadmin(1M) 1706
ufsdump(1M) 1707
ufsrestore(1M) 1714
unshare(1M) 1721
unshare_nfs(1M) 1722
update_drv(1M) 1723
useradd(1M) 1724
userdel(1M) 1729
usermod(1M) 1731
utmpd(1M) 1735
uucheck(1M) 1736
uucico(1M) 1737
uucleanup(1M) 1739
uusched(1M) 1741
Uutry(1M) 1742
uuxqt(1M) 1743
vmstat(1M) 1744
volcopy(1M) 1748
volcopy_ufs(1M) 1750
vold(1M) 1751
wall(1M) 1753
wbemadmin(1M) 1755
wbemconfig(1M) 1758
wbemlogviewer(1M) 1759
whodo(1M) 1761
wracct(1M) 1763
wrsmconf(1M) 1765

wrsmstat(1M) 1767
xntpd(1M) 1770
xntpdc(1M) 1786
ypbind(1M) 1794
ypinit(1M) 1796
ypmake(1M) 1797
yppoll(1M) 1798
yppush(1M) 1799
ypserv(1M) 1801
ypset(1M) 1804
ypstart(1M) 1806
ypxfr(1M) 1807
zdump(1M) 1809
zic(1M) 1810

Index 1815

Preface

Both novice users and those familar with the SunOS operating system can use online man pages to obtain information about the system and its features. A man page is intended to answer concisely the question “What does it do?” The man pages in general comprise a reference manual. They are not intended to be a tutorial.

Overview

The following contains a brief description of each man page section and the information it references:

- Section 1 describes, in alphabetical order, commands available with the operating system.
- Section 1M describes, in alphabetical order, commands that are used chiefly for system maintenance and administration purposes.
- Section 2 describes all of the system calls. Most of these calls have one or more error returns. An error condition is indicated by an otherwise impossible returned value.
- Section 3 describes functions found in various libraries, other than those functions that directly invoke UNIX system primitives, which are described in Section 2.
- Section 4 outlines the formats of various files. The C structure declarations for the file formats are given where applicable.
- Section 5 contains miscellaneous documentation such as character-set tables.
- Section 6 contains available games and demos.
- Section 7 describes various special files that refer to specific hardware peripherals and device drivers. STREAMS software drivers, modules and the STREAMS-generic set of system calls are also described.

- Section 9 provides reference information needed to write device drivers in the kernel environment. It describes two device driver interface specifications: the Device Driver Interface (DDI) and the Driver/Kernel Interface (DKI).
- Section 9E describes the DDI/DKI, DDI-only, and DKI-only entry-point routines a developer can include in a device driver.
- Section 9F describes the kernel functions available for use by device drivers.
- Section 9S describes the data structures used by drivers to share information between the driver and the kernel.

Below is a generic format for man pages. The man pages of each manual section generally follow this order, but include only needed headings. For example, if there are no bugs to report, there is no BUGS section. See the `intro` pages for more information and detail about each section, and `man`(1) for more information about man pages in general.

NAME	This section gives the names of the commands or functions documented, followed by a brief description of what they do.
SYNOPSIS	This section shows the syntax of commands or functions. When a command or file does not exist in the standard path, its full path name is shown. Options and arguments are alphabetized, with single letter arguments first, and options with arguments next, unless a different argument order is required. The following special characters are used in this section: [ ] Brackets. The option or argument enclosed in these brackets is optional. If the brackets are omitted, the argument must be specified. . . . Ellipses. Several values can be provided for the previous argument, or the previous argument can be specified multiple times, for example, "`filename` . . ." . | Separator. Only one of the arguments separated by this character can be specified at a time. { } Braces. The options and/or arguments enclosed within braces are interdependent, such that everything enclosed must be treated as a unit.

PROTOCOL	This section occurs only in subsection 3R to indicate the protocol description file.
DESCRIPTION	This section defines the functionality and behavior of the service. Thus it describes concisely what the command does. It does not discuss OPTIONS or cite EXAMPLES. Interactive commands, subcommands, requests, macros, and functions are described under USAGE.
IOCTL	This section appears on pages in Section 7 only. Only the device class that supplies appropriate parameters to the `ioctl`(2) system call is called `ioctl` and generates its own heading. `ioctl` calls for a specific device are listed alphabetically (on the man page for that specific device). `ioctl` calls are used for a particular class of devices all of which have an `io` ending, such as `mtio`(7I).
OPTIONS	This secton lists the command options with a concise summary of what each option does. The options are listed literally and in the order they appear in the SYNOPSIS section. Possible arguments to options are discussed under the option, and where appropriate, default values are supplied.
OPERANDS	This section lists the command operands and describes how they affect the actions of the command.
OUTPUT	This section describes the output – standard output, standard error, or output files – generated by the command.
RETURN VALUES	If the man page documents functions that return values, this section lists these values and describes the conditions under which they are returned. If a function can return only constant values, such as 0 or –1, these values are listed in tagged paragraphs. Otherwise, a single paragraph describes the return values of each function. Functions declared void do not return values, so they are not discussed in RETURN VALUES.
ERRORS	On failure, most functions place an error code in the global variable `errno` indicating why they failed. This section lists alphabetically all error codes a function can generate and describes the conditions that cause each error. When more than

	one condition can cause the same error, each condition is described in a separate paragraph under the error code.
USAGE	This section lists special rules, features, and commands that require in-depth explanations. The subsections listed here are used to explain built-in functionality: Commands Modifiers Variables Expressions Input Grammar
EXAMPLES	This section provides examples of usage or of how to use a command or function. Wherever possible a complete example including command-line entry and machine response is shown. Whenever an example is given, the prompt is shown as `example%`, or if the user must be superuser, `example#`. Examples are followed by explanations, variable substitution rules, or returned values. Most examples illustrate concepts from the SYNOPSIS, DESCRIPTION, OPTIONS, and USAGE sections.
ENVIRONMENT VARIABLES	This section lists any environment variables that the command or function affects, followed by a brief description of the effect.
EXIT STATUS	This section lists the values the command returns to the calling program or shell and the conditions that cause these values to be returned. Usually, zero is returned for successful completion, and values other than zero for various error conditions.
FILES	This section lists all file names referred to by the man page, files of interest, and files created or required by commands. Each is followed by a descriptive summary or explanation.
ATTRIBUTES	This section lists characteristics of commands, utilities, and device drivers by defining the attribute type and its corresponding value. See `attributes`(5) for more information.
SEE ALSO	This section lists references to other man pages, in-house documentation, and outside publications.

DIAGNOSTICS	This section lists diagnostic messages with a brief explanation of the condition causing the error.
WARNINGS	This section lists warnings about special conditions which could seriously affect your working conditions. This is not a list of diagnostics.
NOTES	This section lists additional information that does not belong anywhere else on the page. It takes the form of an aside to the user, covering points of special interest. Critical information is never covered here.
BUGS	This section describes known bugs and, wherever possible, suggests workarounds.

Introduction

NAME

Intro – introduction to maintenance commands and application programs

DESCRIPTION

This section describes, in alphabetical order, commands that are used chiefly for system maintenance and administration purposes.

Because of command restructuring for the Virtual File System architecture, there are several instances of multiple manual pages that begin with the same name. For example, the `mount`, pages – `mount`(1M), `mount_cachefs`(1M), `mount_hsfs`(1M), `mount_nfs`(1M), `mount_tmpfs`(1M), and `mount_ufs`(1M). In each such case the first of the multiple pages describes the syntax and options of the generic command, that is, those options applicable to all FSTypes (file system types). The succeeding pages describe the functionality of the FSType-specific modules of the command. These pages list the command followed by an underscore (_) and the FSType to which they pertain. Note that the administrator should not attempt to call these modules directly. The generic command provides a common interface to all of them. Thus the FSType-specific manual pages should not be viewed as describing distinct commands, but rather as detailing those aspects of a command that are specific to a particular FSType.

COMMAND SYNTAX

Unless otherwise noted, commands described in this section accept options and other arguments according to the following syntax:

name [*option*(*s*)] [*cmdarg*(*s*)] where:

name	The name of an executable file.
option	– *noargletter*(*s*) or, – *argletter*< >*optarg* where < > is optional white space.
noargletter	A single letter representing an option without an argument.
argletter	A single letter representing an option requiring an argument.
optarg	Argument (character string) satisfying preceding *argletter*.
cmdarg	Pathname (or other command argument) *not* beginning with – or, – by itself indicating the standard input.

ATTRIBUTES

See `attributes`(5) for a discussion of the attributes listed in this section.

SEE ALSO

`getopt`(1), `getopt`(3C), `attributes`(5)

DIAGNOSTICS

Upon termination, each command returns 0 for normal termination and non-zero to indicate troubles such as erroneous parameters, bad or inaccessible data, or other inability to cope with the task at hand. It is called variously ''exit code,'' ''exit status,'' or ''return code,'' and is described only where special conventions are involved.

NOTES

Unfortunately, not all commands adhere to the standard syntax.

System Administration Commands

NAME

accept, reject – accept or reject print requests

SYNOPSIS

accept *destination...*

reject [-r *reason*] *destination...*

DESCRIPTION

`accept` allows the queueing of print requests for the named destinations.

`reject` prevents queueing of print requests for the named destinations.

Use `lpstat -a` to check if destinations are accepting or rejecting print requests.

`accept` and `reject` must be run on the print server; they have no meaning to a client system.

OPTIONS

The following options are supported for `reject`:

`-r` *reason* — Assigns a reason for rejection of print requests for *destination.* Enclose *reason* in quotes if it contains blanks. *reason* is reported by `lpstat -a`. By default, *reason* is `unknown reason` for existing destinations, and `new printer` for destinations added to the system but not yet accepting requests.

OPERANDS

The following operands are supported.

destination — The name of the destination accepting or rejecting print requests. Destination specifies the name of a printer or class of printers (see `lpadmin`(1M)). Specify *destination* using atomic name. See `printers.conf`(4) for information regarding the naming conventions for atomic names.

EXIT STATUS

The following exit values are returned:

`0` — Successful completion.

non-zero — An error occurred.

FILES

`/var/spool/lp/*` — LP print queue.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpcu
CSI	Enabled (see `NOTES`)

SEE ALSO

`enable`(1), `lp`(1), `lpstat`(1), `lpadmin`(1M), `lpsched`(1M), `printers.conf` (4), `attributes`(5)

NOTES `accept` and `reject` affect only queueing on the print server's spooling system. Requests made from a client system remain queued in the client system's queueing mechanism until they are cancelled or accepted by the print server's spooling system.

`accept` is CSI-enabled except for the *destination* name.

NAME

acct, acctdisk, acctdusg, accton, acctwtmp, closewtmp, utmp2wtmp – overview of accounting and miscellaneous accounting commands

SYNOPSIS

/usr/lib/acct/acctdisk

/usr/lib/acct/acctdusg [-u *filename*] [-p *filename*]

/usr/lib/acct/accton [*filename*]

/usr/lib/acct/acctwtmp *reason filename*

/usr/lib/acct/closewtmp

/usr/lib/acct/utmp2wtmp

DESCRIPTION

Accounting software is structured as a set of tools (consisting of both C programs and shell procedures) that can be used to build accounting systems. `acctsh`(1M) describes the set of shell procedures built on top of the C programs.

Connect time accounting is handled by various programs that write records into `/var/adm/wtmpx`, as described in `utmpx`(4). The programs described in `acctcon`(1M) convert this file into session and charging records, which are then summarized by `acctmerg`(1M).

Process accounting is performed by the system kernel. Upon termination of a process, one record per process is written to a file (normally `/var/adm/pacct`). The programs in `acctprc`(1M) summarize this data for charging purposes; `acctcms`(1M) is used to summarize command usage. Current process data may be examined using `acctcom`(1).

Process accounting records and connect time accounting records (or any accounting records in the `tacct` format described in `acct`(3HEAD)) can be merged and summarized into total accounting records by `acctmerg` (see `tacct` format in `acct`(3HEAD)). `prtacct` (see `acctsh`(1M)) is used to format any or all accounting records.

`acctdisk` reads lines that contain user ID, login name, and number of disk blocks and converts them to total accounting records that can be merged with other accounting records. `acctdisk` returns an error if the input file is corrupt or improperly formatted.

`acctdusg` reads its standard input (usually from `find / -print`) and computes disk resource consumption (including indirect blocks) by login.

`accton` without arguments turns process accounting off. If *filename* is given, it must be the name of an existing file, to which the kernel appends process accounting records (see `acct`(2) and `acct`(3HEAD)).

`acctwtmp` writes a `utmpx`(4) record to *filename*. The record contains the current time and a string of characters that describe the *reason*. A record type of `ACCOUNTING` is assigned (see `utmpx`(4)) *reason* must be a string of 11 or fewer characters, numbers, `$`, or spaces. For example, the following are suggestions for use in reboot and shutdown procedures, respectively:

```
acctwtmp "acctg on" /var/adm/wtmpx
acctwtmp "acctg off" /var/adm/wtmpx
```

For each user currently logged on, `closewtmp` puts a false `DEAD_PROCESS` record in the `/var/adm/wtmpx` file. `runacct` (see `runacct`(1M)) uses this false `DEAD_PROCESS` record so that the connect accounting procedures can track the time used by users logged on before `runacct` was invoked.

For each user currently logged on, `runacct` uses `utmp2wtmp` to create an entry in the file `/var/adm/wtmpx`, created by `runacct`. Entries in `/var/adm/wtmpx` enable subsequent invocations of `runacct` to account for connect times of users currently logged in.

OPTIONS

The following options are supported:

`-u` *filename* — Places in *filename* records consisting of those filenames for which `acctdusg` charges no one (a potential source for finding users trying to avoid disk charges).

`-p` *filename* — Specifies a password file, *filename*. This option is not needed if the password file is `/etc/passwd`.

ENVIRONMENT VARIABLES

If any of the `LC_*` variables (`LC_TYPE`, `LC_MESSAGES`, `LC_TIME`, `LC_COLLATE`, `LC_NUMERIC`, and `LC_MONETARY`) (see `environ`(5)) are not set in the environment, the operational behavior of `acct` for each corresponding locale category is determined by the value of the `LANG` environment variable. If `LC_ALL` is set, its contents are used to override both the `LANG` and the other `LC_*` variables. If none of the above variables are set in the environment, the "C" (U.S. style) locale determines how `acct` behaves.

`LC_CTYPE` — Determines how `acct` handles characters. When `LC_CTYPE` is set to a valid value, `acct` can display and handle text and filenames containing valid characters for that locale. `acct` can display and handle Extended Unix Code (EUC) characters where any character can be 1, 2, or 3 bytes wide. `acct` can also handle EUC characters of 1, 2, or more column widths. In the "C" locale, only characters from ISO 8859-1 are valid.

`LC_TIME` — Determines how `acct` handles date and time formats. In the "C" locale, date and time handling follows the U.S. rules.

FILES

`/etc/passwd` — Used for login name to user ID conversions.

`/usr/lib/acct` — Holds all accounting commands listed in sub-class 1M of this manual.

`/var/adm/pacct`	Current process accounting file.
`/var/adm/wtmpx`	History of user access and administration information..

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWaccu

SEE ALSO

`acctcom`(1), `acctcms`(1M), `acctcon`(1M), `acctmerg`(1M), `acctprc`(1M), `acctsh`(1M), `fwtmp`(1M), `runacct`(1M), `acct`(2), `acct`(3HEAD), `passwd`(4), `utmpx`(4), `attributes`(5), `environ`(5)

System Administration Guide: Basic Administration

NAME

acctadm – configure extended accounting facility

SYNOPSIS

/usr/sbin/acctadm [-DErux] [-d *resource_list*] [-e *resource_list*] [-f *filename*] [task | process]

DESCRIPTION

acctadm configures various attributes of the extended accounting facility. Without arguments, acctadm displays the current status of the extended accounting facility.

OPTIONS

The following options are supported:

-d *resource_list* — Disable reporting of resource usage for resource. Specify *resource_list* as a comma-separated list of resources or resource groups.

This option requires an operand. See OPERANDS.

-D — Disable accounting of the given operand type without closing the accounting file. This option can be used to temporarily stop writing accounting records to the accounting file without closing it. To close the file use the -x option. See -x.

-e *resource_list* — Enable reporting of resource usage for resource. Specify *resource_list* as a comma-separated list of resources or resource groups.

This option requires an operand. See OPERANDS.

-E — Enable accounting of the given operand type without sending the accounting output to a file. This option requires an operand. See OPERANDS.

-f *filename* — Send the accounting output for the given operand type to *filename*. If *filename* exists, its contents are lost.

This option requires an operand. See OPERANDS.

-r — Display available resource groups.

When this option is used with an operand, it displays resource groups available for a given accounting type. When no operand is specified, this option displays resource groups for all available accounting types. See OPERANDS.

-u — Configure accounting based on the contents of /etc/acctadm.conf.

-x — Deactivate accounting of the given operand type. This option also closes the accounting file for the given accounting type if it is currently open.

This option requires an operand. See OPERANDS.

OPERANDS

The -d, -D, -e, -E, -f, and -x options require an operand.

The following operands are supported:

process Run `acctadm` on the process accounting components of the extended account facility.

task Run `acctadm` on the task accounting components of the extended account facility.

The optional final parameter to `acctadm` represents whether the command should act on the process or system task accounting components of the extended account facility.

EXAMPLES

EXAMPLE 1 Displaying current status

The following command displays the current status. In this example, system task accounting is active and tracking only CPU resources. Process accounting is not active.

```
$ acctadm
           Task accounting: active
      Task accounting file: /var/adm/exacct/task
    Tracked task resources: extended,mstate
  Untracked task resources: host
         Process accounting: inactive
   Process accounting file: none
 Tracked process resources: none
Untracked process resources: extended,host,mstate
```

EXAMPLE 2 Activating basic process accounting

The following command activates basic process accounting:

```
$ acctadm -e basic -f /var/adm/exacct/proc process
```

EXAMPLE 3 Displaying available resource groups

The following command displays available resource groups:

```
$ acctadm -r
  process:
  extended pid,uid,gid,cpu,time,command,tty,projid,taskid,ancpid,
           wait-status,flag
  basic    pid,uid,gid,cpu,time,command,tty,flag
  task:
  extended taskid,projid,cpu,time,host,mstate,anctaskid
  basic    taskid,projid,cpu,time
```

EXAMPLE 4 Displaying resource groups for task accounting

The following command displays resource groups for task accounting:

```
$ acctadm -r task
  extended taskid,projid,cpu,time,host,mstate,anctaskid
  basic    taskid,projid,cpu,time
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

The modifications to the current configuration were valid and made successfully.

`1` An error occurred.

A fatal error occured either in obtaining or modifying the accounting configuration.

`2` Invalid command line options were specified.

FILES

`/etc/acctadm.conf`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`acct`(2), `attributes`(5)

NOTES

Both extended accounting and regular accounting can be active.

Available resources can vary from system to system, and from platform to platform.

NAME

acctcms – command summary from process accounting records

SYNOPSIS

/usr/lib/acct/acctcms [-a [-o] [-p]] [-c] [-j] [-n] [-s] [-t] *filename...*

DESCRIPTION

acctcms reads one or more *filename*s, normally in the form described in acct(3HEAD). It adds all records for processes that executed identically named commands, sorts them, and writes them to the standard output, normally using an internal summary format.

OPTIONS

-a Print output in ASCII rather than in the internal summary format. The output includes command name, number of times executed, total kcore-minutes, total CPU minutes, total real minutes, mean size (in K), mean CPU minutes per invocation, "hog factor," characters transferred, and blocks read and written, as in acctcom(1). Output is normally sorted by total kcore-minutes.

Use the following options only with the -a option:

-o Output a (non-prime) offshift-time-only command summary.

-p Output a prime-time-only command summary.

When -o and -p are used together, a combination prime-time and non-prime-time report is produced. All the output summaries are total usage except number of times executed, CPU minutes, and real minutes, which are split into prime and non-prime.

-c Sort by total CPU time, rather than total kcore-minutes.

-j Combine all commands invoked only once under "***other".

-n Sort by number of command invocations.

-s Any file names encountered hereafter are already in internal summary format.

-t Process all records as total accounting records. The default internal summary format splits each field into prime and non-prime-time parts. This option combines the prime and non-prime time parts into a single field that is the total of both, and provides upward compatibility with old style acctcms internal summary format records.

EXAMPLES

EXAMPLE 1 Using the acctcms command.

A typical sequence for performing daily command accounting and for maintaining a running total is:

```
example% acctcms filename ... > today
example% cp total previoustotal
example% acctcms -s today previoustotal > total
example% acctcms -a -s today
```

EXAMPLE 1 Using the `acctcms` command. *(Continued)*

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWaccu

SEE ALSO

`acctcom`(1), `acct`(1M), `acctcon`(1M), `acctmerg`(1M), `acctprc`(1M), `acctsh`(1M), `fwtmp`(1M), `runacct`(1M), `acct`(2), `acct`(3HEAD), `utmpx`(4), `attributes`(5)

NOTES

Unpredictable output results if `-t` is used on new style internal summary format files, or if it is not used with old style internal summary format files.

NAME

acctcon, acctcon1, acctcon2 – connect-time accounting

SYNOPSIS

/usr/lib/acct/acctcon [-l *lineuse*] [-o *reboot*]

/usr/lib/acct/acctcon1 [-p] [-t] [-l *lineuse*] [-o *reboot*]

/usr/lib/acct/acctcon2

DESCRIPTION

acctcon converts a sequence of login/logoff records to total accounting records (see the tacct format in acct(3HEAD)). The login/logoff records are read from standard input. The file /var/adm/wtmpx is usually the source of the login/logoff records; however, because it might contain corrupted records or system date changes, it should first be fixed using wtmpfix. The fixed version of file /var/adm/wtmpx can then be redirected to acctcon. The tacct records are written to standard output.

acctcon is a combination of the programs acctcon1 and acctcon2. acctcon1 converts login/logoff records, taken from the fixed /var/adm/wtmpx file, to ASCII output. acctcon2 reads the ASCII records produced by acctcon1 and converts them to tacct records. acctcon1 can be used with the -l and -o options, described below, as well as with the -p and -t options.

OPTIONS

-p — Print input only, showing line name, login name, and time (in both numeric and date/time formats).

-t — acctcon1 maintains a list of lines on which users are logged in. When it reaches the end of its input, it emits a session record for each line that still appears to be active. It normally assumes that its input is a current file, so that it uses the current time as the ending time for each session still in progress. The -t flag causes it to use, instead, the last time found in its input, thus assuring reasonable and repeatable numbers for non-current files.

-l *lineuse* — *lineuse* is created to contain a summary of line usage showing line name, number of minutes used, percentage of total elapsed time used, number of sessions charged, number of logins, and number of logoffs. This file helps track line usage, identify bad lines, and find software and hardware oddities. Hangup, termination of login(1) and termination of the login shell each generate logoff records, so that the number of logoffs is often three to four times the number of sessions. See init(1M) and utmpx(4).

-o *reboot* — reboot is filled with an overall record for the accounting period, giving starting time, ending time, number of reboots, and number of date changes.

EXAMPLES

EXAMPLE 1 Using the acctcon command.

The acctcon command is typically used as follows:

```
example% acctcon -l lineuse -o reboots < tmpwtmp > ctacct
```

The acctcon1 and acctcon2 commands are typically used as follows:

EXAMPLE 1 Using the `acctcon` command. *(Continued)*

```
example% acctcon1 -l lineuse -o reboots < tmpwtmp | sort +1n +2 > ctmp
example% acctcon2 < ctmp > ctacct
```

FILES

`/var/adm/wtmpx` History of user access and administration information

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWaccu

SEE ALSO

`acctcom`(1), `login`(1), `acct`(1M), `acctcms`(1M), `acctmerg`(1M), `acctprc`(1M), `acctsh`(1M), `fwtmp`(1M), `init`(1M), `runacct`(1M), `acct`(2), `acct`(3HEAD), `utmpx`(4), `attributes`(5)

System Administration Guide: Basic Administration

NOTES

The line usage report is confused by date changes. Use `wtmpfix` (see `fwtmp`(1M)), with the `/var/adm/wtmpx` file as an argument, to correct this situation.

During a single invocation of any given command, the `acctcon`, `acctcon1`, and `acctcon2` commands can process a maximum of:

- 6000 distinct session
- 1000 distinct terminal lines
- 2000 distinct login names

If at some point the actual number of any one of these items exceeds the maximum, the command will not succeed.

NAME

acctmerg – merge or add total accounting files

SYNOPSIS

/usr/lib/acct/acctmerg [-a] [-i] [-p] [-t] [-u] [-v] [*filename*] ...

DESCRIPTION

acctmerg reads its standard input and up to nine additional files, all in the tacct format (see acct(3HEAD)) or an ASCII version thereof. It merges these inputs by adding records whose keys (normally user ID and name) are identical, and expects the inputs to be sorted on those keys.

OPTIONS

-a Produce output in ASCII version of tacct.

-i Produce input in ASCII version of tacct.

-p Print input with no processing.

-t Produce a single record that totals all input.

-u Summarize by user ID, rather than by user ID and name.

-v Produce output in verbose ASCII format, with more precise notation for floating-point numbers.

EXAMPLES

EXAMPLE 1 Using the acctmerg command.

The following sequence is useful for making "repairs" to any file kept in this format:

```
example% acctmerg   -v  <filename1  >filename2
```

Edit *filename2* as you want:

```
example% acctmerg   -i  <filename2  >filename1
```

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWaccu

SEE ALSO

acctcom(1), acct(1M), acctcms(1M), acctcon(1M), acctprc(1M), acctsh(1M), fwtmp(1M), runacct(1M), acct(2), acct(3HEAD), utmpx(4), attributes(5)

System Administration Guide: Basic Administration

NAME

acctprc, acctprc1, acctprc2 – process accounting

SYNOPSIS

/usr/lib/acct/acctprc

/usr/lib/acct/acctprc1 [*ctmp*]

/usr/lib/acct/acctprc2

DESCRIPTION

`acctprc` reads the standard input, in the form described by , and converts it to total accounting records (see the `tacct` record in `acct`(3HEAD)). `acctprc` divides CPU time into prime time and non-prime time and determines mean memory size (in memory segment units). `acctprc` then summarizes the `tacct` records, according to user IDs, and adds login names corresponding to the user IDs. The summarized records are then written to the standard output. `acctprc1` reads input in the form described by `acct`(3HEAD), adds login names corresponding to user IDs, then writes for each process an ASCII line giving user ID, login name, prime CPU time (tics), non-prime CPU time (tics), and mean memory size (in memory segment units). If *ctmp* is given, it should contain a list of login sessions sorted by user ID and login name. If this file is not supplied, it obtains login names from the password file, just as `acctprc` does. The information in *ctmp* helps it distinguish between different login names that share the same user ID.

From the standard input, `acctprc2` reads records in the form written by `acctprc1`, summarizes them according to user ID and name, then writes the sorted summaries to the standard output as total accounting records.

EXAMPLES

EXAMPLE 1 Examples of `acctprc`.

The `acctprc` command is typically used as shown below:

```
example% acctprc  < /var/adm/pacct  > ptacct
```

The `acctprc1` and `acctprc2s` commands are typically used as shown below:

```
example% acctprc1  ctmp  </var/adm/pacct
example% acctprc2 > ptacct
```

FILES

`/etc/passwd` system password file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWaccu

SEE ALSO

`acctcom`(1), `acct`(1M), `acctcms`(1M), `acctcon`(1M), `acctmerg`(1M), `acctsh`(1M), `cron`(1M), `fwtmp`(1M), `runacct`(1M), `acct`(2), `acct`(3HEAD), `utmpx`(4), `attributes`(5)

NOTES

Although it is possible for `acctprc1` to distinguish among login names that share user IDs for commands run from a command line, it is difficult for `acctprc1` to make this distinction for commands invoked in other ways. A command run from `cron`(1M) is an example of where `acctprc1` might have difficulty. A more precise conversion can be done using the `acctwtmp` program in `acct`(1M). `acctprc` does not distinguish between users with identical user IDs.

A memory segment of the mean memory size is a unit of measure for the number of bytes in a logical memory segment on a particular processor.

During a single invocation of any given command, the `acctprc`, `acctprc1`, and `acctprc2` commands can process a maximum of

- 6000 distinct sessions
- 1000 distinct terminal lines
- 2000 distinct login names

If at some point the actual number of any one of these items exceeds the maximum, the command will not succeed.

NAME

acctsh, chargefee, ckpacct, dodisk, lastlogin, monacct, nulladm, prctmp, prdaily, prtacct, shutacct, startup, turnacct – shell procedures for accounting

SYNOPSIS

/usr/lib/acct/chargefee *login-name number*

/usr/lib/acct/ckpacct [*blocks*]

/usr/lib/acct/dodisk [-o] [*filename...*]

/usr/lib/acct/lastlogin

/usr/lib/acct/monacct *number*

/usr/lib/acct/nulladm *filename...*

/usr/lib/acct/prctmp *filename*

/usr/lib/acct/prdaily [-c] [-l] [*mmdd*]

/usr/lib/acct/prtacct *filename* ['' *heading* ''']

/usr/lib/acct/shutacct ['' *reason* ''']

/usr/lib/acct/startup

/usr/lib/acct/turnacct on | off | switch

chargefee Command

`chargefee` can be invoked to charge a *number* of units to *login-name*. A record is written to `/var/adm/fee`, to be merged with other accounting records by `runacct`(1M).

ckpacct Command

`ckpacct` should be initiated using `cron`(1M) to periodically check the size of `/var/adm/pacct`. If the size exceeds *blocks*, `500` by default, `turnacct` will be invoked with argument `switch`. To avoid a conflict with `turnacct switch` execution in `runacct`, do not run `ckpacct` and `runacct` simultaneously. If the number of free disk blocks in the `/var` file system falls below `500`, `ckpacct` will automatically turn off the collection of process accounting records via the `off` argument to `turnacct`. When at least `500` blocks are restored, the accounting will be activated again on the next invocation of `ckpacct`. This feature is sensitive to the frequency at which `ckpacct` is executed, usually by `cron`.

dodisk Command

`dodisk` should be invoked by `cron` to perform the disk accounting functions.

lastlogin Command

`lastlogin` is invoked by `runacct`(1M) to update `/var/adm/acct/sum/loginlog`, which shows the last date on which each person logged in.

monacct Command

`monacct` should be invoked once each month or each accounting period. *number* indicates which month or period it is. If *number* is not given, it defaults to the current month (01–12). This default is useful if `monacct` is to executed using `cron`(1M) on the first day of each month. `monacct` creates summary files in `/var/adm/acct/fiscal` and restarts the summary files in `/var/adm/acct/sum`.

nulladm Command

`nulladm` creates *filename* with mode 664 and ensures that owner and group are `adm`. It is called by various accounting shell procedures.

prctmp Command

`prctmp` can be used to print the session record file (normally `/var/adm/acct/nite/ctmp` created by `acctcon1` (see `acctcon`(1M)).

prdaily Command

`prdaily` is invoked by `runacct`(1M) to format a report of the previous day's accounting data. The report resides in `/var/adm/acct/sum/rprt/mmdd` where *mmdd* is the month and day of the report. The current daily accounting reports may be printed by typing `prdaily`. Previous days' accounting reports can be printed by using the *mmdd* option and specifying the exact report date desired.

prtacct Command

`prtacct` can be used to format and print any total accounting (`tacct`)file.

shutacct Command

`shutacct` is invoked during a system shutdown to turn process accounting off and append a *reason* record to `/var/adm/wtmpx`.

startup Command

`startup` can be invoked when the system is brought to a multi-user state to turn process accounting on.

turnacct Command

`turnacct` is an interface to `accton` (see `acct`(1M)) to turn process accounting `on` or `off`. The `switch` argument moves the current `/var/adm/pacct` to the next free name in `/var/adm/pacct`*incr* (where *incr* is a number starting with `1` and incrementing by one for each additional `pacct` file), then turns accounting back on again. This procedure is called by `ckpacct` and thus can be taken care of by the `cron` and used to keep `pacct` to a reasonable size. `shutacct` uses `turnacct` to stop process accounting. `startup` uses `turnacct` to start process accounting.

OPTIONS

The following options are supported:

`-c` This option prints a report of exceptional resource usage by command, and may be used on current day's accounting data only.

`-l` This option prints a report of exceptional usage by login id for the specified date. Previous daily reports are cleaned up and therefore inaccessible after each invocation of `monacct`.

`-o` This option uses `acctdusg` (see `acct`(1M)) to do a slower version of disk accounting by login directory. *filename*s specifies the one or more filesystem names where disk accounting will be done. If *filename*s are used, disk accounting will be done on these filesystems only. If the `-o` option is used, *filename*s should be mount points of mounted filesystems. If the `-o` option is omitted, *filename*s should be the special file names of mountable filesystems.

FILES

`/usr/lib/acct`
holds all accounting commands listed in section 1M of this manual

`/usr/lib/acct/ptecms.awk`
contains the limits for exceptional usage by command name

`/usr/lib/acct/ptelus.awk`
contains the limits for exceptional usage by login ID

`/var/adm/acct/fiscal`
fiscal reports directory

`/var/adm/acct/nite`
working directory

`/var/adm/acct/sum`
summary directory contains information for `monacct`

`/var/adm/acct/sum/loginlog`
file updated by last login

`/var/adm/fee`
accumulator for fees

`/var/adm/pacct`
current file for per-process accounting

`/var/adm/pacct`*incr*
used if `pacct` gets large and during execution of daily accounting procedure

`/var/adm/wtmpx`
history of user access and administration information

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWaccu

SEE ALSO

`acctcom`(1), `acct`(1M), `acctcms`(1M), `acctcon`(1M), `acctmerg`(1M), `acctprc`(1M), `cron`(1M), `fwtmp`(1M), `runacct`(1M), `acct`(2), `acct`(3HEAD), `utmpx`(4), `attributes`(5)

NOTES

See `runacct`(1M) for the main daily accounting shell script, which performs the accumulation of connect, process, fee, and disk accounting on a daily basis. It also creates summaries of command usage.

NAME

adbgen – generate adb script

SYNOPSIS

/usr/lib/adb/adbgen [-m *model*] *filename*.adb . . .

DESCRIPTION

adbgen makes it possible to write adb(1) scripts that do not contain hard-coded dependencies on structure member offsets. The input to adbgen is a file named *filename*.adb that contains header information, then a null line, then the name of a structure, and finally an adb script. adbgen only deals with one structure per file; all member names are assumed to be in this structure. The output of adbgen is an adb script in *filename*. adbgen operates by generating a C program which determines structure member offsets and sizes, which in turn generate the adb script.

The header lines, up to the null line, are copied verbatim into the generated C program. Typically, these are #include statements, which include the headers containing the relevant structure declarations.

The adb script part may contain any valid adb commands (see adb(1)), and may also contain adbgen requests, each enclosed in braces ({ }). Request types are:

- Print a structure member. The request form is {*member*, *format*} . *member* is a member name of the *structure* given earlier, and format is any valid adb format request or any of the adbgen format specifiers (such as {POINTER}) listed below. For example, to print the p_pid field of the *proc* structure as a decimal number, you would write {p_pid,d}.
- Print the appropriate adb format character for the given adbgen format specifier. This action takes the data model into consideration. The request form is {*format specifier*} . The valid adbgen format specifiers are:

{POINTER}	pointer value in hexadecimal
{LONGDEC}	long value in decimal
{ULONGDEC}	unsigned long value in decimal
{ULONGHEX}	unsigned long value in hexadecimal
{LONGOCT}	long value in octal
{ULONGOCT}	unsigned long value in octal

- Reference a structure member. The request form is {**member*, *base*} . *member* is the member name whose value is desired, and *base* is an adb register name which contains the base address of the structure. For example, to get the p_pid field of the *proc* structure, you would get the *proc* structure address in an adb register, for example <f, and write {*p_pid,<f}.
- Tell adbgen that the offset is valid. The request form is {OFFSETOK}. This is useful after invoking another adb script which moves the adb *dot.*
- Get the size of the *structure*. The request form is {SIZEOF}. adbgen replaces this request with the size of the structure. This is useful in incrementing a pointer to step through an array of structures.

- Calculate an arbitrary C expression. The request form is {EXPR, *expression*}. adbgen replaces this request with the value of the expression. This is useful when more than one structure is involved in the script.
- Get the offset to the end of the structure. The request form is {END}. This is useful at the end of the structure to get adb to align the *dot* for printing the next structure member.

adbgen keeps track of the movement of the adb *dot* and generates adb code to move forward or backward as necessary before printing any structure member in a script. adbgen's model of the behavior of adb's *dot* is simple: it is assumed that the first line of the script is of the form *struct_address*/*adb text* and that subsequent lines are of the form +/*adb text*. The adb *dot* then moves in a sane fashion. adbgen does not check the script to ensure that these limitations are met. adbgen also checks the size of the structure member against the size of the adb format code and warns if they are not equal.

OPTIONS

The following option is supported:

-m *model* — Specifies the data type model to be used by adbgen for the macro. This affects the outcome of the {*format specifier*} requests described under DESCRIPTION and the offsets and sizes of data types. *model* can be ilp32 or lp64. If the -m option is not given, the data type model defaults to ilp32.

OPERANDS

The following operand is supported:

filename.adb — Input file that contains header information, followed by a null line, the name of the structure, and finally an adb script.

EXAMPLES

EXAMPLE 1 A sample adbgen file.

For an include file x.h which contained

```
struct x {
        char    *x_cp;
        char    x_c;
        int     x_i;
};
```

then , an adbgen file (call it script.adb) to print the file x.h would be:

```
#include "x.h"
x
./"x_cp"16t"x_c"8t"x_i"n{x_cp,{POINTER}}{x_c,C}{x_i,D}
```

After running adbgen as follows,

```
% /usr/lib/adb/adbgen  script.adb
```

the output file script contains:

```
./"x_cp"16t"x_c"8t"x_i"nXC3+D
```

EXAMPLE 1 A sample `adbgen` file. *(Continued)*

For a macro generated for a 64-bit program using the `lp64` data model as follows,

```
% /usr/lib/adb/adbgen/ -m lp64  script.adb
```

the output file `script` would contain:

```
./"x_cp"16t"x_c"8t"x_i"nJC3+D
```

To invoke the script, type:

```
example% adb program
x$<script
```

FILES

`/usr/platform/`*platform-name*`/lib/adb/*`
platform-specific `adb` scripts for debugging the 32-bit kernel

`/usr/platform/`*platform-name*`/lib/adb/sparcv9/*`
platform-specific `adb` scripts for debugging the 64-bit SPARC V9 kernel

`/usr/lib/adb/*`
`adb` scripts for debugging the 32-bit kernel

`/usr/lib/adb/sparcv9/*`
`adb` scripts for debugging the 64-bit SPARC V9 kernel

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWesu

SEE ALSO

`adb`(1), `uname`(1), `kadb`(1M), `attributes`(5)

DIAGNOSTICS

Warnings are given about structure member sizes not equal to `adb` format items and about badly formatted requests. The C compiler complains if a structure member that does not exist is referenced. It also complains about an ampersand before array names; these complaints may be ignored.

NOTES

platform-name can be found using the `-i` option of `uname`(1).

BUGS

`adb` syntax is ugly; there should be a higher level interface for generating scripts.

Structure members which are bit fields cannot be handled because C will not give the address of a bit field. The address is needed to determine the offset.

NAME

addbadsec – map out defective disk blocks

SYNOPSIS

addbadsec [-p] [-a *blkno* [*blkno*...]] [-f *filename*] *raw_device*

DESCRIPTION

addbadsec is used by the system administrator to map out bad disk blocks. Normally, these blocks are identified during surface analysis, but occasionally the disk subsystem reports unrecoverable data errors indicating a bad block. A block number reported in this way can be fed directly into addbadsec, and the block will be remapped. addbadsec will first attempt hardware remapping. This is supported on SCSI drives and takes place at the disk hardware level. If the target is an IDE drive, then software remapping is used. In order for software remapping to succeed, the partition must contain an alternate slice and there must be room in this slice to perform the mapping.

It should be understood that bad blocks lead to data loss. Remapping a defective block does not repair a damaged file. If a bad block occurs to a disk-resident file system structure such as a superblock, the entire slice might have to be recovered from a backup.

OPTIONS

The following options are supported:

-a Adds the specified blocks to the hardware or software map. If more than one block number is specified, the entire list should be quoted and block numbers should be separated by white space.

-f Adds the specified blocks to the hardware or software map. The bad blocks are listed, one per line, in the specified file.

-p Causes addbadsec to print the current software map. The output shows the defective block and the assigned alternate. This option cannot be used to print the hardware map.

OPERANDS

The following operand is supported:

raw_device The address of the disk drive (see FILES).

FILES

The raw device should be /dev/rdsk/c?[t?]d?p0. See disks(1M) for an explanation of SCSI and IDE device naming conventions.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Architecture	IA
Availability	SUNWcsu

SEE ALSO

disks(1M), diskscan(1M), fdisk(1M), fmthard(1M), format(1M), attributes(5)

NOTES The `format`(1M) utility is available to format, label, analyze, and repair SCSI disks. This utility is included with the `addbadsec`, `diskscan`(1M), `fdisk`(1M), and `fmthard`(1M) commands available for IA. To format an IDE disk, use the DOS "format" utility; however, to label, analyze, or repair IDE disks on IA systems, use the Solaris `format`(1M) utility.

NAME

add_drv – add a new device driver to the system

SYNOPSIS

```
add_drv [-b basedir] [-c class_name] [-i 'identify_name...'] [-m
     'permission','...'] [-n] [-f] [-v] device_driver
```

DESCRIPTION

The `add_drv` command is used to inform the system about newly installed device drivers.

Each device on the system has a name associated with it. This name is represented by the `name` property for the device. Similarly, the device may also have a list of driver names associated with it. This list is represented by the `compatible` property for the device.

The system determines which devices will be managed by the driver being added by examining the contents of the `name` property and the `compatible` property (if it exists) on each device. If the value in the `name` property does not match the driver being added, each entry in the `compatible` property is tried, in order, until either a match occurs or there are no more entries in the `compatible` property.

In some cases, adding a new driver may require a reconfiguration boot. See the `NOTES` section.

Aliases might require quoting (with double-quotes) if they contain numbers. See EXAMPLES.

OPTIONS

`-b` *basedir*	Installs the driver on the system with a root directory of `basedir` rather than installing on the system executing `add_drv`. This option is typically used in package post-installation scripts when the package is not being installed on the system executing the `pkgadd` command. The system using `basedir` as its root directory must reboot to complete the driver installation.
`-c` *class_name*	The driver being added to the system exports the class *class_name.*
`-i` *'identify_name'*	A white-space separated list of aliases for the driver *device_driver*.
`-m` *'permission'*	Specify the file system permissions for device nodes created by the system on behalf of *device_driver*.
`-n`	Do not try to load and attach *device_driver*, just modify the system configuration files for the *device_driver*.
`-f`	Normally if a reconfiguration boot is required to complete the configuration of the driver into the system, `add_drv` will not add the driver. The force flag forces `add_drv` to add the driver even if a reconfiguration boot is required. See the `-v` flag.

`-v`	The verbose flag causes `add_drv` to provide additional information regarding the success or failure of a driver's configuration into the system. See the `EXAMPLES` section.

EXAMPLES

EXAMPLE 1 Adding SUNW Example Driver to the System

The following example adds the `SUNW,example` driver to a 32–bit system, with an alias name of `SUNW,alias`. It assumes the driver has already been copied to `/usr/kernel/drv`.

```
example# add_drv -m '* 0666 bin bin','a 0644 root sys' \
        -i 'SUNW,alias' SUNW,example
```

Every minor node created by the system for the `SUNW,example` driver will have the permission `0666`, and be owned by user `bin` in the group `bin`, except for the minor device `a`, which will be owned by `root`, group `sys`, and have a permission of `0644`.

EXAMPLE 2 Adding Driver to the Client `/export/root/sun1`

The following example adds the driver to the client `/export/root/sun1`. The driver is installed and loaded when the client machine, `sun1`, is rebooted. This second example produces the same result as the first, except the changes are on the diskless client, `sun1`, and the client must be rebooted for the driver to be installed.

```
example# add_drv -m '* 0666 bin bin','a 0644 root sys' \
        -i 'SUNW,alias' -b /export/root/sun1 \
    SUNW,example
```

EXAMPLE 3 Adding Driver for a Device Already Managed by an Existing Driver

The following example illustrates the case where a new driver is added for a device that is already managed by an existing driver. Consider a device that is currently managed by the driver `dumb_framebuffer`. The `name` and `compatible` properties for this device are as follows:

```
name="display"
compatible="whizzy_framebuffer", "dumb_framebuffer"
```

If `add_drv` is used to add the `whizzy_framebuffer` driver, the following will result.

```
example# add_drv whizzy_framebuffer
Error: Could not install driver (whizzy_framebuffer)
Device managed by another driver.
```

If the `-v` flag is specified, the following will result.

```
example# add_drv -v whizzy_framebuffer
Error: Could not install driver (whizzy_framebuffer)
Device managed by another driver.
Driver installation failed because the following
entries in /devices would be affected:
```

EXAMPLE 3 Adding Driver for a Device Already Managed by an Existing Driver *(Continued)*

```
        /devices/iommu@f,e0000000/sbus@f,e0001000/display[:*]
        (Device currently managed by driver "dumb_framebuffer")

The following entries in /dev would be affected:

        /dev/fbs/dumb_framebuffer0
```

If the `-v` and `-f` flags are specified, the driver will be added resulting in the following.

```
example# add_drv -vf whizzy_framebuffer
A reconfiguration boot must be performed to complete the
installation of this driver.

The following entries in /devices will be affected:

        /devices/iommu@f,e0000000/sbus@f,e0001000/display[:*]
        (Device currently managed by driver "dumb_framebuffer"

The following entries in /dev will be affected:

        /dev/fbs/dumb_framebuffer0
```

The above example is currently only relevant to devices exporting a generic device name.

EXAMPLE 4 Use of Double Quotes in Specifying Driver Alias

The following example shows the use of double quotes in specifying a driver alias that contains numbers.

```
example# add_drv -i '"pci10c5,25"' smc
```

EXIT STATUS

`add_drv` returns `0` on success and `1` on failure.

FILES

`/kernel/drv`
  32–bit boot device drivers

`/kernel/drv/sparcv9`
  64–bit boot device drivers

`/usr/kernel/drv`
  other 32–bit drivers that could potentially be shared between platforms

`/usr/kernel/drv/sparcv9`
  other 64–bit drivers that could potentially be shared between platforms

`/platform/‘uname -i‘/kernel/drv`
  32–bit platform-dependent drivers

`/platform/‘uname -i‘/kernel/drv/sparcv9`
  64–bit platform-dependent drivers

`/etc/driver_aliases`
driver aliases file

`/etc/driver_classes`
driver classes file

`/etc/minor_perm`
minor node permissions

`/etc/name_to_major`
major number binding

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`boot`(1M), `devlinks`(1M), `disks`(1M), `drvconfig`(1M), `kernel`(1M), `modinfo`(1M), `ports`(1M), `rem_drv`(1M), `tapes`(1M), `driver.conf`(4), `system`(4), `attributes`(5), `ddi_create_minor_node`(9F)

Writing Device Drivers

NOTES

It is possible to add a driver for a device already being managed by a different driver, where the driver being added appears in the device's `compatible` list before the current driver. In such cases, a reconfiguration boot is required (see `boot`(1M) and `kernel`(1M)). After the reconfiguration boot, device nodes in `/devices`, entries in `/dev`, and references to these files may no longer be valid (see the `-v` flag). If a reconfiguration boot would be required to complete the driver installation, `add_drv` will fail unless the `-f` option is specified. See `Example 3` in the `EXAMPLES` section.

BUGS

`add_drv` will accept a pathname for *device_driver*. However, the kernel does not use the pathname; it only uses the final component and searches the internal driver search path for the driver. This can lead to the kernel loading a different driver than expected.

For this reason, it is `not` recommended that you use `add_drv` with a pathname. See `kernel`(1M) for more information on the driver search path.

A future version of `add_drv` will not support full pathnames.

NAME

admintool – system administration with a graphical user interface

SYNOPSIS

/usr/bin/admintool

DESCRIPTION

`admintool` is a graphical user interface that enables you to accomplish several system administration tasks on a local system. Membership in the sysadmin group (gid 14) is used to restrict access to administrative tasks. Members of the sysadmin group can use `admintool` to create, delete, and modify local system files. Non-members have read-only permissions (where applicable).

Help is available by using the Help button.

`admintool` is not the tool for a distributed environment. It is used for local adminstration.

USAGE

`admintool` allows you to do the following tasks:

Manage users	Use `admintool` to add, delete, or modify user accounts. `admintool` makes the appropriate changes to the system's `/etc/passwd` file (see `passwd`(4)).
Manage groups	Use `admintool` to add, delete, or modify groups. `admintool` makes the appropriate changes to the system's `/etc/group` file (see `group`(4)).
Manage hosts	Use `admintool` to add, delete, or modify hosts. `admintool` makes the appropriate changes to the system's `/etc/hosts` file (see `hosts`(4)).
Manage printers	Use `admintool` to add or delete access to a printer, or to modify a system's printer access. `admintool` makes the appropriate changes to the system's `/etc/lp` directory.
Manage serial port services	Use `admintool` to enable or disable serial port services. `admintool` sets up the software services necessary to use a modem or terminal attached to a system's serial port.
Manage software	Use `admintool` to add or remove software. `admintool` adds software from a product CD or on a hard disk to an installed system, or removes software from an installed system.

EXIT STATUS

`admintool` terminates with exit status `0`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWadmap

SEE ALSO

`group`(4), `hosts`(4), `passwd`(4), `attributes`(5)

Solaris Advanced User's Guide

WARNINGS

If you use `admintool` to add a host, your local system and your site uses a network name service such as NIS or NIS+, `admintool` host operations may not have the desired effect. This is because information in the network name service will take precedence over the information in the local `/etc/hosts` file, which is where `admintool` updates information.

NOTES

`admintool` modifies files on the local system, i.e., the system on which you are running `admintool`. `admintool` does not modify or update global networked databases such as NIS or NIS+.

NAME

afbconfig, SUNWafb_config – configure the AFB Graphics Accelerator

SYNOPSIS

/usr/sbin/afbconfig [-dev *device-filename*] [-res *video-mode* [now | try] [noconfirm | nocheck]] [-file machine | system] [-deflinear true | false] [-defoverlay true | false] [-overlayorder first | last] [-expvis enable | disable] [-sov enable | disable] [-maxwinds *n*] [-extovl enable | disable] [-g *gamma-correction-value*] [-gfile *gamma-correction-file*] [-propt] [-prconf] [-defaults]

/usr/sbin/afbconfig [-propt] [-prconf]

/usr/sbin/afbconfig [-help] [-res ?]

DESCRIPTION

`afbconfig` configures the AFB Graphics Accelerator and some of the `X11` window system defaults for AFB.

The following form of `afbconfig` stores the specified options in the `OWconfig` file:

/usr/sbin/afbconfig [-dev*device-filename*] [-res *video-mode* [now | try] [noconfirm | nocheck]] [-file machine | system] [-deflinear true | false] [-defoverlay true | false] [-overlayorderfirst | last] [-expvisenable | disable] [-sov enable | disable] [-maxwinds*n*] [-extovl enable | disable] [-g*gamma-correction-value*] [-gfile*gamma-correction-file*] [-propt] [-prconf] [-defaults]

The options are used to initialize the AFB device the next time the window system is run on that device. Updating options in the `OWconfig` file provides persistence of these options across window system sessions and system reboots.

The following forms of the `afbconfig` command invoke only the `-prconf`, `-propt`, `-help`, and `-res ?` options. None of these options update the `OWconfig` file.

/usr/sbin/afbconfig [-propt] [-prconf]

/usr/sbin/afbconfig [-help] [-res ?]

Additionally, the following invokation of `afbconfig` ignores all other options:

/usr/sbin/afbconfig [-help] [-res ?]

You can only specify options for one AFB device at a time. Specifying options for multiple AFB devices requires multiple invocations of the `afbconfig` command.

Only AFB-specific options can be specified through `afbconfig`. The normal window system options for specifying default depth, visual class and so forth are still specified as device modifiers on the `openwin` command line.

You can also specify the `OWconfig` file that is to be updated. By default, the machine-specific file in the `/etc/openwin` directory tree is updated. The `-file` option can be used to specify an alternate file to use. For example, the system-global OWconfig file in the `/usr/openwin` directory tree can be updated instead.

Both of these standard `OWconfig` files can only be written by root. Consequently, the `afbconfig` program, which is owned by the root user, always runs with `setuid root` permission.

Option Defaults

For a given invocation of `afbconfig` command line if an option does not appear on the command line, the corresponding `OWconfig` option is not updated; it retains its previous value. When the window system is run, if an AFB option has never been specified by way of `afbconfig`, a default value is used. The option defaults are as follows:

`-dev`	/dev/fbs/afb0
`-file`	machine
`-res`	none
`-deflinear`	false
`-defoverlay`	false
`-linearorder`	last
`-overlayorder`	last
`-expvis`	enabled
`-sov`	enabled
`-maxwids`	32
`-extovl`	enabled
`-g`	2.22

The default for the `-res` option of `none` means that when the window system is run the screen resolution is the video mode currently programmed in the device.

This provides compatibility for users who are used to specifying the device resolution through the PROM. On some devices (for example, `GX`) this is the only way of specifying the video mode. This means that the PROM ultimately determines the default AFB video mode.

OPTIONS

The following options are supported:

`-defaults`
Resets all option values to their default values.

`-deflinear` true | false
AFB possesses two types of visuals: linear and nonlinear. Linear visuals are gamma corrected and nonlinear visuals are not. There are two visuals that have both linear and nonlinear versions: 24-bit TrueColor and 8-bit StaticGray.

If `true`, the default visual is set to the linear visual that satisfies other specified default visual selection options (specifically, the Xsun(1) defdepth and defclass options described in the OpenWindows Reference Manual).

If false, or if there is no linear visual that satisfies the other default visual selection options, the non-linear visual specified by these other options are chosen as the default. This option cannot be used when the `-defoverlay` option is present, because AFB doesn't possess a linear overlay visual.

`-defoverlay` true | false
The AFB provides an 8-bit PseudoColor visual whose pixels are disjoint from the rest of the AFB visuals. This is called the overlay visual. Windows created in this visual do not damage windows created in other visuals. The converse, however, is not true. Windows created in other visuals damage overlay windows.

The number of colors available to the windows created using this visual depends on the settings for the `-extovl` option. If the `-extovl` is enabled, extended overlay with 256 opaque color values is available. See `-extovl`. If `-extovl` is disabled, extended overlay is not available and the visual has `256 -maxwids`) number of opaque color values. See `-maxwids`.

If the value of `-defoverlay` is `true`, the overlay visual is made the default visual. If the value of `-defoverlay` is `false`, the nonoverlay visual that satisfies the other default visual selection options, such as `def`, `depth`, and `defclass`, are chosen as the default visual. See the OpenWindows Reference Manual.

Whenever the `defoverlay true` option is used, the default depth and class specified on the `openwin` command line must be 8-bit PseudoColor. If not, a warning message is printed and the `-defoverlay` option is treated as `false`.

The `-defoverlay` option can not be used when the `-deflinear` option specified, because AFB doesn't possess a linear overlay visual.

`-dev` *device-filename*
Specifies the AFB special file. The default is `/dev/fbs/afb0`.

`-expvis` enable | disable
If enabled, activates OpenGL Visual Expansion. Multiple instances of selected visual groups (8-bit PseudoColor, 24-bit TrueColor and so forth) are in the screen visual list.

`-extovl` enable | disable
If enabled, makes extended overlay available. The overlay visuals have 256 opaque colors. The SOV visuals have 255 opaque colors and 1 transparent color.

This option also enables hardware supported transparency, thus provides better performance for windows using the SOV visuals.

`-file` machine | system
Specifies which `OWconfig` file to update. If `machine` is specified, the machine-specific `OWconfig` file in the `/etc/openwin` directory tree is used. If `system` specifies the global `OWconfig` file in the `/usr/openwin` directory tree. If the specified file does not exist, it is created.

`-g` *gamma-correction value*
Allows changing the gamma correction value. All linear visuals provide gamma correction. By default, the *gamma-correction-value* is `2.22`. Any value less than `0` is illegal. The gamma correction value is applied to the linear visual, which then has an effective gamma value of 1.0, which is the value returned by `XSolarisGetVisualGamma`(3). See `XSolarisGetVisualGamma`(3) for a description of that function.

This option can be used while the window system is running. Changing the gamma correction value affects all the windows being displayed using the linear visuals.

`-gfile` *gamma-correction-file*
Loads the gamma correction table from the specified file (*gamma-correction-file*). This file should be formatted to provide the gamma correction values for `R`, `G` and `B` channels on each line. Each of these values should be in hexadecimal format and seperated from each other by at least one space. *gamma-correction-file* should also provide `256` such triplets.

An example of a *gamma-correction-file* follows.

```
0x00 0x00 0x00
0x01 0x01 0x01
0x02 0x02 0x02
...
...
0xff 0xff 0xff
```

Using this option, the gamma correction table can be loaded while the window system is running. The new gamma correction affects all the windows being displayed using the linear visuals. When gamma correction is being done using user specified table, the gamma correction value is undefined. By default, the window system assumes a gamma correction value of `2.22` and loads the gamma table it creates corresponding to this value.

`-help`
Prints a list of the `afbconfig` command line options, along with a brief explanation of each.

`-linearorder` first | last
If `first`, linear visuals come before their non-linear counterparts on the X11 screen visual list for the AFB screen. If `last`, the nonlinear visuals come before the linear ones.

`-maxwids` *n*
Specifies the maximum number of `AFB X` channel pixel values that are reserved for use as window IDs (WIDs). The remainder of the pixel values in overlay colormaps are used for normal `X11` opaque color pixels. The reserved WIDs are allocated on a first-come first- serve basis by 3D graphics windows (such as `XGL`), MBX windows, and windows that have a non-default visual. The `X` channel codes `0` to (`255` - *n*) are opaque color pixels. The `X` channel codes (`255` - *n* + `1`) to `255` are reserved for use as WIDs. Legal values are `1`, `2`, `4`, `8`, `16`, `32`, and `64`.

This option is available only if the `-extovl` is disabled.

`-overlayorder` first | last
If `first`, the depth 8 PseudoColor Overlay visual comes before the non-overlay visual on the X11 screen visual list for the AFB screen. If `last`, the non-overlay visual comes before the overlay one.

`-propt`
Prints the current values of all `AFB` options in the `OWconfig` file specified by the `-file` option for the device specified by the `-dev` option. Prints the values of options as they will be in the `OWconfig` file after the call to `afbconfig` completes.

The following is a typical display:

```
--- OpenWindows Configuration for /dev/fbs/afb0 ---
OWconfig: machine
Video Mode: 1280x1024x76
Default Visual: Non-Linear Normal Visual
Visual Ordering: Linear Visuals are last
                 Overlay Visuals are last
OpenGL Visual Expansion: enabled
Server Overlay Visuals: enabled
Extended Overlay: enabled
Underlay WIDs: 64 (not configurable)
Overlay WIDs: 4 (not configurable)
Gamma Correction Value: 2.220
Gamma Correction Table: Available
```

`-prconf`
Prints the AFB hardware configuration.

The following is a typical display:

```
--- Hardware Configuration for /dev/fbs/afb0 ---
Type: double-buffered AFB with Z-buffer
Board: rev 0 (Horizontal)
Number of Floats: 6
PROM Information: @(#)afb.fth x.xx xx/xx/xx
AFB ID: 0x101df06d
DAC: Brooktree 9070, version 1 (Pac2)
3DRAM: Mitsubishi 130a, version x
EDID Data: Available - EDID version 1 revision x
Monitor Sense ID: 4  (Sun 37x29cm RGB color monitor)
Monitor possible resolutions: 1024x768x77, 1024x800x84, 1
        1152x900x76, 1280x1024x67, 1280x1024x76, 960x680xx108s
Current resolution setting: 1280x1024x76
```

`-sov` enable | disable
If enabled, the root window's SERVER_OVERLAY_VISUALS property are advertised. SOV visuals are exported and their transparent types, values and layers can be retrieved through this property. If disabled, the SERVER_OVERLAY_VISUALS property are not defined and SOV visuals are not exported.

`-res` *video-mode* [now | try [noconfirm | nocheck]]
Specifies the video mode used to drive the monitor connected to the specified AFB device.

The format of these built-in video modes is: *width*x*height*x*rate*, where *width* is the screen width in pixels, *height* is the screen height in pixels, and *rate* is the vertical frequency of the screen refresh.

The `s` suffix of `960x680x112s` and `960x680x108s` means that these are stereo video modes. The `i` suffix of `640x480x60i` and `768x575x50i` designates interlaced video timing. If absent, non-interlaced timing is used.

As a convenience, the `-res` also accepts formats with an at sign (`@`) in front of the refresh rate instead of *n*, (`1280x1024@76`). Some video-modes, supported by AFB, may not be supported by the monitor. The list of video-modes supported by the AFB device and the monitor can be obtained by running `afbconfig` with the `-res ?` option (the third form shown SYNOPSIS).

A list of all possible video-modes supported on AFB follows:

1024x768x60
1024x768x70
1024x768x75
1024x768x77
1024x800x84
1152x900x66
1152x900x76
1280x800x76
1280x1024x60
1280x1024x67
1280x1024x76
960x680x112s (Stereo)
960x680x108s (Stereo)
640x480x60
640x480x60i (Interlaced)
768x575x50i (Interlaced)

For convenience, some of the video-modes supported on the AFB have *symbolic names* defined for them. Instead of the form *width*x*height*x*rate*, one of these names may be supplied as the argument to the `-res` option. The meaning of the symbolic name `none` is that when the window system is run, the screen resolution is the video mode that is currently programmed in the device.

A list of symbolic names for video-modes supported on AFB follows:

Name	Corresponding Video Mode
svga	1024x768x60
1152	1152x900x76
1280	1280x1024x76
stereo	960x680x112s
ntsc	640x480x60i
pal	768x575x50i
none	(see text above)

The `-res` option also accepts the additional, optional arguments immediately following the video mode specification. Any or all of the following arguments can be specified:

noconfirm — Using the `-res` option, the user could potentially put the system into an unusable state, a state where there is no video output. This can happen if there is ambiguity in the monitor sense codes for the particular code read. To reduce the chance of this, the default behavior of `afbconfig` is to print a warning message to this effect and to prompt the user to find out if it is okay to continue. The `noconfirm` option instructs `afbconfig` to bypass this confirmation and to program the requested video mode anyway. This option is useful when `afbconfig` is being run from a shell script.

nocheck — If present, the normal error checking based on the monitor sense code is suspended. The video mode specified by the user is accepted regardless of whether it is appropriate for the currently attached monitor. (This option is useful if a different monitor is to be connected to the AFB device). Use of this option implies `noconfirm` well.

now — Updates the video mode in the `OWconfig` file, and immediately programs the AFB device to display this video mode. This is useful for changing the video mode before starting the window system.

It is inadvisable to use this argument with afbconfig while the configured device is being used (for example, while running the window system); unpredictable results may occur. To run `afbconfig` with the now argument, first bring the window system down. If the now argument is used within a window system session, the video mode is changed immediately, but the width and height of the affected screen won't change until the window system is exited and re-entered again. In addition, the system may not recognize changes in stereo mode. Consequently, this usage is strongly discouraged.

try — If present, the specified video mode is programmed on a trial basis. The user is asked to confirm the video mode by typing `y` within 10 seconds. Or the user may terminate the trial before 10 seconds are up by typing any character. Any character other

than `y` or Return is considered a `no`. The previous video mode is restored and `afbconfig` does not change the video mode in the OWconfig file (other options specified still take effect). If a Return is typed, the user is prompted for a `yes` or `no` answer on whether to keep the new video mode. This option implies the now argument (see the warning note on the now argument).

EXAMPLES

EXAMPLE 1 Switching the monitor type

The following example switches the monitor type to a resolution of `1280 x 1024 at 76 Hz`:

```
example% /usr/sbin/afbconfig -res 1280x1024x76
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWafbcf

SEE ALSO

`mmap`(2), `attributes`(5)

NAME

aliasadm – manipulate the NIS+ aliases map

SYNOPSIS

aliasadm -a *alias expansion* [*options comments*] *optional flags*

aliasadm -c *alias expansion* [*options comments*] [*optional flags*]

aliasadm -d *alias* [*optional flags*]

aliasadm -e *alias* [*optional flags*]

aliasadm -l *alias* [*optional flags*]

aliasadm -m *alias* [*optional flags*]

aliasadm [-I] [-D *domainname*] [-f *filename*] [-M *mapname*]

DESCRIPTION

`aliasadm` makes changes to the alias map.

The alias map is an NIS+ table object with four columns:

`alias`	The name of the alias as a null terminated string.
expansion	The value of the alias as it would appear in a `sendmail /etc/aliases` file.
options	A list of options applicable to this alias. The only option currently supported is `CANON`. With this option, if the user has requested an inverse alias lookup, and there is more than one alias with this expansion, this alias is given preference.
comments	An arbitrary string containing comments about this alias. The `sendmail`(1M) command reads this map in addition to the NIS aliases map and the local `/etc/aliases` database.

OPTIONS

`-a`	Add an alias.
`-c`	Change an alias.
`-d`	Delete an alias.
`-e`	Edit the alias map.
`-I`	Initialize the NIS+ aliases database.
`-l`	List the alias map.
`-m`	Print or match an alias.
`-D` *domainname*	Edit the map in domain `domainname` instead of the current domain.
`-f` *filename*	When editing or listing the database, use *filename* instead of invoking the editor.
`-M` *mapname*	Edit *mapname* instead of *mail_aliases*.

FILES

`/etc/aliases`	mail aliases for the local host in ASCII format

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`sendmail`(1M), `attributes`(5)

NOTES

NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

answerbook2_admin – bring up AnswerBook2 administration tool GUI

SYNOPSIS

/usr/dt/bin/answerbook2_admin [-h]

DESCRIPTION

The AnswerBook2 server product is no longer included with Solaris or the Solaris Documentation CD products. Solaris docmentation is now provided in HTML and PDF format on the Documentation CD and does not require the Answerbook2 server to be viewed.

`answerbook2_admin` brings up the default web browser showing the administration interface for the local AnswerBook2 server.

The administration functionality is also accessible through the AnswerBook2 `Admin` option within the `System_Admin` subset of the Application Manager function on the CDE front panel Applications menu.

If you need an AnswerBook2 server, you can download the AnswerBook2 server software from http://www.sun.com.

OPTIONS

The following option is supported:

`-h` Displays a usage statement.

USAGE

At startup time, `answerbook2_admin` starts up the default web browser (for example, HotJava or Netscape) and displays the URL specified for administering the local AnswerBook2 server (`http://localhost:8888`). If the user has set up administration access control, the web browser prompts for a valid administrator login and password for this document server before displaying the administration tool.

FILES

`/usr/lib/ab2/dweb/data/config/admin_passwd`
File containing *username*: *password*

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	http://www.sun.com

SEE ALSO

`attributes`(5)

NOTES

Once there is an open web browser and access to the AnswerBook2 Administration tool, use its online Help system to find out more about administering the AnswerBook2 server.

NAME

apache – Apache hypertext transfer protocol server overview

DESCRIPTION

`apache` consists of a main server daemon, loadable server modules, some additional support utilities, configuration files, and documentation.

FILES

The `apache` `HTTPD` server is integrated with Solaris.

The following files specify the installation locations for `apache`:

`/etc/apache`	Contains server configuration files. A newly-installed server must be manually configured before use. Typically this involves copying `httpd.conf-example` to the `httpd.conf` file and making local configuration adjustments.
`/usr/apache/bin`	Contains the `httpd` executable as well as other utility programs.
`/usr/apache/htdocs`	Contains the Apache manual in HTML format. This documentation is accessible by way of a link on the server test page that gets installed upon fresh installation.
`/usr/apache/include`	Contains the Apache header files, which are needed for building various optional server extensions with `apxs`(8)
/usr/apache/jserv	Contains documention for the `mod_jserv` java servlet module. Documention can be read with a web browser using the url: `file:/usr/apache/jserv/docs/index.html`
`/usr/apache/libexec`	Contains loadable modules (`DSO`s) supplied with the server. Any modules which are added using `apxs`(8)are also copied into this directory.
`/usr/apache/man`	Contains man pages for the server, utility programs, and `mod_perl`. Add this directory to your `MANPATH` to read the Apache man pages. See `NOTES`.
`/usr/apache/perl5`	Contains the modules and library files used by the `mod_perl` extension to Apache.
`/var/apache/cgi-bin`	Default location for the CGI scripts. This can be changed by altering the `httpd.conf` file and restarting the server.
`/var/apache/htdocs`	Default document root.

	This can be changed by altering the `httpd.conf` file and restarting the server.
`/var/apache/icons`	Icons used by the server. This normally shouldn't need to be changed.
`/var/apache/logs`	Contains server log files. The formats, names, and locations of the files in this directory can be altered by various configuration directives in the `httpd.conf` file.
`/var/apache/proxy`	Directory used to cache pages if the caching feature of `mod_proxy` is enabled in the `httpd.conf` file. The location of the cache can also be changed by changing the proxy configuration in the `httpd.conf` file.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWapchr SUNWapchu SUNWapchd

SEE ALSO

`attributes`(5)

http://www.apache.org

NOTES

In addition to the documentation and man pages included with Solaris, more information is available at http://www.apache.org

The Apache man pages are provided with the programming modules. To view the manual pages for the Apache modules with the man command, add `/usr/apache/man` to the `MANPATH` environment variable. See `man`(1) for more information. Running `catman`(1M) on the Apache manual pages is not supported.

NAME

arp – address resolution display and control

SYNOPSIS

arp *hostname*

arp `-a [-n]`

arp `-d` *hostname*

arp `-f` *filename*

arp `-s` *hostname ether_address* `[temp] [pub] [trail]`

DESCRIPTION

The `arp` program displays and modifies the Internet-to-Ethernet address translation tables used by the address resolution protocol (see `arp`(7P)).

With no flags, the program displays the current ARP entry for *hostname*. The host may be specified by name or by number, using Internet dot notation.

OPTIONS

`-a` Display all of the current ARP entries. The definition for the flags in the table are:

- `P` Publish; includes IP address for the machine and the addresses that have explicitly been added by the `-s` option. ARP will respond to ARP requests for this address.
- `S` Static; not learned for the ARP protocol.
- `U` Unresolved; waiting for ARP response.
- `M` Mapping; only used for the multicast entry for `224.0.0.0`

You can use the `-n` option with the `-a` option to disable the automatic numeric IP address-to-name translation. Use `arp -an` or `arp -na` to display numeric IP addresses.

`-d` Delete an entry for the host called *hostname*. This option may only be used by the super-user.

`-f` Read the file named *filename* and set multiple entries in the ARP tables. Entries in the file should be of the form

hostname ether_address [`temp`] [`pub`] [`trail`]

(see option `-s` for argument definitions).

`-s` Create an ARP entry for the host called *hostname* with the Ethernet address *ether_address*. The Ethernet address is given as six hexadecimal bytes separated by colons. The entry will be permanent unless the word `temp` is given in the command. If the word `pub` is given, the entry will be published. For instance, this system will respond to ARP requests for *hostname* even though the hostname is not its own. The word `trail` indicates that trailer encapsulations may be sent to this host. `arp -s` can be used for a limited form of proxy `ARP` when a host on one of the directly attached networks is not physically present on the subnet. Another

machine can then be configured to respond to ARP requests using arp -s. This is useful in certain SLIP or PPP configurations.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

ifconfig(1M), arp(7P), attributes(5)

NAME

aset – monitors or restricts accesses to system files and directories

SYNOPSIS

aset [-p] [-d *aset_dir*] [-l *sec_level*] [-n *user@host*] [-u *userlist_file*]

DESCRIPTION

The Automated Security Enhancement Tool (ASET) is a set of administrative utilities that can improve system security by allowing the system administrators to check the settings of system files, including both the attributes (permissions, ownership, and the like) and the contents of the system files. It warns the users of potential security problems and, where appropriate, sets the system files automatically according to the security level specified.

The security level for `aset` can be specified by setting the `-l` command line option or the `ASETSECLEVEL` environment variable to be one of 3 values: `low`, `med`, or `high`. All the functionality operates based on the value of the security level.

At the `low` level, `aset` performs a number of checks and reports any potential security weaknesses.

At the `med` level, `aset` modifies some of the settings of system files and parameters, thus restricting system access, to reduce the risks from security attacks. Again reports the security weaknesses and the modifications performed to restrict access. This does not affect the operations of system services. All the system applications and commands maintain all of their original functionality.

At the `high` level, further restrictions are made to system access, rendering a very defensive system. Security practices which are not normally required are included. Many system files and parameters settings are modified to minimum access permissions. At this level, security is the foremost concern, higher than any other considerations that affect system behavior. The vast majority of system applications and commands maintain their functionality, although there may be a few that exhibit behaviors that are not familiar in normal system environment.

More exact definitions of what exactly `aset` does at each level can be found in the *System Administration Guide: Basic Administration*. The `asetenv`(4) file and the *master files* determine to a large extent what `aset` performs at each level, and can be used by the experienced administrators to redefine the definitions of the levels to suit their particular needs. See `asetmasters`(4). These files are provided by default to fit most security conscious environments and in most cases provide adequate security safeguards without modification. They are, however, designed in a way that can be easily edited by experienced administrators with specific needs.

`aset` can be periodically activated at the specified security level with default definitions using the `-p` option. `aset` is automatically activated at a frequency specified by the administrator starting from a designated future time (see `asetenv`(4)). Without the `-p` option, `aset` operates only once immediately.

OPTIONS

The following options are supported:

`-d` *aset_dir* — Specifies a working directory other than `/usr/aset` for ASET. `/usr/aset` is the default working directory.

It is where ASET is installed, and is the *root* directory of all ASET utilities and data files. If another directory is to be used as the ASET working directory, you can either define it with the `-d` option, or set the `ASETDIR` environment variable before invoking `aset`. The command line option, if specified, overwrites the environment variable.

`-l` *sec_level*
: Specifies a security level, `low`, `med`, or `high`, for `aset` to operate at. The default level is `low`. Each security level is explained in detail above. The level can also be specified by setting the `ASETSECLEVEL` environment variable before invoking `aset`. The command line option, if specified, overwrites the environment variable.

`-n` *user@host*
: Notifies *user* at machine *host*. Send the output of `aset` to *user* through e-mail. If this option is not specified, the output is sent to the standard output. Note that this is not the reports of ASET, but rather an execution log including error messages if there are any. This output is typically brief. The actual reports of ASET are found in the `/usr/aset/reports/latest` directory. See the `-d` option.

`-p`
: Schedules `aset` to be executed periodically. This adds an entry for `aset` in the `/etc/crontab` file. The `PERIODIC_SCHEDULE` environment variable in the `/usr/aset/asetenv` file is used to define the time for execution. See `crontab`(1) and `asetenv`(4). If a `crontab` (1) entry for `aset` already exists, a warning is produced in the execution log.

`-u` *userlist_file*
: Specifies a file containing a list of users. `aset` performs environment checks, for example, `UMASK` and `PATH` variables, on these users. By default, `aset` only checks for `root`. *userlist_file* is an ASCII text file. Each entry in the file is a line that contains only one user name (login name).

USAGE

The following paragraphs discuss the features provided by ASET. Hereafter, each feature is referred to as a *task*. The first task, `tune`, is executed only once per installation of ASET. The other tasks are executed periodically at the specified frequency.

tune Task

This task is used to tighten system file permissions. In standard releases, system files or directories have permissions defined to maximize open information sharing. In a more security conscious environment, the administrator may want to redefine these permission settings to more restrictive values. `aset` allows resetting of these

permissions, based on the specified security level. Generally, at the low level the permissions are set to what they should be as released. At the medium level, the permissions are tightened to ensure reasonable security that is adequate for most environments. At the high level they are further tightened to very restrictive access. The system files affected and the respective restrictions at different levels are configurable, using the `tune.low`, `tune.med`, and `tune.high` files. See `asetmasters`(4).

cklist Task

System directories that contain relatively static files, that is, their contents and attributes do not change frequently, are examined and compared with a master description file. The `/usr/aset/masters/cklist.`*level* files are automatically generated the first time the `cklist` task is executed. See `asetenv`(4). Any discrepancy found is reported. The directories and files are compared based on the following:

- owner and group
- permission bits
- size and checksum (if file)
- number of links
- last modification time

The lists of directories to check are defined in `asetenv`(4), based on the specified security level, and are configurable using the `CKLISTPATH_LOW` , `CKLISTPATH_MED` , and `CKLISTPATH_HIGH` environment variables. Typically, the lower level lists are subsets of the higher level lists.

usrgrp Task

`aset` checks the consistency and integrity of user accounts and groups as defined in the `passwd` and `group` databases, respectively. Any potential problems are reported. Potential problems for the `passwd` file include:

- `passwd` file entries are not in the correct format.
- User accounts without a password.
- Duplicate user names.
- Duplicate user IDs. Duplicate user IDs are reported unless allowed by the `uid_alias` file. See `asetmasters`(4)).
- Invalid login directories.
- If C2 is enabled, check C2 hidden `passwd` format.

Potential problems for the group file include:

- Group file entries not in the right format.
- Duplicate group names.
- Duplicate group IDs.
- Null group passwords.

`aset` checks the local `passwd` file. If the `YPCHECK` environment variable is set to `true`, `aset` also checks the NIS `passwd` files. See `asetenv`(4). Problems in the NIS passwd file are only reported and not corrected automatically. The checking is done for all three security levels except where noted.

sysconf Task

`aset` checks various system configuration tables, most of which are in the `/etc` directory. `aset` checks and makes appropriate corrections for each system table at all three levels except where noted. The following discussion assumes familiarity with the various system tables. See the manual pages for these tables for further details.

The operations for each system table are:

`/etc/hosts.equiv`

The default file contains a single "+" line, thus making every known host a trusted host, which is not advised for system security. `aset` performs the following operations:

`Low` Warns the administrators about the "+" line.

`Medium`

`High` Warns about and deletes that entry.

`/etc/inetd.conf`

The following entries for system daemons are checked for possible weaknesses.

`tftp`(1) does not do any authentication. `aset` ensures that `in.tftpd`(1M) is started in the right directory on the server and is not running on clients. At the low level, it gives warnings if the mentioned condition is not true. At the medium and high levels it gives warnings, and changes (if necessary) the `in.tftpd` entry to include the `-s /tftpboot` option after ensuring the directory `/tftpboot` exists.

`ps`(1) and `netstat`(1M) provide valuable information to potential system crackers. These are disabled when `aset` is executed at a high security level.

`rexd` is also known to have poor authentication mechanism. `aset` disables `rexd` for medium and high security levels by commenting out this entry. If `rexd` is activated with the `-s` (secure RPC) option, it is not disabled.

`/etc/aliases`

The decode alias of UUCP is a potential security weakness. `aset` disables the alias for medium and high security levels by commenting out this entry.

`/etc/default/login`

The `CONSOLE=` line is checked to allow root login only at a specific terminal depending on the security level:

	`Low`	No action taken.
	`Medium`	
	`High`	Adds the following line to the file: `CONSOLE=/dev/console`
`/etc/vfstab`	`aset` checks for world-readable or writable device files for mounted file systems.	
`/etc/dfs/dfstab`	`aset` checks for file systems that are exported without any restrictions.	
`/etc/ftpd/ftpusers`	At high security level, `aset` ensures `root` is in `/etc/ftpd/ftpusers`, thus disallowing root from logging into `in.ftpd`(1M). If necessary, create `/etc/ftpd/ftpusers`. See `ftpusers`(4).	
`/var/adm/utmpx`	`aset` makes these files not world-writable for the high level (some applications may not run properly with this setting.)	
`/.rhosts`	The usage of a `.rhosts` file for the entire system is not advised. `aset` gives warnings for the low level and moves it to `/.rhosts.bak` for levels medium and high.	

env Task

`aset` checks critical environment variables for `root` and users specified with the `-u` *userlist_file* option by parsing the `/.profile`, `/.login`, and `/.cshrc` files. This task checks the `PATH` variable to ensure that it does not contain '`.`' as a directory, which makes an easy target for *trojan horse* attacks. It also checks that the directories in the `PATH` variable are not world-writable. Furthermore, it checks the `UMASK` variable to ensure files are not created as readable or writable by world. Any problems found by these checks are reported.

eeprom Task

Newer versions of the EEPROM allow specification of a `secure` parameter. See `eeprom`(1M). `aset` recommends that the administrator sets the parameter to `command` for the medium level and to `full` for the high level. It gives warnings if it detects the parameter is not set adequately.

firewall Task

At the high security level, `aset` takes proper measures such that the system can be safely used as a firewall in a network. This mainly involves disabling IP packets forwarding and making routing information invisible. Firewalling provides protection against external access to the network.

ENVIRONMENT VARIABLES

`ASETDIR`	Specify ASET's working directory. Defaults to `/usr/aset`.
`ASETSECLEVEL`	Specify ASET's security level. Defaults to `low`.
`TASKS`	Specify the tasks to be executed by `aset`. Defaults to `all tasks`.

FILES

`/usr/aset/reports`	directory of ASET reports

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWast

SEE ALSO

`crontab`(1), `ps`(1), `tftp`(1), `aset.restore`(1M), `eeprom`(1M), `in.ftpd`(1M), `in.tftpd`(1M), `netstat`(1M), `asetenv`(4), `asetmasters`(4), `ftpusers`(4), `attributes`(5)

System Administration Guide: Basic Administration

NAME

aset.restore – restores system files to their content before ASET is installed

SYNOPSIS

`aset.restore` `[-d` *aset_dir*`]`

DESCRIPTION

`aset.restore` restores system files that are affected by the Automated Security Enhancement Tool (ASET) to their pre-ASET content. When ASET is executed for the first time, it saves and archives the original system files in the `/usr/aset/archives` directory. The `aset.restore` utility reinstates these files. It also deschedules ASET, if it is currently scheduled for periodic execution. See `asetenv`(4).

If you have made changes to system files after running ASET, these changes are lost when you run `aset.restore`. If you want to be absolutely sure that you keep the existing system state, it is recommended that you back-up your system before using `aset.restore`.

You should use `aset.restore`, under the following circumstances:

You want to remove ASET permanently and restore the original system (if you want to deactivate ASET, you can remove it from scheduling).

You are unfamiliar with ASET and want to experiment with it. You can use `aset.restore` to restore the original system state.

When some major system functionality is not working properly and you suspect that ASET is causing the problem; you may want to restore the system to see if the problem persists without ASET.

`aset.restore` requires root privileges to execute.

OPTIONS

The following options are supported:

`-d` *aset_dir* Specify the working directory for ASET. By default, this directory is `/usr/aset`. With this option the `archives` directory will be located under *aset_dir*.

FILES

`/usr/aset/archives` archive of system files prior to executing `aset`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWast

SEE ALSO

`aset`(1M), `asetenv`(4), `attributes`(5)

System Administration Guide: Basic Administration

NAME

audit – control the behavior of the audit daemon

SYNOPSIS

audit -n | -s | -t

DESCRIPTION

The audit command is the general administrator's interface to maintaining the audit trail. The audit daemon may be notified to read the contents of the audit_control(4) file and re-initialize the current audit directory to the first directory listed in the audit_control file or to open a new audit file in the current audit directory specified in the audit_control file as last read by the audit daemon. The audit daemon may also be signaled to close the audit trail and disable auditing.

OPTIONS

-n Signal audit daemon to close the current audit file and open a new audit file in the current audit directory.

-s Signal audit daemon to read audit control file. The audit daemon stores the information internally.

-t Signal audit daemon to close the current audit trail file, disable auditing and die.

DIAGNOSTICS

The audit command will exit with 0 upon success and a positive integer upon failure.

FILES

/etc/security/audit_user

/etc/security/audit_control

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

bsmconv(1M), praudit(1M), audit(2), audit_control(4), audit_user(4), attributes(5)

NOTES

The functionality described in this man page is available only if the Basic Security Module (BSM) has been enabled. See bsmconv(1M) for more information.

This command does not modify a process's preselection mask. It only affects which audit directories are used for audit data storage and to specify the minimum size free.

NAME

auditconfig – configure auditing

SYNOPSIS

auditconfig *option...*

DESCRIPTION

`auditconfig` provides a command line interface to get and set kernel audit parameters.

The functionality described in this man page is available only if the Basic Security Module (BSM) has been enabled. See `bsmconv`(1M) for more information.

OPTIONS

`-aconf`
Set the non-attributable audit mask from the `audit_control`(4) file. For example:

```
# auditconfig -aconf
Configured non-attributable events.
```

`-audit` *event sorf retval string*
This command constructs an audit record for audit event *event* using the process's audit characteristics containing a text token *string*. The return token is constructed from the *sorf* (success/failure flag) and the *retval* (return value). The event is type `char*`, the *sorf* is 0/1 for success/failure, *retval* is an errno value, *string* is type `*char`. This command is useful for constructing an audit record with a shell script. An example of this option:

```
# auditconfig -audit AUE_ftpd 0 0 "test string"
#

audit record from audit trail:
    header,76,2,ftp access,,Fri Dec 08 08:44:02 2000, + 669 msec
    subject,abc,root,other,root,other,104449,102336,235 197121 elbow
    text,test string
    return,success,0
```

`-chkaconf`
Checks the configuration of the non-attributable events set in the kernel against the entries in `audit_control`(4). If the runtime class mask of a kernel audit event does not match the configured class mask, a mismatch is reported.

`-chkconf`
Check the configuration of kernel audit event to class mappings. If the runtime class mask of a kernel audit event does not match the configured class mask, a mismatch is reported.

`-conf`
Configure kernel audit event to class mappings. Runtime class mappings are changed to match those in the audit event to class database file.

`-getasid`
Prints the audit session ID of the current process. For example:

```
# auditconfig -getasid
audit session id = 102336
```

`-getaudit`
Returns the audit characteristics of the current process.

```
# auditconfig -getaudit
audit id = abc(666)
process preselection mask = lo(0x1000,0x1000)
terminal id (maj,min,host) = 235,197121,elbow(129.146.89.77)
audit session id = 102336
```

`-getauid`
Prints the audit ID of the current process. For example:

```
# auditconfig -getauid
audit id = abc(666)
```

`-getcar`
Prints current active root location (anchored from root at system boot). For example:

```
# auditconfig -getcar
current active root = /
```

`-getclass` *event*
Display the preselection mask associated with the specified kernel audit event. *event* is the kernel event number or event name.

`-getcond`
Display the kernel audit condition. The condition displayed is the literal string `auditing` meaning auditing is enabled and turned on (the kernel audit module is constructing and queuing audit records); `noaudit`, meaning auditing is enabled but turned off (the kernel audit module is not constructing and queuing audit records); `disabled`, meaning that the audit module has not been enabled; or `nospace`, meaning there is no space for saving audit records. See `auditon`(2) and `auditd`(1M) for further information.

`-getestate` *event*
For the specified event (string or event number), print out classes *event* has been assigned. For example:

```
# auditconfig -getestate 20
audit class mask for event AUE_REBOOT(20) = 0x800
# auditconfig -getestate AUE_RENAME
audit class mask for event AUE_RENAME(42) = 0x30
```

`-getfsize`
Return the maximum audit file size in bytes and the current size of the audit file in bytes.

`-getkaudit`
Get audit characteristics of machine. For example:

```
# auditconfig -getkaudit
audit id = unknown(-2)
process preselection mask = lo,na(0x1400,0x1400)
terminal id (maj,min,host) = 0,0,(0.0.0.0)
audit session id = 0
```

`-getkmask`
Get non-attributable pre-selection mask for machine. For example:

```
# auditconfig -getkmask
audit flags for non-attributable events = lo,na(0x1400,0x1400)
```

`-getpinfo` *pid*
Display the audit ID, preselection mask, terminal ID, and audit session ID for the specified process.

`-getpolicy`
Display the kernel audit policy.

`-getcwd`
Prints current working directory (anchored from root at system boot). For example:

```
# cd /usr/tmp
# auditconfig -getcwd
current working directory = /var/tmp
```

`-getqbufsz`
Get audit queue write buffer size. For example:

```
# auditconfig -getqbufsz
        audit queue buffer size (bytes) = 1024
```

`-getqctrl`
Get audit queue write buffer size, audit queue `hiwater` mark, audit queue `lowater` mark, audit queue `prod` interval (ticks).

```
# auditconfig -getqctrl
audit queue hiwater mark (records) = 100
audit queue lowater mark (records) = 10
audit queue buffer size (bytes) = 1024
audit queue delay (ticks) = 20
```

`-getqdelay`
Get interval at which audit queue is prodded to start output. For example:

```
# auditconfig -getqdelay
audit queue delay (ticks) = 20
```

`-getqhiwater`
Get high water point in undelivered audit records when audit generation will block. For example:

```
# ./auditconfig -getqhiwater
audit queue hiwater mark (records) = 100
```

`-getqlowater`
Get low water point in undelivered audit records where blocked processes will resume. For example:

```
# auditconfig -getqlowater
audit queue lowater mark (records) = 10
```

`-getstat`
Print current audit statistics information. For example:

```
# auditconfig -getstat
gen nona kern  aud  ctl  enq wrtn wblk rblk drop  tot  mem
910    1  725  184    0  910  910    0  231    0   88   48
```

`-gettid`
Print audit terminal ID for current process. For example:

```
# auditconfig -gettid
terminal id (maj,min,host) = 235,197121,elbow(129.146.89.77)
```

`-lsevent`
Display the currently configured (runtime) kernel and user level audit event information.

`-lspolicy`
Display the kernel audit policies with a description of each policy.

`-setasid` *session-ID* [*cmd*]
Execute shell or *cmd* with specified *session-ID*. For example:

```
# ./auditconfig -setasid 2000 /bin/ksh
#
# ./auditconfig -getpinfo 104485
audit id = abc(666)
process preselection mask = lo(0x1000,0x1000)
terminal id (maj,min,host) = 235,197121,elbow(129.146.89.77)
audit session id = 2000
```

`-setaudit` *audit-ID preselect_flags term-ID session-ID* [*cmd*]
Execute shell or *cmd* with the specified audit characteristics.

`-setauid` *audit-ID* [*cmd*]
Execute shell or *cmd* with the specified *audit–ID*.

`-setclass` *event audit_flag*[*,audit_flag . . .*]
Map the kernel event *event* to the classes specified by *audit_flags*. *event* is an event number or name. An *audit_flag* is a two character string representing an audit class. See `audit_control`(4) for further information.

`-setcond` [`auditing`|`noaudit`|`nospace`]
Set the kernel audit condition to the *condition* specified where *condition* is the literal string `auditing`, indicating auditing should be enabled; `noaudit`, indicating auditing should be disabled; or `nospace`, which forces a no-space condition. (See `-getcond`, above.)

`-setfsize` *size*
Set the maximum size of an audit file to *size* bytes. When the size limit is reached, the audit file is closed and another is started.

`-setkaudit` *IP-address_type IP_address*
Set IP address of machine to specified values. IP-address_type is `ipv6` or `ipv4`.

`-setkmask` *audit_flags*
Set non-attributes selection flags of machine.

`-setpmask` *pid flags*
Set the preselection mask of the specified process. `flags` is the ASCII representation of the flags similar to that in `audit_control`(4).

`-setpolicy` [+|-]*policy_flag*[,*policy_flag ...*]
Set the kernel audit policy. A policy *policy_flag* is literal strings that denotes an audit policy. A prefix of + adds the policies specified to the current audit policies. A prefix of - removes the policies specified from the current audit policies. The following are the valid policy flag strings (`auditconfig -lspolicy` also lists the current valid audit policy flag strings):

`all` Include all policies.

`arge` Include the `execv`(2) system call environment arguments to the audit record. This information is not included by default.

`argv` Include the `execv`(2) system call parameter arguments to the audit record. This information is not included by default.

`cnt` Do not suspend processes when audit resources are exhausted. Instead, drop audit records and keep a count of the number of records dropped. By default, process are suspended until audit resources become available.

`group` Include the supplementary group token in audit records. By default, the group token is not included.

`none` Include no policies.

`path` Add secondary path tokens to audit record. These are typically the pathnames of dynamically linked shared libraries or command interpreters for shell scripts. By default, they are not included.

`trail` Include the trailer token in every audit record. By default, the trailer token is not included.

`seq` Include the sequence token as part of every audit record. By default, the sequence token attaches a sequence number to every audit record.

`-setqbufsz` *buffer_size*
Set the audit queue write buffer size (bytes).

`-setqctrl` *hiwater lowater bufsz interval*
Set the audit queue write buffer size (bytes), hiwater audit record count, lowater audit record count, and wakeup interval (ticks).

`-setqdelay` *interval*
Set the audit queue wakeup interval (ticks). This determines the interval at which the kernel pokes the audit queue, to write audit records to the audit trail.

`-setqhiwater` *hiwater*
Set the number of undelivered audit records in the audit queue at which audit record generation blocks.

`-setqlowater` *lowater*
Set the number of undelivered audit records in the audit queue at which blocked auditing processes unblock.

`-setsmask` *asid flags*
Set the preselection mask of all processes with the specified audit session ID.

`-setstat`
Reset audit statistics counters.

`-setumask` *auid flags*
Set the preselection mask of all processes with the specified audit ID.

EXAMPLES

EXAMPLE 1 A sample `auditconfig` program

```
#
# map kernel audit event number 10 to the "fr" audit class
#
% auditconfig -setclass 10 fr

#
# turn on inclusion of exec arguments in exec audit records
#
% auditconfig -setpolicy +argv
```

EXIT STATUS

`0` Successful completion.

`1` An error occurred.

FILES

`/etc/security/audit_event`

`/etc/security/audit_class`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`auditd`(1M), `bsmconv`(1M), `praudit`(1M), `auditon`(2), `execv`(2), `audit_class`(4), `audit_control`(4), `audit_event`(4), `attributes`(5)

NAME

auditd – audit daemon

SYNOPSIS

/usr/sbin/auditd

DESCRIPTION

The audit daemon controls the generation and location of audit trail files. If auditing is desired, `auditd` reads the `audit_control`(4) file to get a list of directories into which audit files can be written and the percentage limit for how much space to reserve on each filesystem before changing to the next directory.

If `auditd` receives the signal `SIGUSR1`, the current audit file is closed and another is opened. If `SIGHUP` is received, the current audit trail is closed, the `audit_control` file reread, and a new trail is opened. If `SIGTERM` is received, the audit trail is closed and auditing is terminated. The program `audit`(1M) sends these signals and is recommended for this purpose.

Each time the audit daemon opens a new audit trail file, it updates the file `audit_data`(4) to include the correct name.

Auditing Conditions

The audit daemon invokes the program `audit_warn`(1M) under the following conditions with the indicated options:

`audit_warn soft` *pathname*
The file system upon which *pathname* resides has exceeded the minimum free space limit defined in `audit_control`(4). A new audit trail has been opened on another file system.

`audit_warn allsoft`
All available file systems have been filled beyond the minimum free space limit. A new audit trail has been opened anyway.

`audit_warn hard` *pathname*
The file system upon which *pathname* resides has filled or for some reason become unavailable. A new audit trail has been opened on another file system.

`audit_warn allhard` *count*
All available file systems have been filled or for some reason become unavailable. The audit daemon will repeat this call to `audit_warn` every twenty seconds until space becomes available. *count* is the number of times that `audit_warn` has been called since the problem arose.

`audit_warn ebusy`
There is already an audit daemon running.

`audit_warn tmpfile`
The file `/etc/security/audit/audit_tmp` exists, indicating a fatal error.

`audit_warn nostart`
The internal system audit condition is `AUC_FCHDONE`. Auditing cannot be started without rebooting the system.

`audit_warn auditoff`
The internal system audit condition has been changed to not be `AUC_AUDITING` by someone other than the audit daemon. This causes the audit daemon to exit.

`audit_warn postsigterm`
An error occurred during the orderly shutdown of the auditing system.

`audit_warn getacdir`
There is a problem getting the directory list from `/etc/security/audit/audit_control`.

The audit daemon will hang in a sleep loop until this file is fixed.

FILES

`/etc/security/audit/audit_control`

`/etc/security/audit/audit_data`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`audit`(1M), `audit_warn`(1M), `bsmconv`(1M), `praudit`(1M), `auditon`(2), `auditsvc`(2), `audit.log`(4), `audit_control`(4), `audit_data`(4), `attributes`(5)

NOTES

The functionality described in this man page is available only if the Basic Security Module (BSM) has been enabled. See `bsmconv`(1M) for more information.

NAME

auditreduce – merge and select audit records from audit trail files

SYNOPSIS

auditreduce [*options*] [*audit-trail-file...*]

DESCRIPTION

`auditreduce` allows you to select or merge records from audit trail files. Audit files may be from one or more machines.

The merge function merges together audit records from one or more input audit trail files into a single output file. The records in an audit trail file are assumed to be sorted in chronological order (oldest first) and this order is maintained by `auditreduce` in the output file.

Unless instructed otherwise, `auditreduce` will merge the entire audit trail, which consists of all the audit trail files in the directory structure *audit_root_dir*`/*/`files (see `audit_control`(4) for details of the structure of the audit root). Unless stated with the `-R` or `-S` option, *audit_root_dir* defaults to `/etc/security/audit`. By using the file selection options it is possible to select some subset of these files, or files from another directory, or files named explicitly on the command line.

The select function allows audit records to be selected on the basis of numerous criteria relating to the record's content (see `audit.log`(4) for details of record content). A record must meet all of the *record-selection-option* criteria to be selected.

Audit Trail Filename Format

Any audit trail file not named on the command line must conform to the audit trail filename format. Files produced by the audit system already have this format. Output file names produced by `auditreduce` are in this format. It is:

start-time`.` *end-time*`.` *suffix*

where *start-time* is the 14-character timestamp of when the file was opened, *end-time* is the 14-character timestamp of when the file was closed, and *suffix* is the name of the machine which generated the audit trail file, or some other meaningful suffix (e.g., `all`, if the file contains a combined group of records from many machines). The *end-time* may be the literal string `not_terminated`, to indicate that the file is still being written to by the audit system. Timestamps are of the form *yyyymmddhhmmss* (year, month, day, hour, minute, second). The timestamps are in Greenwich Mean Time (GMT).

File Selection Options

The file selection options indicate which files are to be processed and certain types of special treatment.

`-A`
All of the records from the input files will be selected regardless of their timestamp. This option effectively disables the `-a`, `-b`, and `-d` options. This is useful in preventing the loss of records if the `-D` option is used to delete the input files after they are processed. Note, however, that if a record is *not* selected due to another option, then `-A` will not override that.

`-C`
Only process complete files. Files whose filename *end-time* timestamp is `not_terminated` are not processed (such a file is currently being written to by the audit system). This is useful in preventing the loss of records if `-D` is used to delete the input files after they are processed. It does not apply to files specified on the command line.

`-D` *suffix*
Delete input files after they deleted if the entire run is successful. If `auditreduce` detects an error while reading a file, then that file is not deleted. If `-D` is specified, `-A`, `-C` and `-O` are also implied. *suffix* is given to the `-O` option. This helps prevent the loss of audit records by ensuring that all of the records are written, only complete files are processed, and the records are written to a file before being deleted. Note that if both `-D` and `-O` are specified in the command line, the order of specification is significant. The *suffix* associated with the latter specification is in effect.

`-M` *machine*
Allows selection of records from files with *machine* as the filename suffix. If `-M` is not specified, all files are processed regardless of suffix. `-M` can also be used to allow selection of records from files that contain combined records from many machines and have a common suffix (such as `all`).

`-N`
Select objects in *new mode*.This flag is off by default, thus retaining backward compatibility. In the existing, *old mode*, specifying the `-e`, `-f`, `-g`, `-r`, or `-u` flags would select not only actions taken with those IDs, but also certain objects owned by those IDs. When running in *new mode*, only actions are selected. In order to select objects, the `-o` option must be used.

`-O` *suffix*
Direct output stream to a file in the currenti `audit_root_dir` with the indicated suffix. *suffix* may alternatively contain a full pathname, in which case the last component is taken as the suffix, ahead of which the timestamps will be placed, ahead of which the remainder of the pathname will be placed. If the `-O` option is not specified, the output is sent to the standard output. When `auditreduce` places timestamps in the filename, it uses the times of the first and last records in the merge as the *start-time* and *end-time*.

`-Q`
Quiet. Suppress notification about errors with input files.

`-R` *pathname*
Specify the pathname of an alternate audit root directory *audit_root_dir* to be *pathname*. Therefore, rather than using `/etc/security/audit/*/`files by default, *pathname*`/*/`files will be examined instead.

`-S` *server*
This option causes `auditreduce` to read audit trail files from a specific location (server directory). *server* is normally interpreted as the name of a subdirectory of the audit root, therefore `auditreduce` will look in *audit_root_dir*`/`*server*`/`files for

the audit trail files. But if *server* contains any '/' characters, it is the name of a specific directory not necessarily contained in the audit root. In this case, *server*/files will be consulted. This option allows archived files to be manipulated easily, without requiring that they be physically located in a directory structure like that of `/etc/security/audit`.

`-V`
Verbose. Display the name of each file as it is opened, and how many records total were written to the output stream.

Record Selection Options

The record selection options listed below are used to indicate which records are written to the output file produced by `auditreduce`.

Multiple arguments of the same type are not permitted.

`-a` *date-time*
Select records that occurred at or after *date-time*. The *date-time* argument is described under `Option Arguments`, below. *date-time* is in local time. The `-a` and `-b` options can be used together to form a range.

`-b` *date-time*
Select records that occurred before *date-time*.

`-c` *audit-classes*
Select records by audit class. Records with events that are mapped to the audit classes specified by *audit-classes* are selected. Audit class names are defined in `audit_class`(4). The *audit-classes* can be a comma separated list of `audit` *flags* like those described in `audit_control`(4). Using the `audit` *flags*, one can select records based upon success and failure criteria.

`-d` *date-time*
Select records that occurred on a specific day (a 24-hour period beginning at 00:00:00 of the day specified and ending at 23:59:59). The day specified is in local time. The time portion of the argument, if supplied, is ignored. Any records with timestamps during that day are selected. If any hours, minutes, or seconds are given in *time,* they are ignored. `-d` can not be used with `-a` or `-b`.

`-e` *effective-user*
Select records with the specified *effective-user.*

`-f` *effective-group*
Select records with the specified *effective-group.*

`-g` *real-group*
Select records with the specified *real-group.*

`-j` *subject-ID*
Select records with the specified *subject-ID* where *subject-ID* is a process ID.

`-m` *event*
Select records with the indicated *event*. The *event* is the literal string or the *event* number.

`-o` *object_type=objectID_value*

Select records by object type. A match occurs when the record contains the information describing the specified *object_type* and the object ID equals the value specified by *objectID_value.* The allowable object types and values are as follows:

file=*pathname*

Select records containing file system objects with the specified pathname, where pathname is a comma separated list of regular expressions. If a regular expression is preceeded by a tilde (~), files matching the expression are excluded from the output. For example, the option `file=~/usr/openwin,/usr,/etc` would select all files in `/usr` or `/etc` except those in `/usr/openwin`. The order of the regular expressions is important because auditreduce processes them from left to right, and stops when a file is known to be eitherselected or excluded. Thus the option `file= /usr, /etc, ~/usr/openwin` would select all files in `/usr` and all files in `/etc`. Files in `/usr/openwin` are not excluded because the regular expression `/usr` is matched first. Care should be given in surrounding the *pathname* with quotes so as to prevent the shell from expanding any tildes.

filegroup=*group*

Select records containing file system objects with *group* as the owning group.

fileowner=*user*

Select records containing file system objects with *user* as the owning user.

msgqid=*ID*

Select records containing message queue objects with the specified *ID* where *ID* is a message queue ID.

msgqgroup=*group*

Select records containing message queue objects with *group* as the owning or creating group.

msgqowner=*user*

Select records containing message queue objects with *user* as the owning or creating user.

pid=*ID*

Select records containing process objects with the specified *ID* where *ID* is a process ID. Process are objects when they are receivers of signals.

procgroup=*group*

Select records containing process objects with *group* as the real or effective group.

procowner=*user*

Select records containing process objects with *user* as the real or effective user.

semid=*ID*

Select records containing semaphore objects with the specified *ID* where *ID* is a semaphore ID.

semgroup=*group*

Select records containing semaphore objects with *group* as the owning or creating group.

semowner=*user*
Select records containing semaphore objects with *user* as the owning or creating user.

shmid=*ID*
Select records containing shared memory objects with the specified *ID* where *ID* is a shared memory ID.

shmgroup=*group*
Select records containing shared memory objects with *group* as the owning or creating group.

shmowner=*user*
Select records containing shared memory objects with *user* as the owning or creating user.

sock=*port_number* | *machine*
Select records containing socket objects with the specified *port_number* or the specified *machine* where *machine* is a machine name as defined in `hosts`(4).

`-r` *real-user*
Select records with the specified *real-user*.

`-u` *audit-user*
Select records with the specified *audit-user.* When one or more *filename* arguments appear on the command line, only the named files are processed. Files specified in this way need not conform to the audit trail filename format. However, `-M`, `-S`, and `-R` may not be used when processing named files. If the *filename* is "–" then the input is taken from the standard input.

Option Arguments

audit-trail-file
An audit trail file as defined in `audit.log`(4). An audit trail file not named on the command line must conform to the audit trail file name format. Audit trail files produced as output of `auditreduce` are in this format as well. The format is:

start-time . end-time . suffix

start-time is the 14 character time stamp denoting when the file was opened. *end-time* is the 14 character time stamp denoting when the file was closed. *end-time* may also be the literal string `not_terminated`, indicating the file is still be written to by the audit daemon or the file was not closed properly (a system crash or abrupt halt occurred). *suffix* is the name of the machine that generated the audit trail file (or some other meaningful suffix; e.g. `all` would be a good suffix if the audit trail file contains a combined group of records from many machines).

date-time
The *date-time* argument to `-a`, `-b`, and `-d` can be of two forms: An absolute *date-time* takes the form:

yyyymmdd [*hh* [*mm* [*ss*]]]

where *yyyy* specifies a year (with 1970 as the earliest value), *mm* is the month (01-12), `dd` is the day (01-31), *hh* is the hour (00-23), *mm* is the minute (00-59), and *ss* is the second (00-59). The default is 00 for *hh*, *mm* and *ss*.

An offset can be specified as: +*n* `d|h|m| s` where *n* is a number of units, and the tags `d`, `h`, `m`, and `s` stand for days, hours, minutes and seconds, respectively. An offset is relative to the starting time. Thus, this form can only be used with the `-b` option.

event
The literal string or ordinal event number as found in `audit_event`(4). If *event* is not found in the `audit_event` file it is considered invalid.

group
The literal string or ordinal group ID number as found in `group`(4). If *group* is not found in the `group` file it is considered invalid. *group* may be negative.

pathname
A regular expression describing a pathname.

user
The literal username or ordinal user ID number as found in `passwd`(4). If the username is not found in the `passwd` file it is considered invalid. *user* may be negative.

EXAMPLES

EXAMPLE 1 The `auditreduce` command.

`praudit`(1M) is available to display audit records in a human-readable form.

This will display the entire audit trail in a human-readable form:

```
% auditreduce | praudit
```

If all the audit trail files are being combined into one large file, then deleting the original files could be desirable to prevent the records from appearing twice:

```
% auditreduce -V -d /etc/security/audit/combined/all
```

This will print what user `milner` did on April 13, 1988. The output will be displayed in a human-readable form to the standard output:

```
% auditreduce -d 19880413 -u milner | praudit
```

The above example may produce a large volume of data if `milner` has been busy. Perhaps looking at only login and logout times would be simpler. The `-c` option will select records from a specified class:

```
% auditreduce -d 19880413 -u milner -c lo | praudit
```

To see `milner`'s login/logout activity for April 13, 14, and 15 the following is used. The results are saved to a file in the current working directory. Note that the name of the output file will have `milnerlo` as the *suffix*, with the appropriate timestamp prefixes. Note that the long form of the name is used for the `-c` option:

```
% auditreduce -a 19880413 -b +3d -u milner -c login_logout -o milnerlo
```

EXAMPLE 1 The `auditreduce` command. *(Continued)*

To follow `milner`'s movement about the file system on April 13, 14, and 15 the `chdir` record types could be viewed. Note that in order to get the same time range as the above example we needed to specify the `-b` time as the day after our range. This is because `19880416` defaults to midnight of that day, and records before that fall on `0415`, the end-day of the range.

```
% auditreduce -a 19880413 -b 19880416 -u milner -m AUE_CHDIR | praudit
```

In this example the audit records are being collected in summary form (the login/logout records only). The records are being written to a summary file in a different directory than the normal audit root to prevent the selected records from existing twice in the audit root.

```
% auditreduce -d 19880330 -c lo -o /etc/security/audit_summary/logins
```

If activity for user ID 9944 has been observed, but that user is not known to the system administrator, then the following example will search the entire audit trail for any records generated by that user. `auditreduce` will query the system as to the current validity of ID 9944, and print a warning message if it is not currently active:

```
% auditreduce -o /etc/security/audit_suspect/user9944 -u 9944
```

FILES

`/etc/security/audit/`*`server`*`/files/*`
location of audit trails, when stored

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`bsmconv`(1M), `praudit`(1M), `audit.log`(4), `audit_class`(4), `audit_control`(4), `group`(4), `hosts`(4), `passwd`(4), `attributes`(5)

DIAGNOSTICS

`auditreduce` will print out error messages if there are command line errors and then exit. If there are fatal errors during the run `auditreduce` will print an explanatory message and exit. In this case the output file may be in an inconsistent state (no trailer or partially written record) and `auditreduce` will print a warning message before exiting. Successful invocation returns `0` and unsuccessful invocation returns `1`.

Since `auditreduce` may be processing a large number of input files, it is possible that the machine-wide limit on open files will be exceeded. If this happens, `auditreduce` will print a message to that effect, give information on how many file there are, and exit.

If `auditreduce` prints a record's timestamp in a diagnostic message, that time is in local time. However, when filenames are displayed, their timestamps are in GMT.

BUGS Conjunction, disjunction, negation, and grouping of record selection options should be allowed.

NOTES The functionality described in this man page is available only if the Basic Security Module (BSM) has been enabled. See `bsmconv`(1M) for more information.

NAME

audit_startup – audit subsystem initialization script

SYNOPSIS

`/etc/security/audit_startup`

DESCRIPTION

The `audit_startup` script is used to initialize the audit subsystem before the audit deamon is started. This script is configurable by the system administrator, and currently consists of a series of `auditconfig`(1M) commands to set the system default policy, and download the initial event to class mapping.

SEE ALSO

`auditconfig`(1M), `auditd`(1M), `bsmconv`(1M), `attributes`(5)

NOTES

The functionality described in this man page is available only if the Basic Security Module (BSM) has been enabled. See `bsmconv`(1M) for more information.

NAME

auditstat – display kernel audit statistics

SYNOPSIS

auditstat [-c *count*] [-h *numlines*] [-i *interval*] [-n] [-v]

DESCRIPTION

`auditstat` displays kernel audit statistics. The fields displayed are as follows:

`aud` The total number of audit records processed by the `audit`(2) system call.

`ctl` This field is obsolete.

`drop` The total number of audit records that have been dropped. Records are dropped according to the kernel audit policy. See `auditon`(2), AUDIT_CNT policy for details.

`enq` The total number of audit records put on the kernel audit queue.

`gen` The total number of audit records that have been constructed (not the number written).

`kern` The total number of audit records produced by user processes (as a result of system calls).

`mem` The total number of Kbytes of memory currently in use by the kernel audit module.

`nona` The total number of non-attributable audit records that have been constructed. These are audit records that are not attributable to any particular user.

`rblk` The total number of times that `auditsvc`(2) has blocked waiting to process audit data.

`tot` The total number of Kbytes of audit data written to the audit trail.

`wblk` The total number of times that user processes blocked on the audit queue at the high water mark.

`wrtn` The total number of audit records written. The difference between `enq` and `wrtn` is the number of outstanding audit records on the audit queue that have not been written.

OPTIONS

`-c` *count* Display the statistics a total of *count* times. If *count* is equal to zero, statistics are displayed indefinitely. A time interval must be specified.

`-h` *numlines* Display a header for every *numlines* of statistics printed. The default is to display the header every 20 lines. If *numlines* is equal to zero, the header is never displayed.

`-i` *interval* Display the statistics every *interval* where *interval* is the number of seconds to sleep between each collection.

`-n` Display the number of kernel audit events currently configured.

`-v` Display the version number of the kernel audit module software.

EXIT STATUS

`auditstat` returns `0` upon success and `1` upon failure.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`auditconfig`(1M), `praudit`(1M), `bsmconv`(1M), `audit`(2), `auditon`(2), `auditsvc`(2), `attributes`(5)

NOTES

The functionality described in this man page is available only if the Basic Security Module (BSM) has been enabled. See `bsmconv`(1M) for more information.

NAME

audit_warn – audit daemon warning script

SYNOPSIS

/etc/security/audit_warn [*option* [*arguments*]]

DESCRIPTION

The `audit_warn` script processes warning or error messages from the audit daemon. When a problem is encountered, the audit daemon, `auditd`(1M) calls `audit_warn` with the appropriate arguments. The *option* argument specifies the error type.

The system administrator can specify a list of mail recipients to be notified when an audit_warn situation arises by defining a mail alias called `audit_warn` in `aliases`(4). The users that make up the `audit_warn` alias are typically the `audit` and `root` users.

OPTIONS

`allhard` *count*
: Indicates that the hard limit for all filesystems has been exceeded *count* times. The default action for this option is to send mail to the `audit_warn` alias only if the *count* is `1`, and to write a message to the machine console every time. It is recommended that mail *not* be sent every time as this could result in a the saturation of the file system that contains the mail spool directory.

`allsoft`
: Indicates that the soft limit for all filesystems has been exceeded. The default action for this option is to send mail to the `audit_warn` alias and to write a message to the machine console.

`auditoff`
: Indicates that someone other than the audit daemon changed the system audit state to something other than `AUC_AUDITING`. The audit daemon will have exited in this case. The default action for this option is to send mail to the `audit_warn` alias and to write a message to the machine console.

`ebusy`
: Indicates that the audit daemon is already running. The default action for this option is to send mail to the `audit_warn` alias and to write a message to the machine console.

`getacdir` *count*
: Indicates that there is a problem getting the directory list from `audit_control`(4). The audit daemon will hang in a sleep loop until the file is fixed. The default action for this option is to send mail to the `audit_warn` alias only if *count* is `1`, and to write a message to the machine console every time. It is recommended that mail *not* be sent every time as this could result in a the saturation of the file system that contains the mail spool directory.

`hard` *filename*	Indicates that the hard limit for the file has been exceeded. The default action for this option is to send mail to the `audit_warn` alias and to write a message to the machine console.
`nostart`	Indicates that auditing could not be started. The default action for this option is to send mail to the `audit_warn` alias and to write a message to the machine console. Some administrators may prefer to modify `audit_warn` to reboot the system when this error occurs.
`postsigterm`	Indicates that an error occurred during the orderly shutdown of the audit daemon. The default action for this option is to send mail to the `audit_warn` alias and to write a message to the machine console.
`soft` *filename*	Indicates that the soft limit for *filename* has been exceeded. The default action for this option is to send mail to the `audit_warn` alias and to write a message to the machine console.
`tmpfile`	Indicates that the temporary audit file already exists indicating a fatal error. The default action for this option is to send mail to the `audit_warn` alias and to write a message to the machine console.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsr

SEE ALSO

`audit`(1M), `auditd`(1M), `bsmconv`(1M), `aliases`(4), `audit.log`(4), `audit_control`(4), `attributes`(5)

NOTES

The functionality described in this man page is available only if the Basic Security Module (BSM) has been enabled. See `bsmconv`(1M) for more information.

NAME

automount – install automatic mount points

SYNOPSIS

/usr/sbin/automount [-t *duration*] [-v]

DESCRIPTION

The automount utility installs autofs mount points and associates an automount map with each mount point. The autofs file system monitors attempts to access directories within it and notifies the automountd(1M) daemon. The daemon uses the map to locate a file system, which it then mounts at the point of reference within the autofs file system. A map can be assigned to an autofs mount using an entry in the /etc/auto_master map or a direct map.

If the file system is not accessed within an appropriate interval (10 minutes by default), the automountd daemon unmounts the file system.

The file /etc/auto_master determines the locations of all autofs mount points. By default, this file contains four entries:

```
# Master map for automounter
#
+auto_master
/net          -hosts    -nosuid
/home         auto_home
/xfn          -xfn
```

The +auto_master entry is a reference to an external NIS or NIS+ master map. If one exists, then its entries are read as if they occurred in place of the +auto_master entry. The remaining entries in the master file specify a directory on which an autofs mount will be made followed by the automounter map to be associated with it. Optional mount options may be supplied as an optional third field in the each entry. These options are used for any entries in the map that do not specify mount options explicitly. The automount command is usually run without arguments. It compares the entries /etc/auto_master with the current list of autofs mounts in /etc/mnttab and adds, removes or updates autofs mounts to bring the /etc/mnttab up to date with the /etc/auto_master. At boot time it installs all autofs mounts from the master map. Subsequently, it may be run to install autofs mounts for new entries in the master map or the direct map, or to perform unmounts for entries that have been removed from these maps.

OPTIONS

The following options are supported:

-t *duration* — Specifies a *duration*, in seconds, that a file system is to remain mounted when not in use. The default is 10 minutes.

-v — Verbose mode. Notifies of autofs mounts, unmounts, or other non-essential information.

Map Entry Format

A simple map entry (mapping) takes the form:

```
key [ -mount-options ] location . . .
```

where `key` is the full pathname of the directory to mount when used in a direct map, or the simple name of a subdirectory in an indirect map. *mount-options* is a comma-separated list of `mount` options, and *location* specifies a file system from which the directory may be mounted. In the case of a simple NFS mount, the options that can be used are as specified in `mount_nfs`(1M), and *location* takes the form:

host `:` *pathname*

host is the name of the host from which to mount the file system, and *pathname* is the absolute pathname of the directory to mount.

Options to other file systems are documented on the other `mount_*` reference manual pages, for example, `mount_cachefs`(1M).

Replicated File Systems

Multiple *location* fields can be specified for replicated NFS file systems, in which case `automount` and the kernel will each try to use that information to increase availability. If the read-only flag is set in the map entry, `automount` mounts a list of locations that the kernel may use, sorted by several criteria. When a server does not respond, the kernel will switch to an alternate server. The sort ordering of `automount` is used to determine how the next server is chosen. If the read-only flag is not set, `automount` will mount the best single location, chosen by the same sort ordering, and new servers will only be chosen when an unmount has been possible, and a remount is done. Servers on the same local subnet are given the strongest preference, and servers on the local net are given the second strongest preference. Among servers equally far away, response times will determine the order if no weighting factors (see below) are used.

If the list includes server locations using both the NFS Version 2 Protocol and the NFS Version 3 Protocol, `automount` will choose only a subset of the server locations on the list, so that all entries will be the same protocol. It will choose servers with the NFS Version 3 Protocol so long as an NFS Version 2 Protocol server on a local subnet will not be ignored. See the *System Administration Guide: IP Services* for additional details.

If each *location* in the list shares the same *pathname* then a single *location* may be used with a comma-separated list of hostnames:

`hostname,hostname . . . :` *pathname*

Requests for a server may be weighted, with the weighting factor appended to the server name as an integer in parentheses. Servers without a weighting are assumed to have a value of zero (most likely to be selected). Progressively higher values decrease the chance of being selected. In the example,

```
man -ro alpha,bravo,charlie(1),delta(4) : /usr/man
```

hosts `alpha` and `bravo` have the highest priority; host `delta` has the lowest.

Server proximity takes priority in the selection process. In the example above, if the server `delta` is on the same network segment as the client, but the others are on different network segments, then `delta` will be selected; the weighting value is ignored. The weighting has effect only when selecting between servers with the same network proximity.

In cases where each server has a different export point, the weighting can still be applied. For example:

```
man -ro alpha:/usr/man  bravo,charlie(1):/usr/share/man
     delta(3):/export/man
```

A mapping can be continued across input lines by escaping the NEWLINE with a backslash (\) Comments begin with a number sign (#) and end at the subsequent NEWLINE.

Map Key Substitution

The ampersand (`&`) character is expanded to the value of the `key` field for the entry in which it occurs. In this case:

```
jane sparcserver : /home/&
```

the `&` expands to `jane`.

Wildcard Key

The asterisk (`*`) character, when supplied as the `key` field, is recognized as the catch-all entry. Such an entry will match any key not previously matched. For instance, if the following entry appeared in the indirect map for `/config`:

```
*         & : /export/config/&
```

this would allow automatic mounts in `/config` of any remote file system whose location could be specified as:

```
hostname : /export/config/hostname
```

Variable Substitution

Client specific variables can be used within an `automount` map. For instance, if `$HOST` appeared within a map, `automount` would expand it to its current value for the client's host name. Supported variables are:

`ARCH`	The application architecture is derived from the output of `uname -m`	The architecture name. For example, "sun4" on a sun4u machine.
`CPU`	The output of `uname -p`	The processor type. For example, "sparc"
`HOST`	The output of `uname -n`	The host name. For example, "biggles"

OSNAME	The output of `uname -s`	The OS name. For example, "SunOS"
OSREL	The output of `uname -r`	The OS release name. For example "5.3"
OSVERS	The output of `uname -v`	The OS version. For example, "beta1.0"
NATISA	The output of `isainfo -n`	The native instruction set architecture for the system. For example, "sparcv9"

If a reference needs to be protected from affixed characters, you can surround the variable name with curly braces ({ }).

Multiple Mounts

A multiple mount entry takes the form:

`key [-mount-options] [ [mountpoint] [-mount-options] location . . . ] . . .`

The initial /[*mountpoint*] is optional for the first mount and mandatory for all subsequent mounts. The optional *mountpoint* is taken as a pathname relative to the directory named by `key`. If *mountpoint* is omitted in the first occurrence, a *mountpoint* of / (root) is implied.

Given an entry in the indirect map for `/src`

```
beta     -ro\
  /           svr1,svr2:/export/src/beta  \
  /1.0        svr1,svr2:/export/src/beta/1.0  \
  /1.0/man    svr1,svr2:/export/src/beta/1.0/man
```

All offsets must exist on the server under `beta`. `automount` will automatically mount `/src/beta`, `/src/beta/1.0`, and `/src/beta/1.0/man`, as needed, from either `svr1` or `svr2`, whichever host is nearest and responds first.

Other File System Types

The automounter assumes NFS mounts as a default file system type. Other file system types can be described using the `fstype` mount option. Other mount options specific to this file system type can be combined with the `fstype` option. The location field must contain information specific to the file system type. If the location field begins with a slash, a colon character must be prepended, for instance, to mount a CD file system:

```
cdrom -fstype=hsfs,ro   : /dev/sr0
```

or to perform an `autofs` mount:

```
src   -fstype=autofs    auto_src
```

Note: Use this procedure only if you are not using Volume Manager.

Mounts using CacheFS are most useful when applied to an entire map as map defaults. The following entry in the master map describes cached home directory mounts. It assumes the default location of the cache directory, `/cache`.

```
/home auto_home       -fstype=cachefs,backfstype=nfs
```

See the `NOTES` section for information on option inheritance.

Indirect Maps

An indirect map allows you to specify mappings for the subdirectories you wish to mount under the `directory` indicated on the command line. In an indirect map, each `key` consists of a simple name that refers to one or more file systems that are to be mounted as needed.

Direct Maps

Entries in a direct map are associated directly with `autofs` mount points. Each `key` is the full pathname of an `autofs` mount point. The direct map as a whole is not associated with any single directory.

Included Maps

The contents of another map can be included within a map with an entry of the form

+mapname

If *mapname* begins with a slash, it is assumed to be the pathname of a local file. Otherwise, the location of the map is determined by the policy of the name service switch according to the entry for the automounter in `/etc/nsswitch.conf`, such as

```
automount: files nis
```

If the name service is `files`, then the name is assumed to be that of a local file in `/etc`. If the key being searched for is not found in the included map, the search continues with the next entry.

Special Maps

There are three special maps available: `-hosts`, `-xfn`, and `-null`. The `-hosts` map is used with the `/net` directory and assumes that the map key is the hostname of an NFS server. The `automountd` daemon dynamically constructs a map entry from the server's list of exported file systems. References to a directory under `/net/hermes` will refer to the corresponding directory relative to `hermes` root.

The `-xfn` map is used to mount the initial context of the Federated Naming Service (FNS) namespace under the `/xfn` directory. For more information on FNS, see `fns`(5), `fns_initial_context`(5), `fns_policies`(5), and the Federated Naming Service Guide.

The `-null` map cancels a previous map for the directory indicated. This is most useful in the `/etc/auto_master` for cancelling entries that would otherwise be inherited from the `+auto_master` include entry. To be effective, the `-null` entries must be inserted before the included map entry.

Executable Maps

Local maps that have the execute bit set in their file permissions will be executed by the automounter and provided with a key to be looked up as an argument. The executable map is expected to return the content of an automounter map entry on its stdout or no output if the entry cannot be determined. A direct map cannot be made executable.

Configuration and the auto_master Map

When initiated without arguments, `automount` consults the master map for a list of `autofs` mount points and their maps. It mounts any `autofs` mounts that are not already mounted, and unmounts `autofs` mounts that have been removed from the master map or direct map.

The master map is assumed to be called `auto_master` and its location is determined by the name service switch policy. Normally the master map is located initially as a local file `/etc/auto_master`.

Browsing

The Solaris 2.6 release supports browsability of indirect maps. This allows all of the potential mount points to be visible, whether or not they are mounted. The `-nobrowse` option can be added to any indirect `autofs` map to disable browsing. For example:

```
/net     -hosts      -nosuid,nobrowse
/home    auto_home
```

In this case, any *hostnames* would only be visible in `/net` after they are mounted, but all potential mount points would be visible under `/home`. The `-browse` option enables browsability of `autofs` file systems. This is the default for all indirect maps.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` An error occurred.

FILES

`/etc/auto_master`	master automount map.
`/etc/auto_home`	map to support automounted home directories.
`/etc/nsswitch.conf`	the name service switch configuration file.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`isainfo`(1), `ls`(1), `uname`(1), `automountd`(1M), `mount`(1M), `mount_cachefs`(1M), `mount_nfs`(1M), `attributes`(5), `fns`(5), `fns_initial_context`(5), `fns_policies`(5), `nfssec`(5)

System Administration Guide: IP Services

NOTES

`autofs` mount points must not be hierarchically related. `automount` does not allow an `autofs` mount point to be created within another `autofs` mount.

Since each direct map entry results in a new `autofs` mount such maps should be kept short.

Entries in both direct and indirect maps can be modified at any time. The new information is used when `automountd` next uses the map entry to do a mount.

New entries added to a master map or direct map will not be useful until the automount command is run to install them as new `autofs` mount points. New entries added to an indirect map may be used immediately.

As of the Solaris 2.6 release, a listing (see `ls`(1)) of the `autofs` directory associated with an indirect map shows all potential mountable entries. The attributes associated with the potential mountable entries are temporary. The real file system attributes will only be shown once the file system has been mounted.

Default mount options can be assigned to an entire map when specified as an optional third field in the master map. These options apply only to map entries that have no mount options. Note that map entities with options override the default options, as at this time, the options do not concatenate. The concatenation feature is planned for a future release.

When operating on a map that invokes an NFS mount, the default number of retries for the automounter is 0, that is, a single mount attempt, with no retries. Note that this is significantly different from the default (10000) for the `mount_nfs`(1M) utility.

The Network Information Service (NIS) was formerly known as Sun Yellow Pages (YP). The functionality of the two remains the same.

NAME

automountd – autofs mount/unmount daemon

SYNOPSIS

automountd [-Tvn] [-D *name=value*]

DESCRIPTION

automountd is an RPC server that answers file system mount and unmount requests from the autofs file system. It uses local files or name service maps to locate file systems to be mounted. These maps are described with the automount(1M) command.

The automountd daemon is automatically invoked in run level 2.

OPTIONS

-T	Trace. Expand each RPC call and display it on the standard output.
-v	Verbose. Log status messages to the console.
-n	Turn off browsing for all autofs mount points. This option overrides the -browse autofs map option on the local host.
-D *name=value*	Assign *value* to the indicated automount map substitution variable. These assignments cannot be used to substitute variables in the master map auto_master.

USAGE

See largefile(5) for the description of the behavior of automountd when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

FILES

/etc/auto_master	master map for automounter

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

automount(1M), attributes(5), largefile(5)

NAME

autopush – configures lists of automatically pushed STREAMS modules

SYNOPSIS

autopush -f *filename*

autopush -g -M *major* -m *minor*

autopush -r -M *major* -m *minor*

DESCRIPTION

The `autopush` command configures the list of modules to be automatically pushed onto the stream when a device is opened. It can also be used to remove a previous setting or get information on a setting.

OPTIONS

The following options are supported:

`-f` *filename*

Sets up the `autopush` configuration for each driver according to the information stored in *filename*. An `autopush` file consists of lines of four or more fields, separated by spaces as shown below:

major minor last-minor module1 module2 . . . module8

The first field is a string that specifies the *major* device name, as listed in the `/kernel/drv` directory. The next two fields are integers that specify the *minor* device number and *last-minor* device number. The fields following represent the names of modules. If *minor* is `–1`, then all minor devices of a major driver specified by *major* are configured, and the value for *last-minor* is ignored. If *last-minor* is `0`, then only a single minor device is configured. To configure a range of minor devices for a particular major, *minor* must be less than *last-minor*.

The remaining fields list the names of modules to be automatically pushed onto the stream when opened, along with the position of an optional anchor. The maximum number of modules that can be pushed is eight. The modules are pushed in the order they are specified. The optional special character sequence `[anchor]` indicates that a STREAMS anchor should be placed on the stream at the module previously specified in the list; it is an error to specify more than one anchor or to have an anchor first in the list.

A nonzero exit status indicates that one or more of the lines in the specified file failed to complete successfully.

`-g`

Gets the current configuration setting of a particular *major* and *minor* device number specified with the `-M` and `-m` options respectively and displays the autopush modules associated with it. It will also return the starting minor device number if the request corresponds to a setting of a range (as described with the `-f` option).

`-m` *minor*

Specifies the minor device number.

`-M` *major*	Specifies the major device number.
`-r`	Removes the previous configuration setting of the particular *major* and *minor* device number specified with the `-M` and `-m` options respectively. If the values of *major* and *minor* correspond to a previously established setting of a range of minor devices, where *minor* matches the first minor device number in the range, the configuration would be removed for the entire range.

EXIT STATUS

The following exit values are returned:

0	Successful completion.
non-zero	An error occurred.

EXAMPLES

EXAMPLE 1 Using the `autopush` command.

The following example gets the current configuration settings for the *major* and *minor* device numbers as indicated and displays the `autopush` modules associated with them for the character-special device `/dev/term/a:`

```
example# autopush -g -M 29 -m 0
Major     Minor     Lastminor     Modules
  29        0         1           ldterm  ttcompat
```

FILES

`/etc/iu.ap`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`bdconfig`(1M), `ttymon`(1M), `attributes`(5), `ldterm`(7M), `sad`(7D), `streamio`(7I), `ttcompat`(7M)

STREAMS Programming Guide

NAME

bdconfig – configures the bd (buttons and dials) stream

SYNOPSIS

bdconfig [startup] [off] [on] [*term*] [status] [verbose]

DESCRIPTION

The bdconfig utility is responsible for configuring the autopush facility and defining to the system what serial device to use for the bd stream.

OPTIONS

If no options are given, then an interactive mode is assumed. In this mode the current status is presented along with this usage line, and a series of interactive questions asked to determine the user's desires.

Root privilege is required to change the configuration. The status option does not require root privilege. bdconfig can be installed as a setuid root program.

The non-interactive options below can be given in any order.

term Specify to the system the serial device for bd use. This option implies the on option unless the off option is present.

iff Reconfigure the configured term for tty use.

on Reconfigure the configured term for bd use. If *term* has not been previously specified, interactive questions are asked to determine the user's desires.

startup Configure as was last configured before the system went down. This option is used by the startup script, and precludes the use of the on, off, and *term* options. This option implies non-interactive mode.

status Emit the current configuration in terms of the words used as options: off, on, /dev/term/a, /dev/term/b, and so forth. This option implies non interactive mode.

verbose bdconfig describes what it finds and what it is doing.

EXIT STATUS

The bdconfig utility returns 0 on success, 1 on general error, and 2 on argument error.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWdialh

SEE ALSO

autopush(1M), attributes(5), x_buttontest(6), x_dialtest(6), bd(7M), sad(7D), streamio(7I)

NOTES

All bdconfig does is configure the AUTOPUSH facility. bdconfig does not actually manipulate the serial port or stream in any way. Only the first open of a dismantled stream will see the effects of a previously run bdconfig.

The `bdconfig` utility is silent except for error messages unless:

a) invoked with no args: status / usage line emitted

b) interactive modes are invoked as described above

c) the verbose option is used

BUGS

The interface does not support more than one dialbox and one buttonbox, both of which must be on the same serial device.

There should be a library routine to read, parse, and validate records in the `iu.ap` file, so that `bdconfig` could return to the appropriate record in `iu.ap` as the default configuration.

NAME

boot – start the system kernel or a standalone program

SPARC

boot [*OBP names*] [*file*] [-afV] [-D *default-file*] [*boot-flags*] [––] [*client-program-args*]

b [*device* [*(c, u, p)*]] [-afV] [-D *default-file*] [*boot-flags*] [––] [*client-program-args*]

IA

b [*file*] [-f] [*boot-args*]

i

DESCRIPTION

Bootstrapping is the process of loading and executing a standalone program. For the purpose of this discussion, bootstrapping means the process of loading and executing the bootable operating system. Typically, the standalone program is the operating system kernel (see kernel(1M)), but any standalone program can be booted instead. On a SPARC-based system, the diagnostic monitor for a machine is a good example of a standalone program other than the operating system that can be booted.

If the standalone is identified as a dynamically-linked executable, boot will load the interpreter (linker/loader) as indicated by the executable format and then transfer control to the interpreter. If the standalone is statically-linked, it will jump directly to the standalone.

Once the kernel is loaded, it starts the UNIX system, mounts the necessary filesystems (see vfstab(4)), and runs /sbin/init to bring the system to the "initdefault" state specified in /etc/inittab. See inittab(4).

SPARC Bootstrap Procedure

On SPARC based systems, the bootstrap procedure on most machines consists of the following basic phases.

After the machine is turned on, the system firmware (in PROM) executes power-on self-test (POST). The form and scope of these tests depends on the version of the firmware in your system.

After the tests have been completed successfully, the firmware attempts to autoboot if the appropriate flag has been set in the non-volatile storage area used by the firmware. The name of the file to load, and the device to load it from can also be manipulated.

These flags and names can be set using the eeprom(1M) command from the shell, or by using PROM commands from the ok prompt after the system has been halted.

The second level program is either ufsboot (when booting from a disk), or inetboot (when booting across the network).

Network Booting

Network booting can follow either of two paths, RARP/bootparams or DHCP (Dynamic Host Configuration Protocol), depending on the functions available in and configuration of the PROM. Machines of the sun4u kernel architecture have

DHCP-capable PROMs and boot from the network using RARP/`bootparams` by default. Whichever network boot path is specified, RARP or DHCP, is followed all the way through to multi-user mode; there is no mixture of the RARP and DHCP activities.

The boot command syntax for specifying the two methods of network booting are:

```
boot net:rarp
boot net:dhcp
```

The command:

```
boot net
```

without a `rarp` or `dhcp` specifier, invokes the default method for network booting over the network interface for which `net` is an alias.

The sequence of events for network booting using RARP/bootparams is described in the following paragraphs. The sequence for DHCP follows the RARP/bootparams description.

When booting over the network using RARP/bootparams, the PROM makes a reverse ARP request and when it receives a reply, the PROM broadcasts a TFTP request to fetch `inetboot` over the network from any server that responds and executes it. `inetboot` also makes another reverse ARP request, then uses the `bootparams` protocol (see `bootparams`(4)) to locate its root filesystem. It then fetches the kernel across the network using the NFS protocol and then executes it.

When booting over the network using DHCP, the PROM broadcasts the hardware address and kernel architecture and requests an IP address, boot parameters, and network configuration information. After a DHCP server responds and is selected (from among potentially multiple servers), that server sends to the client an IP address and all other information needed to boot the client. After receipt of this information, the client PROM downloads `inetboot`, loads that file into memory, and executes it. `inetboot` invokes the kernel, which loads the files it needs and releases inetboot. Startup scripts then initiate the DHCP agent (see `dhcpagent`(1M)), which implements the further activities of the DHCP.

Booting from Disk

When booting from disk (or disk-like device), the bootstrapping process consists of two conceptually distinct phases, primary boot and secondary boot. In the primary boot phase, the PROM loads the primary boot block from blocks 1 to 15 of the disk partition selected as the boot device.

If the pathname to the standalone is relative (does not begin with a slash), the second level boot will look for the standalone in a platform-dependent search path. This path is guaranteed to contain `/platform/`*platform-name*. Many SPARC platforms next

search the platform-specific path entry `/platform/`*hardware-class-name*. See `filesystem`(5). If the pathname is absolute, `boot` will use the specified path. The `boot` program then loads the standalone at the appropriate address, and then transfers control.

If the filename is not given on the command line or otherwise specified, for example, by the `boot-file` NVRAM variable, `boot` chooses an appropriate default file to load based on what software is installed on the system, the capabilities of the hardware and firmware, and on a user configurable policy file (see FILES, below).

OpenBoot PROM `boot` Command Behavior

The OpenBoot `boot` command takes arguments of the following form:

`ok boot` [*device-specifier*] [*arguments*]
The default `boot` command has no arguments:

```
ok boot
```

If no *device-specifier* is given on the `boot` command line, OpenBoot typically uses the *boot-device* or *diag-device* `nvram` variable. If no optional *arguments* are given on the command line, OpenBoot typically uses the *boot-file* or *diag-file* `nvram` variable as default `boot` arguments. (If the system is in diagnostics mode, *diag-device* and *diag-file* are used instead of *boot-device* and *boot-file*).

arguments may include more than one string. All *argument* strings are passed to the secondary booter; they are not interpreted by OpenBoot.

If any *arguments* are specified on the `boot` command line, then neither the *boot-file* nor the *diag-file* `nvram` variable is used. The contents of the `nvram` variables are not merged with command line arguments. For example, the command

`ok boot -s`ignores the settings in both *boot-file* and *diag-file*; it interprets the string `"-s"` as *arguments*. `boot` will not use the contents of *boot-file* or *diag-file*.

The commands

`ok boot net`and

`ok boot cdrom`have no arguments; they will use the settings in *boot-file* or *diag-file*, if they are set, as default filename and arguments and pass them to `boot`. Accordingly, if *boot-file* is set to the 64-bit kernel filename and you attempt to boot the installation CD with `boot cdrom`, `boot` will fail if the installation CD contains only a 32-bit kernel.

Since the contents of *boot-file* or *diag-file* may be ignored depending on the form of the `boot` command used, reliance upon the *boot-file* should be discouraged for most production systems. To change the OS policy, change the policy file. A significant exception is when a production system has both 32-bit and 64-bit packages installed, but the production system requires use of the 32-bit OS.

In most cases, it is best to allow the `boot` command to choose an appropriate default based upon the system type, system hardware and firmware, and upon what is installed on the root filesystem. It is accepted practice to augment the `boot` command's policy by modifying the policy file; however, changing *boot-file* or *diag-file* may generate unexpected results in certain circumstances.

This behavior is found on most OpenBoot 2.x and 3.x based systems. Note that differences may occur on some platforms.

IA Bootstrap Procedure

On IA based systems, the bootstrapping process consists of two conceptually distinct phases, primary boot and secondary boot. The primary boot is implemented in the BIOS ROM on the system board, and BIOS extensions in ROMs on peripheral boards. It is distinguished by its ability to control the installed peripheral devices and to provide I/O services through software interrupts. It begins the booting process by loading the first physical sector from a floppy disk, hard disk, or CD-ROM, or, if supported by the system or network adapter BIOS, by reading a bootstrap program from a network boot server. The primary boot is implemented in IA real-mode code.

The secondary boot is loaded by the primary boot. It is implemented in 32-bit, paged, protected mode code. It also loads and uses peripheral-specific BIOS extensions written in IA real-mode code. The secondary boot is called `boot.bin` and is capable of reading and booting from a UFS file system on a hard disk or a CD or by way of a LAN using the NFS protocol.

The secondary boot is responsible for running the `Configuration Assistant` program which determines the installed devices in the system (possibly with help from the user). The secondary boot then reads the script in `/etc/bootrc`, which controls the booting process. This file contains `boot interpreter` commands, which are defined below, and can be modified to change defaults or to adapt to a specific machine.

The standard `/etc/bootrc` script prompts the user to enter a `b` character to boot with specified options, an `i` character to invoke the interpreter interactively, or any other character to boot the default kernel. Once the kernel is loaded, it starts the operating system, loads the necessary modules, mounts the necessary filesystems (see `vfstab`(4)), and runs `/sbin/init` to bring the system to the ''initdefault'' state specified in `/etc/inittab`. See `inittab`(4).

SPARC

OBP names — Specify the open boot prom designations. For example, on Desktop SPARC based systems, the designation `/sbus/esp@0,800000/sd@3,0:a` indicates a SCSI disk (sd) at target 3, lun0 on the SCSI bus, with the esp host adapter plugged into slot 0.

file — Name of a standalone program to `boot`. If a filename is not explicitly specified, either on the `boot` command line or in the *boot-file* NVRAM variable, `boot` chooses an appropriate default filename. On most systems, the default filename is the 32-bit kernel. On systems

capable of supporting both the 32-bit and 64-bit kernels, the 64-bit kernel will be chosen in preference to the 32-bit kernel. `boot` chooses an appropriate default *file* to `boot` based on what software is installed on the system, the capabilities of the hardware and firmware, and on a user configurable policy file.

`-a` — The boot program interprets this flag to mean `ask me`, and so it prompts for the name of the standalone. The `'-a'` flag is then passed to the standalone program.

`-f` — When booting an Autoclient system, this flag forces the boot program to bypass the client's local cache and read all files over the network from the client's file server. This flag is ignored for all non-Autoclient systems. The `-f` flag is then passed to the standalone program.

`-V` — Display verbose debugging information.

`-D` *default-file* — Explicitly specify the *default-file*. On some systems, `boot` chooses a dynamic default file, used when none is otherwise specified. This option allows the *default-file* to be explicitly set and can be useful when booting `kadb`(1M) since, by default, `kadb` loads the default-file as exported by the `boot` program.

boot-flags — The boot program passes all *boot-flags* to `file`. They are not interpreted by `boot`. See the `kernel`(1M) and `kadb`(1M) manual pages for information about the options available with the default standalone program.

client-program-args — The `boot` program passes all *client-program-args* to *file*. They are not interpreted by `boot`.

IA

`file` — Name of a standalone program to boot. The default is to boot `/platform/`*platform-name*`/kernel/unix` from the root partition, but you can specify another program on the command line.

`-f` — When booting an Autoclient system, this flag forces the boot program to bypass the client's local cache and read all files over the network from the client's file server. This flag is ignored for all non-Autoclient systems. The `-f` flag is then passed to the standalone program.

boot-args — The boot program passes all *boot-args* to `file`. They are not interpreted by `boot`. See `kernel`(1M) and `kadb`(1M) for information about the options available with the kernel.

IA BOOT SEQUENCE DETAILS

After a PC-compatible machine is turned on, the system firmware in the BIOS ROM executes a power-on self test (POST), runs BIOS extensions in peripheral board ROMs, and invokes software interrupt INT 19h, Bootstrap. The INT 19h handler typically performs the standard PC-compatible boot, which consists of trying to read the first physical sector from the first diskette drive, or, if that fails, from the first hard disk. The processor then jumps to the first byte of the sector image in memory.

IA Primary Boot

The first sector on a floppy disk contains the master boot block. The boot block is responsible for loading the image of the boot loader `strap.com`, which then loads the secondary boot, `boot.bin`. A similar sequence occurs for CD-ROM boot, but the master boot block location and contents are dictated by the El Torito specification. The El Torito boot also leads to `strap.com`, which in turn loads `boot.bin`.

The first sector on a hard disk contains the master boot block, which contains the master boot program and the FDISK table, named for the PC program that maintains it. The master boot finds the active partition in the FDISK table, loads its first sector, and jumps to its first byte in memory. This completes the standard PC-compatible hard disk boot sequence.

An IA FDISK partition for the Solaris software begins with a one-cylinder boot slice, which contains the partition boot program (`pboot`) in the first sector, the standard Solaris disk label and volume table of contents (VTOC) in the second and third sectors, and the `bootblk` program in the fourth and subsequent sectors. When the FDISK partition for the Solaris software is the active partition, the master boot program (`mboot`) reads the partition boot program in the first sector into memory and jumps to it. It in turn reads the `bootblk` program into memory and jumps to it. Regardless of the type of the active partition, if the drive contains multiple FDISK partitions, the user is given the opportunity to reboot another partition.

`bootblk` or `strap.com` (depending upon the active partition type) reads `boot.bin` from the file system in the Solaris root slice and jumps to its first byte in memory.

For network booting, you have the choice of the boot floppy or Intel's Preboot eXecution Environment (PXE) standard. When booting from the network using the boot floppy, you can select which network configuration strategy you want by editing the boot properties, changing the setting for `net-config-strategy`. By default, `net-config-strategy` is set to `rarp`. It can have two settings, `rarp` or `dhcp`. When booting from the network using PXE, the system or network adapter BIOS uses DHCP to locate a network bootstrap program (NBP) on a boot server and reads it using Trivial File Transfer Protocol (TFTP). The BIOS executes the NBP by jumping to its first byte in memory. The NBP uses DHCP to locate the secondary bootstrap on a boot server, reads it using TFTP, and executes it.

IA Secondary Boot

The secondary boot, `boot.bin`, switches the processor to 32-bit, paged, protected mode, and performs some limited machine initialization. It runs the `Configuration Assistant` program which either auto-boots the system, or presents a list of possible boot devices, depending on the state of the `auto-boot?` variable (see `eeprom`(1M)).

Disk target devices (including CDROM drives) are expected to contain UFS filesystems. Network devices can be configured to use either DHCP or Reverse Address Resolution Protocol (RARP) and `bootparams` RPC to discover the machine's IP address and which server will provide the root file system. The root file system is then mounted using NFS. After a successful root mount, `boot.bin` invokes a command interpreter, which interprets `/etc/bootrc`.

Secondary Boot Programming Language for IA

The wide range of hardware that must be supported on IA based systems demands great flexibility in the booting process. This flexibility is achieved in part by making the secondary boot programmable. The secondary boot contains an interpreter that accepts a simple command language similar to those of `sh` and `csh`. The primary differences are that pipelines, loops, standard output, and output redirection are not supported.

IA Lexical Structure

The boot interpreter splits input lines into words separated by blanks and tabs. The metacharacters are dollar sign (`$`), single-quote (`'`), double-quote (`"`), number sign (`#`), new-line, and backslash (`\`). The special meaning of metacharacters can be avoided by preceding them with a backslash. A new-line preceded by a backslash is treated as a blank. A number sign introduces a comment, which continues to the next new-line.

A string enclosed in a pair of single-quote or double-quote characters forms all or part of a single word. White space and new-line characters within a quoted string become part of the word. Characters within a quoted string can be quoted by preceding them with a backslash character; thus a single-quote character can appear in a single-quoted string by preceding it with a backslash. Two backslashes produce a single backslash, and a new-line preceded by a backslash produces a new-line in the string.

IA Variables

The boot maintains a set of variables, each of which has a string value. The first character of a variable name must be a letter, and subsequent characters can be letters, digits, or underscores. The `set` command creates a variable and/or assigns a value to it, or displays the values of variables. The `unset` command deletes a variable.

Variable substitution is performed when the interpreter encounters a dollar-sign that is not preceded by a backslash. The variable name following the dollar sign is replaced by the value of the variable, and parsing continues at the beginning of the value. Variable substitution is performed in double-quoted strings, but not in single-quoted strings. A variable name can be enclosed in braces to separate it from following characters.

IA Commands

A command is a sequence of words terminated by a new-line character. The first word is the name of the command and subsequent words are arguments to the command. All commands are built-in commands. Standalone programs are executed with the `run` command.

IA Conditional Execution of Commands

Commands can be conditionally executed by surrounding them with the `if`, `elseif`, `else`, and `endif` commands:

```
if expr1
    . . .
```

```
elseif expr2
     . . .
elseif expr3
     . . .
else
     . . .
endif
```

An `if` block may be embedded in other `if` blocks.

IA Expressions

The `set`, `if`, and `elseif` commands evaluate arithmetic expressions with the syntax and semantics of the C programming language. The `||`, `&&`, `|`, `^`, `&`, `==`, `!=`, `<`, `>`, `<=`, `>=`, `>>`, `<<`, `+`, `–`, `*`, `/`, `%`, `~`, and `!` operators are accepted, as are `(`, `)`, and comma. Signed 32-bit integer arithmetic is performed.

Expressions are parsed after the full command line has been formed. Each token in an expression must be a separate argument word, so blanks must separate all tokens on the command line.

Before an arithmetic operation is performed on an operand word, it is converted from a string to a signed 32-bit integer value. After an optional leading sign, a leading 0 produces octal conversion and a leading 0x or 0X produces hexadecimal conversion. Otherwise, decimal conversion is performed. A string that is not a legal integer is converted to zero.

Several built-in functions for string manipulation are provided. Built-in function names begin with a dot. String arguments to these functions are not converted to integers. To cause an operator, for example, `-`, to be treated as a string, it must be preceded by a backslash, and that backslash must be quoted with another backslash. Also be aware that a null string can produce a blank argument, and thus an expression syntax error. For example:

```
if .strneq ( ${usrarg}X , \– , 1 )
```

is the safe way to test whether the variable *usrarg* starts with a –, even if it could be null.

IA I/O

The boot interpreter takes its input from the system console or from one or more files. The source command causes the interpreter to read a file into memory and begin parsing it. The console command causes the interpreter to take its input from the system console. Reaching EOF causes the interpreter to resume parsing the previous input source. CTRL-D entered at the beginning of console line is treated as EOF.

The `echo` command writes its arguments to the display. The `read` command reads the system console and assigns word values to its argument variables.

IA Debugging

The `verbose` command turns verbose mode on and off. In verbose mode, the interpreter displays lines from the current source file and displays the command as actually executed after variable substitution.

The `singlestep` command turns singlestep mode on and off. In singlestep mode, the interpreter displays `step ?` before processing the next command, and waits for keyboard input, which is discarded. Processing proceeds when ENTER is pressed. This allows slow execution in verbose mode.

IA Initialization

When the interpreter is first invoked by the `boot`, it begins execution of a compiled-in initialization string. This string typically consists of "`source /etc/bootrc\n`" to run the boot script in the root file system.

IA Communication With Standalone Programs

The boot passes information to standalone programs through arguments to the `run` command. A standalone program can pass information back to the boot by setting a boot interpreter variable using the `var_ops()` boot service function. It can also pass information to the kernel using the `setprop()` boot service function. The `whoami` property is set to the name of the standalone program.

IA Built-in Commands

`console`
Interpret input from the console until CTRL-D.

`echo` *arg1 . . .*
Display the arguments separated by blanks and terminate with a new-line.

`echo -n` *arg1 . . .*
Display the arguments separated by blanks, but do not terminate with a new-line.

`getprop` *propname varname*
Assign the value of property *propname* to the variable *varname*. A property value of length zero produces a null string. If the property does not exist, the variable is not set.

`getproplen` *propname varname*
Assign the length in hexadecimal of the value of property *propname* to the variable *varname*. Property value lengths include the terminating null. If the property does not exist, the variable is set to 0xFFFFFFFF (-1).

`if` *expr*
If the expression `expr` is true, execute instructions to the next `elseif`, `else`, or `endif`. If `expr` is false, do not execute the instructions.

`elseif` *expr*
If the preceding `if` and `elseif` commands all failed, and `expr` is true, execute instructions to the next `elseif`, `else`, or `endif`. Otherwise, do not execute the instructions.

`else`
If the preceding `if` and `elseif` commands all failed, execute instructions to the next `elseif`, `else`, or `endif`. Otherwise, do not execute the instructions.

`endif`
Revert to the execution mode of the surrounding block.

`help`
Display a help screen that contains summaries of all available boot shell commands.

`read` *name1 . . .*
Read a line from the console, break it into words, and assign them as values to the variables *name1*, and so forth.

`readt` *time . . .*
Same as `read`, but timeout after `time` seconds.

`run` *name arg1 . . .*
Load and transfer control to the standalone program *name*, passing it *arg1* and further arguments.

`set`
Display all the current variables and their values.

`set` *name*
Set the value of the variable name to the null string.

`set` *name word*
Set the value of the variable *name* to *word*.

`set` *name expr*
Set the value of the variable name to the value of `expr`. `expr` must consist of more than one word. The value is encoded in unsigned hexadecimal, so that –1 is represented by 0xFFFFFFFF.

`setcolor`
Set the text mode display attributes. Allowable colors are *black*, *blue*, *green*, *cyan*, `red`, *magenta*, *brown*, *white*, *gray*, *lt_blue*, *lt_green*, *lt_cyan*, *lt_red*, *lt_magenta*, *yellow*, and *hi_white*.

`setprop` *propname word*
Set the value of the property *propname* to *word*.

`singlestep` or `singlestep on`
Turn on singlestep mode, in which the interpreter displays `step ?` before each command is processed, and waits for keyboard input. Press ENTER to execute the next command.

`singlestep off`
Turn off singlestep mode.

`source` *name*
Read the file *name* into memory and begin to interpret it. At EOF, return to the previous source of input.

`unset` *name*
Delete the variable name.

`verbose` or `verbose on`
Turn on verbose mode, which displays lines from source files and commands to be executed.

`verbose off`
Turn off verbose mode.

IA Built-in Functions

The following built-in functions are accepted within expressions:

`.strcmp`(*string1*, *string2*)	Returns an integer value that is less than, equal to, or greater than zero, as *string1* is lexicographically less than, equal to, or greater than *string2*.
`.strncmp`(*string1*, *string2*, *n*)	Returns an integer value that is less than, equal to, or greater than zero, as *string1* is lexicographically less than, equal to, or greater than *string2*. At most, *n* characters are compared.
`.streq` (*string1*, *string2*)	Returns true if *string1* is equal to *string2*, and false otherwise.
`.strneq` (*string1*, *string2*, *n*)	Returns true if *string1* is equal to *string2*, and false otherwise. At most, *n* characters are compared.
`.strfind` (*string*, *addr*, *n*)	Scans *n* locations in memory starting at *addr*, looking for the beginning of *string*. The *string* in memory need not be null-terminated. Returns true if *string* is found, and false otherwise. `.strfind` can be used to search for strings in the ROM BIOS and BIOS extensions that identify different machines and peripheral boards.

SPARC

EXAMPLE 1 To Boot the Default Kernel In Single-User Interactive Mode

To boot the default kernel in single-user interactive mode, respond to the `ok` prompt with one of the following:

```
boot -as

boot disk3 -as
```

32-bit SPARC

EXAMPLE 2 To Boot `kadb` Specifying The 32–Bit Kernel As The Default File

To boot `kadb` specifying the 32–bit kernel as the default file:

```
boot kadb -D kernel/unix
```

EXAMPLE 3 To Boot the 32-Bit Kernel Explicitly

To boot the 32-bit kernel explicitly, the kernel file name should be specified. So, to boot the 32-bit kernel in single-user interactive mode, respond to the `ok` prompt with one of the following:

EXAMPLE 3 To Boot the 32-Bit Kernel Explicitly *(Continued)*

```
boot kernel/unix -as

boot disk3 kernel/unix -as
```

64-bit SPARC

EXAMPLE 4 To Boot the 64-Bit Kernel Explicitly

To boot the 64-bit kernel explicitly, the kernel file name should be specified. So, to boot the 64-bit kernel in single-user interactive mode, respond to the `ok` prompt with one of the following:

```
boot kernel/sparcv9/unix -as

boot disk3 kernel/sparcv9/unix -as
```

Refer to the NOTES section `"Booting UltraSPARC Systems"` before booting the 64–bit kernel using an explicit filename.

IA

EXAMPLE 5 To Boot the Default Kernel In Single-User Interactive Mode

To boot the default kernel in single-user interactive mode, respond to the `>` prompt with one of the following:

```
b -as

b kernel/unix -as
```

FILES

`/platform/`*platform-name*`/ufsboot`
second level program to boot from a disk or CD.

`/etc/inittab`
table in which the "initdefault" state is specified.

`/sbin/init`
program that brings the system to the "initdefault" state.

`/platform/`*platform-name*`/boot.conf`
`/platform/`*hardware-class-name*`/boot.conf`
Primary and alternate pathnames for the `boot` policy file. Note that the policy file is not implemented on all platforms.

32-bit SPARC and IA

`/platform/`*platform-name*`/kernel/unix`
default program to boot system.

64-bit SPARC only

`/platform/`*platform-name*`/kernel/sparcv9/unix`
default program to boot system.

See NOTES section `"Booting UltraSPARC Systems."`

IA Only

`/etc/bootrc`
script that controls the booting process.

`/platform/`*platform-name*`/boot/solaris/boot.bin`
second level boot program used on IA systems in place of `ufsboot`.

`/platform/`*platform-name*`/boot`
directory containing boot-related files.

SEE ALSO

`uname`(1), `eeprom`(1M), `init`(1M), `installboot`(1M), `kadb`(1M), `kernel`(1M), `shutdown`(1M), `uadmin`(2), `bootparams`(4), `inittab`(4), `vfstab`(4), `filesystem`(5)

System Administration Guide: Basic Administration

Sun Hardware Platform Guide

OpenBoot Command Reference Manual

WARNINGS

The `boot` utility is unable to determine which files can be used as bootable programs. If the booting of a file that is not bootable is requested, the `boot` utility loads it and branches to it. What happens after that is unpredictable.

NOTES

platform-name can be found using the `-i` option of `uname`(1). *hardware-class-name* can be found using the `-m` option of `uname`(1).

64–bit SPARC

Booting UltraSPARC Systems

Certain platforms may need a firmware upgrade to run the 64-bit kernel. See the *Sun Hardware Platform Guide* for details. If the 64-bit kernel packages are installed and `boot` detects that the platform needs a firmware upgrade to run 64-bit, `boot` displays a message on the console and chooses the 32-bit kernel as the default file instead.

On systems containing 200MHz or lower UltraSPARC-1 processors, it is possible for a user to run a 64-bit program designed to exploit a problem that could cause a processor to stall. Since 64-bit progams cannot run on the 32-bit kernel, the 32-bit kernel is chosen as the default file on these systems.

The code sequence that exploits the problem is very unusual and is not likely to be generated by a compiler. Assembler code had to be specifically written to demonstrate the problem. It is highly unlikely that a legitimate handwritten assembler routine would use this code sequence.

Users willing to assume the risk that a user might accidentally or deliberately run a program that was designed to cause a processor to stall may choose to run the 64–bit kernel by modifying the `boot` policy file. Edit `/platform/`*platform-name*`/boot.conf` so that it contains an uncommented line with the variable named `ALLOW_64BIT_KERNEL_ON_UltraSPARC_1_CPU` set to the value `true` as shown in the example that follows:

```
ALLOW_64BIT_KERNEL_ON_UltraSPARC_1_CPU=true
```

For more information, see the *Sun Hardware Platform Guide*.

IA Only Because the "-" key on national language keyboards has been moved, an alternate key must be used to supply arguments to the `boot` command on an IA based system using these keyboards. Use the "-" on the numeric keypad. The specific language keyboard and the alternate key to be used in place of the "-" during bootup is shown below.

`Keyboard`	`Substitute Key`
Italy	'
Spain	'
Sweden	+
France	?
Germany	?

For example, `b -r` would be typed as `b +r` on Swedish keyboards, although the screen display will show as `b -r`.

NAME

bsmconv, bsmunconv – enable or disable the Basic Security Module (BSM) on Solaris

SYNOPSIS

/etc/security/bsmconv [*rootdir*...]

/etc/security/bsmunconv [*rootdir*...]

DESCRIPTION

The `bsmconv` and `bsmunconv` scripts are used to enable or disable the BSM features on a Solaris system. The optional argument *rootdir* is a list of one or more root directories of diskless clients which have already been configured by way of the Host Manager, see `admintool`(1M)

To enable or disable BSM on a diskless client, a server, or a stand-alone system, logon as super-user to the system being converted and use the `bsmconv` or `bsmunconv` commands without any options.

To enable or disable BSM on a diskless client from that client's server, logon to the server as super-user and use `bsmconv`, specifying the root directory of each diskless client you wish to affect. For example, the command:

```
myhost# bsmconv /export/root/client1 /export/root/client2
```

enables BSM on the two machines named `client1` and `client2`. While the command:

```
myhost# bsmconv
```

enables BSM only on the machine called `myhost`. It is no longer necessary to enable BSM on both the server and its diskless clients.

After running `bsmconv` the system can be configured by editing the files in `/etc/security`. Each diskless client has its own copy of configuration files in its root directory. You might want to edit these files before rebooting each client.

Following the completion of either script, the affected system(s) should be rebooted to allow the auditing subsystem to come up properly initialized.

FILES

The following files are created by `bsmconv`:

`/etc/security/device_maps`	Administrative file defining the mapping of device special files to allocatable device names.
`/etc/security/device_allocate`	Administrative file defining parameters for device allocation.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsr

SEE ALSO | `admintool`(1M), `auditconfig`(1M), `auditd`(1M), `audit_startup`(1M), `audit.log`(4), `audit_control`(4), `attributes`(5)

NAME

bsmrecord – display Basic Security Module (BSM) audit record formats

SYNOPSIS

```
/usr/bin/bsmrecord [-d] [ [-a] | [-e string] | [-c class] | [-i id] |
     [-p programname] | [-s systemcall] | [-h]]
```

DESCRIPTION

For audit record types defined in `/etc/security/audit_event`, the `bsmrecord` utility shows the audit id, audit class, selection mask, and record format. It can be used to generate a list of all audit records or to select records based on class, event name, program or system call name, or id. The `bsmrecord` utility requires root privileges.

There are two listing formats. The default format is intended for display in a terminal window; the optional HTML format is intended for viewing with a web browser.

OPTIONS

The following options are supported:

`-a`
List all audit records.

`-c` *class*
List all audit records selected by *class*. *class* is one of the two-character class codes from the file `/etc/security/audit_class`.

`-d`
Debug mode. Display number of audit records that are defined in `audit_event`, the number of classes defined in `audit_class`, any mismatches between the two files, and report which defined events do not have format information available to `bsmrecord`.

`-e` *string*
List all audit records for which the id label contains the string *string*. The match is case insensitive.

`-h`
Generate the output in HTML format.

`-i` *id*
List the audit record having the numeric id *id*.

`-p` *programname*
List all audit records generated by the program *programname*, for example, audit records generated by a user-space program.

`-s` *systemcall*
List all audit records generated by the system call *systemcall*, for example, audit records generated by a system call.

The `-p` and `-s` options are different names for the same thing and are mutually exclusive. The `-a` option is ignored if any of `-c`, `-e`, `-i`, `-p`, or `-s` are given. Combinations of `-c`, `-e`, `-i`, and either `-p` or `-s` are ANDed together.

EXAMPLES

EXAMPLE 1 Display Audit Record with a Specified ID

The following example shows how to display the contents of a specified audit record.

EXAMPLE 1 Display Audit Record with a Specified ID *(Continued)*

```
% bsmrecord -i 6152
login: terminal login
  program     /usr/sbin/login      see login(1)
  event ID    6152                 AUE_login
  class       lo                   (0x00001000)
      header-token
      subject-token
      text-token                   error message
      exit-token
```

EXAMPLE 2 Display Audit Record with ID Label that Contains a Specified String

The following example shows how to display the contents of a audit record with an id label that contains the string `login`.

```
# bsmrecord -e login
terminal login
  program     /usr/sbin/login      see login(1)
  event ID    6152                 AUE_login
  class       lo                   (0x00001000)
      header-token
      subject-token
      text-token                   error message
      exit-token

rlogin
  program     /usr/sbin/login      see login(1) - rlogin
  event ID    6155                 AUE_rlogin
  class       lo                   (0x00001000)
     header-token
     subject-token
     text-token                   error message
     exit-token
```

EXIT STATUS

`0` Successful operation

`non-zero` Error

FILES

`/etc/security/audit_class`
Provides the list of valid classes and the associated audit mask.

`/etc/security/audit_event`
Provides the numeric id, the literal event name, and the name of the associated system call or program.

`/etc/security/audit_display_attr`
Provides annotation information for each audit event. The primary content is the record format information but `audit_display_attr` also overrides the program name or system call information from `audit_event` as needed.

`/usr/perl5/5.6.1/lib/Sun/Solaris/BSM/_BSMparse.pm`
Perl module used by `bsmrecord`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsr
CSI	Enabled
Interface Stability	Unstable

SEE ALSO

`attributes`(5)

DIAGNOSTICS

If unable to read either of its input files or to write its output file, `bsmrecord` shows the name of the file on which it failed and exits with a non-zero return.

If no options are provided, if an invalid option is provided, or if both `-s` and `-p` are provided, an error message is displayed and `bsmrecord` displays a usage message then exits with a non-zero return.

NOTES

If the file permissions for `/etc/security/audit_event` or `/etc/security/audit_class` are not world-readable, then `bsmrecord` must be run by root.

If `/etc/security/audit_event` has been modified to add user-defined audit events, `bsmrecord` displays the record format as `undefined`.

NAME

busstat – report bus-related performance statistics

SYNOPSIS

busstat `-e` *device-inst* | `-h` | `-l`

busstat `[-a] [-n] [-w` *device-inst* `[,pic0=`*event*`,pic`*n*`=`*event* `]]…` `[-r` *device-inst*`]…` `[`*interval* `[`*count*`]]`

DESCRIPTION

`busstat` provides access to the bus-related performance counters in the system. These performance counters allow for the measurement of statistics like hardware clock cycles, bus statistics including DMA and cache coherency transactions on a multiprocessor system. Each bus device that supports these counters can be programmed to count a number of events from a specified list. Each device supports one or more Performance Instrumentation Counters (PIC) that are capable of counting events independently of each other.

Separate events can be selected for each PIC on each instance of these devices. `busstat` summarizes the counts over the last interval seconds, repeating forever. If a count is given, the statistics are repeated count times.

Only root users can program these counters. Non-root users have the option of reading the counters that have been programmed by a root user.

The default value for the *interval* argument is 1 second, and the default *count* is unlimited.

The devices that export these counters are highly platform-dependent and the data may be difficult to interpret without an in-depth understanding of the operation of the components that are being measured and of the system they reside in.

OPTIONS

The following options are supported:

`-a`
Display absolute counter values. The default is `delta` values.

`-e` *device-inst*
Display the list of events that the specified device supports for each pic.

Specify *device-inst* as device (`name`) followed by an optional instance number. If an instance number is specified, the events for that instance are displayed. If no instance number is specified, the events for the first instance of the specified device are displayed.

`-h`
Print a usage message.

`-l`
List the devices in the system which support performance counters.

`-n`
Do not display a title in the output. The default is to display titles.

`-r` *device-inst*
Read and display all pic values for the specified device

Specify *device-inst* as *device* (`name`) followed by *instance number*, if specifying an instance number of a device whose counters are to be read and displayed. If all instances of this device are to be read, use *device* (`name`) without an instance number. All pic values will be sampled when using the `-r` option.

`-w` *device-inst* [,pic0=*event*] [,pic*n*=*event*]
Program (write) the specified devices to count the specified events. Write access to the counters is restricted to root users only. Non-root users can use `-r` option.

Specify *device-inst* as *device* (`name`) followed by an optional *instance number*. If specifying an instance number of a device to program these events on. If all instances of this device are to be programmed the same, then use *device* without an instance number. Specify an event to be counted for a specified pic by providing a comma separated list of `pic`*n*=*event* values.

The `-e` option displays all valid event names for each device. Any devices that are programmed will be sampled every interval seconds and repeated count times. It is recommended that the interval specified is small enough to ensure that counter wraparound will be detected. The rate at which counters wraparound varies from device to device. If a user is programming events using the `-w` option and `busstat` detects that another user has changed the events that are being counted, the tool will terminate as the programmed devices are now being controlled by another user. Only one user can be programming a device instance at any one time. Extra devices can be sampled using the `-r` option. Using multiple instances of the `-w` option on the same command line, with the same *device-inst* specifying a different list of events for the pics will give the effect of multiplexing for that device. `busstat` will switch between the list of events for that device every interval seconds. Event can be a string representing the event name, or even a number representing the bit pattern to be programmed into the Performance Control Register (PCR). This assumes explicit knowledge of the meaning of the control register bits for a device. The number can be specified in hexadecimal, decimal, or octal, using the usual conventions of `strtol`(3C).

EXIT STATUS

The following exit values are returned:

0 Successful completion.

1 An error occurred.

2 Another user is writing to the same devices.

SPARC Only

EXAMPLE 1 Programming and monitoring the Address Controller counters

In this example, `ac0` refers to the Address Controller instance 0. The counters are programmed to count Memory Bank stalls on an Ultra Enterprise system at 10 second intervals with the values displayed in absolute form instead of deltas.

```
# busstat -a -w ac0,pic0=mem_bank0_stall,pic1=mem_bank1_stall 10
time  dev   event0            pic0   event1             pic1
10    ac0   mem_bank0_stall   1234   mem_bank1_stall    5678
20    ac0   mem_bank0_stall   5678   mem_bank1_stall   12345
30    ac0   mem_bank0_stall  12345   mem_bank1_stall   56789
```

EXAMPLE 1 Programming and monitoring the Address Controller counters *(Continued)*

```
...
```

For a complete list of the supported events for a device, use the `-e` option.

EXAMPLE 2 Programming and monitoring the counters on all instances of the Address Controller

In this example, `ac` refers to all `ac` instances. This example programs all instances of the Address Controller counters to `count_clock` cycles and `mem_bank0_rds` at 2 second intervals, 100 times, displaying the values as deltas.

```
# busstat -w ac,pic0=clock_cycles,pic1=mem_bank0_rds 2 100
time  dev     event0          pic0            event1          pic1
2     ac0     clock_cycles    167242902       mem_bank0_rds   3144
2     ac1     clock_cycles    167254476       mem_bank0_rds   1392
4     ac0     clock_cycles    168025190       mem_bank0_rds   40302
4     ac1     clock_cycles    168024056       mem_bank0_rds   40580
...
```

EXAMPLE 3 Monitoring the events being counted

This example monitors the events that are being counted on the sbus1 device, 100 times at 1 second intervals. It suggests that a root user has changed the events that `sbus1` was counting to be `dvma_tlb_misses` and interrupts instead of `pio_cycles`.

```
% busstat -r sbus0 1 100

time    dev     event0              pic0       event1          pic1
1       sbus1   pio_cycles          2321       pio_cycles      2321
2       sbus1   pio_cycles          48         pio_cycles      48
3       sbus1   pio_cycles          49         pio_cycles      49
4       sbus1   pio_cycles          2281       pio_cycles      2281
5       sbus1   dvma_tlb_misses     0          interrupts      0
6       sbus1   dvma_tlb_misses     6          interrupts      2
7       sbus1   dvma_tlb_misses     8          interrupts      11
...
```

EXAMPLE 4 Event Multiplexing

This example programs `ac0` to alternate between counting (clock cycles, `mem_bank0_rds`) and (`addr_pkts`, `data_pkts`) at 2 second intervals while also monitoring what `ac1` is counting :

It shows the expected output of the above `busstat` command. Another root user on the machine has changed the events that this user had programmed and `busstat` has detected this and terminates the command with a message.

```
# busstat -w ac0,pic0=clock_cycles,pic1=mem_bank0_rds \
          -w ac0,pic0=addr_pkts,pic1=data_pkts \
```

EXAMPLE 4 Event Multiplexing *(Continued)*

```
          -r ac1 2

time    dev     event0          pic0            event1          pic1
2       ac0     addr_pkts       12866           data_pkts       17015
2       ac1     rio_pkts        385             rio_pkts        385
4       ac0     clock_cycles    168018914       mem_bank0_rds   2865
4       ac1     rio_pkts        506             rio_pkts        506
6       ac0     addr_pkts       144236          data_pkts       149223
6       ac1     rio_pkts        522             rio_pkts        522
8       ac0     clock_cycles    168021245       mem_bank0_rds   2564
8       ac1     rio_pkts        387             rio_pkts        387
10      ac0     addr_pkts       144292          data_pkts       159645
10      ac1     rio_pkts        506             rio_pkts        506
12      ac0     clock_cycles    168020364       mem_bank0_rds   2665
12      ac1     rio_pkts        522             rio_pkts        522
busstat: events changed (possibly by another busstat).
#
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`iostat`(1M), `mpstat`(1M), `vmstat`(1M), `strtol`(3C), `attributes`(5)

NAME

cachefsd – CacheFS daemon

SYNOPSIS

/usr/lib/fs/cachefs/cachefsd

DESCRIPTION

The `cachefsd` server implements features of the cache filesystem (CacheFS). It is invoked at boot time and run if the / (root) and `/usr` filesystems are being cached. If `/usr` is being cached, `cachefsd` is invoked by `inetd`(1M) from `inetd.conf`(4). At run time, `cachefsd` is invoked by the `inetd` mechanism in response to an RPC request from a user command such as `mount_cachefs`(1M).

The `cachefsd` server supports the "disconnected mode" of CacheFS. In this mode, a user can continue to read and, depending on the option selected, write to files in a cached filesystem when the NFS server for the cached files is not available.

The `cachefsd` daemon performs the following functions in support of the CacheFS:

- Implements the connection policy. The daemon determines whether the NFS server backing the cache is connected or disconnected from the cache, or is in transition from the connected or disconnected states.
- Implements "log rolling," wherein the daemon monitors a disconnected NFS server for reconnection. After such a server returns to a connected state, `cachefsd` rolls any local changes to cached files (kept in a log) back to the server.
- Manages "packing," wherein `cachefsd` makes a best effort to ensure that files in a user-specified list are available in the cache in disconnected mode.
- Supports user interfaces by supplying statistics, reporting conflicts between the cache and the back filesystem, and supporting a list of files for packing.

The running of `cachefsd` is required for the disconnected mode of CacheFS.

OPTIONS

`-r` Used for invoking `cachefsd` for the / filesystem.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`cachefspack`(1M), `cfsadmin`(1M), `mount_cachefs`(1M), `inetd.conf`(4), `attributes`(5)

System Administration Guide: Basic Administration

NAME

cachefslog – Cache File System logging

SYNOPSIS

cachefslog [-f *logfile* | -h] *cachefs_mount_point*

DESCRIPTION

The `cachefslog` command displays where CacheFS statistics are being logged. Optionally, it sets where CacheFS statistics are being logged, or it halts logging for a cache specified by *cachefs_mount_point*. The *cachefs_mount_point* argument is a mount point of a cache file system. All file systems cached under the same cache as *cachefs_mount_point* will be logged.

OPTIONS

The following options are supported. You must be super-user to use the `-f` and `-h` options.

`-f` *logfile*	Specify the log file to be used.
`-h`	Halt logging.

OPERANDS

cachefs_mount_point	A mount point of a cache file system.

USAGE

See `largefile`(5) for the description of the behavior of `cachefslog` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXAMPLES

EXAMPLE 1 Checking the Logging of a directory.

The example below checks if the directory `/home/sam` is being logged:

```
example% cachefslog /home/sam
not logged: /home/sam
```

EXAMPLE 2 Changing the *logfile*.

The example below changes the *logfile* of `/home/sam` to `/var/tmp/samlog`:

```
example# cachefslog -f /var/tmp/samlog /home/sam
/var/tmp/samlog: /home/sam
```

EXAMPLE 3 Verifying the change of a *logfile*.

The example below verifies the change of the previous example:

```
example% cachefslog /home/sam
/var/tmp/samlog: /home/sam
```

EXAMPLE 4 Halting the logging of a directory.

The example below halts logging for the `/home/sam` directory:

```
example# cachefslog -h /home/sam
not logged: /home/sam
```

EXIT STATUS

The following exit values are returned:

`0`	success

non-zero an error has occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`cachefsstat`(1M), `cachefswssize`(1M), `cfsadmin`(1M), `attributes`(5), `largefile`(5)

DIAGNOSTICS

`Invalid path` It is illegal to specify a path within a cache file system.

NAME

cachefspack – pack files and file systems in the cache

SYNOPSIS

```
cachefspack [-h] [-i | -p | -u] [-f packing-list] [-U cache-directory]
     [file...]
```

DESCRIPTION

The `cachefspack` utility is used to set up and maintain files in the cache. This utility affords greater control over the cache, ensuring that the specified files will be in the cache whenever possible.

OPTIONS

The following options are supported:

`-f` *packing-list*	Specify a file containing a list of files and directories to be packed. Options within subdirectories and files can also be specified. The format and rules governing *packing-list* are described on the `packingrules`(4) manual page. Directories are packed recursively. Symlinks that match a regular expression on a `LIST` command are followed. Symlinks encountered while recursively processing directories are not followed.
`-h`	Help. Print a brief summary of all the options.
`-i`	View information about the packed files.
`-p`	Pack the file or files specified by `file`. This is the default behavior.
`-u`	Unpack the file or files specified by `file`.
`-U` *cache-directory*	Unpack all files in the specified cache directory.

OPERANDS

The following operands are supported:

`file` A path name of a file to be packed or unpacked.

USAGE

See `largefile`(5) for the description of the behavior of `cachefspack` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXAMPLES

EXAMPLE 1 The following example packs the file `projects` in the cache.

```
% cachefspack -p projects
```

EXAMPLE 2 The following example packs the files `projects`, `updates`, and `master_plan` in the cache.

```
% cachefspack -p projects updates master_plan
```

EXAMPLE 3 The following example unpacks the file `projects` from the cache.

```
% cachefspack -u projects
```

EXAMPLE 4 The following example unpacks the files `projects`, `updates`, and `master_plan` from the cache.

```
% cachefspack -u projects updates master_plan
```

EXAMPLE 5 The following example unpacks all files in the cache directory `cache1`.

```
% cachefspack -U /cache/cache1
```

EXAMPLE 6 The following example illustrates the use of a packing list to specify files to be packed in the cache. The contents of `lists.pkg` are as follows:

```
IGNORE SCCS BASE /src/junk LIST *.c LIST *.h
```

This example will pack all files in the directory `/src/junk` with `.c` and `.h` extensions that do not contained the string SCCS in the file's path name.

```
% cachefspack -f lists.pkg
```

EXIT STATUS

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`cfsadmin`(1M), `mount_cachefs`(1M), `packingrules`(4), `attributes`(5), `largefile`(5)

NAME

cachefsstat – Cache File System statistics

SYNOPSIS

/usr/bin/cachefsstat [-z] [*path*...]

DESCRIPTION

The cachefsstat command displays statistical information about the cache file system mounted on *path*. The statistical information includes cache hits and misses, consistency checking, and modification operations. If *path* is not specified, all mounted cache file systems are used.

cachefsstat can also be used to reinitialize this information (see -z option).

The statistical information has the following format:

<*cache hit rate*>
<*consistency checks*>
<*modifies*>
where:

hit rate	The percentage of cache hits over the total number of attempts, followed by the actual numbers of hits and misses.
consistency checks	The number of consistency checks performed, followed by the number that passed, and the number that failed.
modifies	The number of modify operations, including writes, creates, etc.

OPTIONS

The following option is supported:

-z	Zero (reinitialize) statistics. Execute cachefsstat -z before executing cachefsstat again to gather statistics on the cache performance. This option can only be use by the superuser. The statistics printed reflect those just before the statistics are reinitialized.

USAGE

See largefile(5) for the description of the behavior of cachefsstat when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXAMPLES

EXAMPLE 1 Example of cachefsstat.

```
example% cachefsstat /home/sam cache hit rate: 73% (1234 hits, 450
misses) consistency checks: 700 (650 pass, 50 fail) modifies: 321
```

EXIT STATUS

The following exit values are returned:

0	success
non-zero	an error has occurred.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO `cachefslog`(1M), `cachefswssize`(1M), `cfsadmin`(1M), `attributes`(5), `largefile`(5)

NAME

cachefswssize – determine working set size for cachefs

SYNOPSIS

`cachefswssize` *logfile*

DESCRIPTION

The `cachefswssize` command displays the workspace size determined from *logfile*. This includes the amount of cache space needed for each filesystem that was mounted under the cache, as well as a total.

USAGE

See `largefile`(5) for the description of the behavior of `cachefswssize` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXAMPLES

EXAMPLE 1 A sample output of `cachefswssize`.

```
example% cachefswssize /var/tmp/samlog
```

```
/home/sam
                        end size:               10688k
                        high water size:        10704k

/foo
                        end size:               128k
                        high water size:        128k

/usr/dist
                        end size:               1472k
                        high water size:        1472k

total for cache
                        initial size:           110960k
                        end size:               12288k
                        high water size:        12304k
```

EXIT STATUS

The following exit values are returned:

`0`	success
non-zero	an error has occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`cachefslog`(1M), `cachefsstat`(1M), `cfsadmin`(1M), `attributes`(5), `largefile`(5)

DIAGNOSTICS

`problems were encountered writing log file`
There were problems encountered when the kernel was writing the logfile. The most common problem is running out of disk space.

`invalid log file`
The logfile is not a valid logfile or was created with a newer version of Solaris than the one where `cachefswssize` is running.

NAME

captoinfo – convert a termcap description into a terminfo description

SYNOPSIS

captoinfo [-1] [-v...] [-V] [-w *width*] *filename*...

DESCRIPTION

captoinfo looks in *filename* for termcap descriptions. For each one found, an equivalent terminfo description is written to standard output, along with any comments found. A description which is expressed as relative to another description (as specified in the termcap tc = field) is reduced to the minimum superset before being displayed.

If no *filename* is given, then the environment variable TERMCAP is used for the filename or entry. If TERMCAP is a full pathname to a file, only the terminal whose name is specified in the environment variable TERM is extracted from that file. If the environment variable TERMCAP is not set, then the file /usr/share/lib/termcap is read.

OPTIONS

–1	Display the fields one to a line. Otherwise, the fields are printed several to a line, with a maximum width of 60 characters.
-v	Display tracing information on the standard error as the program runs. Specifying additional -v options displays more detailed information.
-V	Display the version of the program in use on the standard error and then exit.
-w *width*	Change the output to *width* characters.

FILES

/usr/share/lib/terminfo/?/*	compiled terminal description database
/usr/share/lib/termcap	

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

infocmp(1M), curses(3CURSES), terminfo(4), attributes(5)

NOTES

captoinfo should be used to convert termcap entries to terminfo entries because the termcap database may not be supplied in future releases.

NAME

catman – create the formatted files for the reference manual

SYNOPSIS

/usr/bin/catman [-c] [-n] [-p] [-t] [-w] [-M *directory*] [-T *macro-package*] [*sections*]

DESCRIPTION

The catman utility creates the preformatted versions of the on-line manual from the nroff(1) or sgml(5) input files. This feature allows easy distribution of the preformatted manual pages among a group of associated machines (for example, with rdist(1)), since it makes the directories of preformatted manual pages self-contained and independent of the unformatted entries.

catman also creates the windex database file in the directories specified by the MANPATH or the -M option. The windex database file is a three column list consisting of a keyword, the reference page that the keyword points to, and a line of text that describes the purpose of the utility or interface documented on the reference page. Each keyword is taken from the comma separated list of words on the NAME line before the '–' (dash). The reference page that the keyword points to is the first word on the NAME line. The text after the – on the NAME line is the descriptive text in the third column. The NAME line must be immediately preceded by the page heading line created by the .TH macro (see NOTES for required format).

Each manual page is examined and those whose preformatted versions are missing or out of date are recreated. If any changes are made, catman recreates the windex database.

If a manual page is a *shadow* page, that is, it sources another manual page for its contents, a symbolic link is made in the cat*x* or fmt*x* directory to the appropriate preformatted manual page.

Shadow files in an unformatted nroff source file are identified by the first line being of the form .so man*x*/yyy.*x*.

Shadow files in the SGML sources are identified by the string SHADOW_PAGE. The file entity declared in the shadow file identifies the file to be sourced.

OPTIONS

The following options are supported:

-c — Create unformatted nroff source files in the appropriate man subdirectories from the SGML sources. This option will overwrite any existing file in the man directory of the same name as the SGML file.

-n — Do not create (or recreate) the windex database. If the -n option is specified, the windex database is not created and the apropos, whatis, man -f, and man -k commands will fail.

-p — Print what would be done instead of doing it.

`-t`	Create `troff`ed entries in the appropriate `fmt` subdirectories instead of `nroff`ing into the `cat` subdirectories.
`-w`	Only create the `windex` database that is used by `whatis`(1) and the `man`(1) `-f` and `-k` options. No manual reformatting is done.
`-M` *directory*	Update manual pages located in the specified *directory*, (`/usr/share/man` by default). If the `-M` option is specified, the directory argument must not contain a ',' (comma), since a comma is used to delineate section numbers. See `man`(1).
`-T` *macro-package*	Use *macro-package* in place of the standard manual page macros, (`man`(5) by default).

OPERANDS

The following operand is supported:

sections If there is one parameter not starting with a '–', it is taken to be a space separated list of manual sections to be processed by `catman`. If this operand is specified, only the manual sections in the list will be processed. For example,

```
catman 1 2 3
```

only updates manual sections `1`, `2`, and `3`. If specific sections are not listed, all sections in the `man` directory specified by the environment variable `MANPATH` are processed.

ENVIRONMENT VARIABLES

`TROFF` The name of the formatter to use when the `-t` flag is given. If not set, `troff`(1) is used.

`MANPATH` A colon-separated list of directories that are processed by `catman` and `man`(1). Each directory can be followed by a comma-separated list of sections. If set, its value overrides `/usr/share/man` as the default directory search path, and the `man.cf` file as the default section search path. The `-M` and `-s` flags, in turn, override these values.

FILES

`/usr/share/man`	default manual directory location
`/usr/share/man/man*/*.*`	raw nroff input files
`/usr/share/man/sman*/*.*`	raw SGML input files
`/usr/share/man/cat*/*.*`	preformatted `nroff`ed manual pages
`/usr/share/man/fmt*/*.*`	preformatted `troff`ed manual pages
`/usr/share/man/windex`	table of contents and keyword database
`/usr/lib/makewhatis`	command script to make `windex` database
`/usr/share/lib/tmac/an`	default macro package

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWdoc
CSI	Enabled

SEE ALSO

apropos(1), man(1), nroff(1), rdist(1), rm(1), troff(1), whatis(1), attributes(5), man(5), sgml(5)

DIAGNOSTICS

`man?/xxx.? (.so'ed from man?/yyy.?): No such file or directory`
The file outside the parentheses is missing, and is referred to by the file inside them.

`target of .so in man?/xxx.? must be relative to /usr/man`
catman only allows references to filenames that are relative to the directory /usr/man.

`opendir:man?: No such file or directory`
A harmless warning message indicating that one of the directories catman normally looks for is missing.

`*.*: No such file or directory`
A harmless warning message indicating catman came across an empty directory.

WARNINGS

If a user, who has previously run catman to install the cat* directories, upgrades the operating system, the entire cat* directory structure should be removed prior to running catman. See rm(1).

Do not re-run catman to re-build the whatis database unless the complete set of man* directories is present. catman builds this windex file based on the man* directories.

NOTES

To generate a valid windex index file, catman has certain requirements. Within the individual man page file, catman requires two macro lines to have a specific format. These are the .TH page heading line and the .SH NAME line.

The .TH macro requires at least the first three arguments, that is, the filename, section number, and the date. The .TH line starts off with the .TH macro, followed by a space, the man page filename, a single space, the section number, another single space, and the date. The date should appear in double quotes and is specified as "day month year," with the month always abbreviated to the first three letters (Jan, Feb, Mar, and so forth).

The .SH NAME macro, also known as the NAME line, must immediately follow the .TH line, with nothing in between those lines. No font changes are permitted in the NAME line. The NAME line is immediately followed by a line containing the man page filename; then shadow page names, if applicable, separated by commas; a dash; and a brief summary statement. These elements should all be on one line; no carriage returns are permitted.

An example of proper coding of these lines is:

```
.TH nismatch 1M "10 Apr 1998"
.SH NAME
nismatch, nisgrep \- utilities for searching NIS+ tables
```

NAME

cfgadm – configuration administration

SYNOPSIS

/usr/sbin/cfgadm [-f] [-y | -n] [-v] [-o *hardware_options*] -c *function ap_id...*

/usr/sbin/cfgadm [-f] [-y | -n] [-v] [-o *hardware_options*] -x *hardware_function ap_id...*

/usr/sbin/cfgadm [-v] [-a] [-s *listing_options*] [-o *hardware_options*] [-l [*ap_id* | *ap_type*]]

/usr/sbin/cfgadm [-v] [-o *hardware_options*] -t *ap_id...*

/usr/sbin/cfgadm [-v] [-o *hardware_options*] -h [*ap_id* | *ap_type*]

DESCRIPTION

The `cfgadm` command provides configuration administration operations on dynamically reconfigurable hardware resources. These operations include displaying status, (`-l`), initiating testing, (`-t`), invoking configuration state changes, (`-c`), invoking hardware specific functions, (`-x`), and obtaining configuration administration help messages (`-h`). Configuration administration is performed at *attachment points*, which are places where system software supports dynamic reconfiguration of hardware resources during continued operation of Solaris.

Configuration administration makes a distinction between hardware resources that are physically present in the machine and hardware resources that are configured and visible to Solaris. The nature of configuration administration functions are hardware specific, and are performed by calling hardware specific libraries.

Configuration administration operates on an *attachment point*. Hardware resources located at attachment points can or can not be physically replaceable during system operation, but are dynamically reconfigurable by way of the configuration administration interfaces.

An attachment point defines two unique elements, which are distinct from the hardware resources that exist beyond the attachment point. The two elements of an attachment point are a *receptacle* and an *occupant*. Physical insertion or removal of hardware resources occurs at attachment points and results in a receptacle gaining or losing an occupant. Configuration administration supports the physical insertion and removal operations as well as other configuration administration functions at an attachment point.

Attachment points have associated state and condition information. The configuration administration interfaces provide control for transitioning attachment point states. A receptacle can exist in one of three states: `empty`, `disconnected` or `connected`, while an occupant can exist in one of two states: `configured` or `unconfigured`.

A receptacle can provide the `empty` state, which is the normal state of a receptacle when the attachment point has no occupants. A receptacle can also provide the `disconnected` state if it has the capability of isolating its occupants from normal system access. Typically this state is used for various hardware specific testing prior to bringing the occupant's resources into full use by the system, or as a step in preparing

an occupant for physical removal or reconfiguration. A receptacle in the disconnected state isolates its occupant from the system as much as its hardware allows, but can provide access for testing and setup. A receptacle must provide the `connected` state, which allows normal access to hardware resources contained on any occupants. The connected state is the normal state of a receptacle that contains an occupant and that is not currently undergoing configuration administration operations.

The hardware resources contained on an occupant in the `unconfigured` state are not represented by normal Solaris data structures and are thus not available for use by Solaris. Operations allowed on an unconfigured occupant are limited to configuration administration operations. The hardware resources of an occupant in the `configured` state are represented by normal Solaris data structures and thus some or all of those hardware resources can be in use by Solaris. All occupants provide both the `configured` and `unconfigured` states,

An attachment point can be in one of five conditions: `unknown`, `ok`, `failing`, `failed`, or `unusable`. An attachment point can enter the system in any condition depending upon results of power-on tests and non-volatile record keeping.

An attachment point with an occupant in the `configured` state is in one of four conditions: `unknown`, `ok`, `failing`, or `failed`. If the condition is not `failing` or `failed` an attachment point can change to `failing` during the course of operation if a hardware dependent recoverable error threshold is exceeded. If the condition is not `failed` an attachment point can change to `failed` during operation as a result of an unrecoverable error.

An attachment point with an occupant in the `unconfigured` state can be in any of the defined conditions. The condition of an attachment point with an `unconfigured` occupant can decay from `ok` to `unknown` after a machine dependent time threshold. Initiating a test function changes the attachment point's condition to `ok`, `failing` or `failed` depending on the outcome of the test. An attachment point that does not provide a test function can leave the attachment point in the `unknown` condition. If a test is interrupted, the attachment point's condition can be set to the previous condition, `unknown` or `failed`. An attachment point in the `unknown`, `ok`, `failing`, or `failed` conditions can be re-tested.

An attachment point can exist in the `unusable` condition for a variety of reasons, such as inadequate power or cooling for the receptacle, an occupant that is unidentifiable, unsupported, incorrectly configured, etc. An attachment point in the `unusable` condition can never be used by the system. It typically remains in this condition until the physical cause is remedied.

An attachment point also maintains busy information that indicates when a state change is in progress or the condition is being reevaluated.

Attachment points are referred to using hardware specific identifiers (*ap_id*s) that are related to the type and location of the attachment points in the system device hierarchy. An *ap_id* can not be ambiguous, it must identify a single attachment point.

Two types of *ap_id* specifications are supported: physical and logical. A physical *ap_id* contains a fully specified pathname, while a logical *ap_id* contains a shorthand notation that identifies an attachment point in a more user-friendly way.

For example, an attachment point representing a system's backplane slot number `7` could have a physical *ap_id* of `/devices/central/fhc/sysctrl:slot7` while the logical *ap_id* could be `system:slot7`. Another example, the third receptacle on the second PCI I/O bus on a system could have a logical *ap_id* of `pci2:plug3`.

Attachment points may also be created dynamically. A dynamic attachment point is named relative to a base attachment point which is present in the system. `ap_id`s for dynamic attachment points consist of a base component followed by two colons (`::`) and a dynamic component. The base component is the base attachment point `ap_id`. The dynamic component is hardware specific and generated by the corresponding hardware specific library.

For example, consider a base attachment point, which represents a SCSI HBA, with the physical `ap_id` `/devices/sbus@1f,0/SUNW,fas@e,8800000:scsi` and logical `ap_id` `c0` . A disk attached to this SCSI HBA could be represented by a dynamic attachment point with logical `ap_id` `c0::dsk/c0t0d0` where `c0` is the base component and `dsk/c0t0d0` is the hardware specific dynamic component. Similarly the physical `ap_id` for this dynamic attachment point would be: `/devices/sbus@1f,0/SUNW,fas@e,8800000:scsi::dsk/c0t0d0`

An *ap_type* is a partial form of a logical `ap_id` that can be ambiguous and not specify a particular attachment point. An *ap_type* is a substring of the portion of the logical `ap_id` up to but not including the colon (`:`) separator. For example, an *ap_type* of `pci` would show all attachment points whose logical *ap_id*s begin with `pci`.

The use of *ap_types* is discouraged. The new select sub-option to the `-s` option provides a more general and flexible mechanism for selecting attachment points. See `OPTIONS`.

The `cfgadm` command interacts primarily with hardware dependent functions contained in hardware specific libraries and thus its behavior is hardware dependent.

For each configuration administration operation a service interruption can be required. Should the completion of the function requested require a noticeable service interruption to interactive users, a prompt is output on the standard error output for confirmation on the standard input before the function is started. Confirmation can be overridden using the `-y` or `-n` options to always answer yes or no respectively. Hardware specific options, such as test level, are supplied as sub-options using the `-o` option.

Operations that change the state of the system configuration are audited by the system log daemon `syslogd`(1M).

The arguments for this command conform to the `getopt`(3C) and `getsubopt`(3C) syntax convention.

OPTIONS The following options are supported:

`-a`
Specifies that the `-l` option must also list dynamic attachment points.

`-c` *function*
Performs the state change *function* on the attachment point specified by *ap_id.*

Specify *function* as `insert`, `remove`, `disconnect`, `connect`, `configure` or `unconfigure`. These functions cause state transitions at the attachment point by calling hardware specific library routines and are defined in the following list.

`insert`	Performs operations that allows the user to manually insert an occupant or to activate a hardware supplied mechanism that performs the physical insertion. `insert` can have hardware specific side effects that temporarily suspend activity in portions of the system. In such cases the hardware specific library generates appropriate warning messages and informs the user of any special considerations or procedures unique to that hardware. Various hardware specific errors can cause this function to fail and set the receptacle condition to `unusable`.
`remove`	Performs operations that allow the user to manually remove an occupant or to activate a hardware supplied mechanism to perform the physical removal. `remove` can have hardware specific side effects that temporarily suspend activity in portions of the system. In such cases the hardware specific library generates appropriate warning messages and informs the user of any special considerations or procedures unique to that hardware. Various hardware specific errors can cause this function to fail and set the receptacle condition to unusable.
`disconnect`	Performs hardware specific operations to put a receptacle in the disconnected state, which can prevent an occupant from operating in a normal fashion through the receptacle.
`connect`	Performs hardware specific operations to put the receptacle in the `connected` state, which allows an occupant to operate in a normal fashion through the receptacle.
`configure`	Performs hardware specific operations that allow an occupant's hardware resources to be usable by Solaris. Occupants that are configured are part of the system configuration and are available for manipulation by Solaris device manipulation maintenance commands (eg: `psradm`(1M), `mount`(1M), `ifconfig`(1M)).
`unconfigure`	Performs hardware specific operations that logically remove an occupant's hardware resources from the system. The occupant must currently be configured and its hardware resources must not be in use by Solaris.

State transition functions can fail due to the condition of the attachment point or other hardware dependent considerations. All state change *function*s in the direction of adding resources, (`insert`, `connect` and `configure`) are passed onto the hardware specific library when the attachment point is in the `ok` or `unknown` condition. All other conditions require the use of the force option to allow these *function*s to be passed on to the hardware specific library. Attachment point condition does not prevent a hardware specific library being called for related to the removal (`remove`, `disconnect` and `unconfigure`), of hardware resources from the system. Hardware specific libraries can reject state change *function*s if the attachment point is in the `unknown` condition.

The condition of an attachment point is not necessarily changed by the state change functions, however errors during state change operations can change the attachment point condition. An attempt to override a condition and force a state change that would otherwise fail can be made by specifying the force option (`-f`). Hardware specific safety and integrity checks can prevent the force option from having any effect.

`-f`
Forces the specified action to occur. Typically, this is a hardware dependent override of a safety feature. Forcing a state change operation can allow use of the hardware resources of occupant that is not in the `ok` or `unknown` conditions, at the discretion of any hardware dependent safety checks.

`-h` [*ap_id* | *ap_type* ...]
Prints out the help message text. If *ap_id* or *ap_type* is specified, the help routine of the hardware specific library for the attachment point indicated by the argument is called.

`-l` [*ap_id* | *ap_type* ...]
Lists the state and condition of attachment points specified. Attachment points can be filtered by using the `-s` option and `select` sub-option. Invoking `cfgadm` without one of the action options is equivalent to `-l` without an argument. The format of the list display is controlled by the `-v` and `-s` options. When the `-a` option is specified attachment points are dynamically expanded.

`-n`
Suppress any interactive confirmation and assume that the answer is *no*. If neither `-n` or `-y` is specified, interactive confirmation is obtained through the standard error output and the standard input. If either of these standard channels does not correspond to a terminal (as determined by `isatty`(3C)) then the `-n` option is assumed.

`-o` *hardware_options*
Supplies hardware specific options to the main command option. The format and content of the hardware option string is completely hardware specific. The option string *hardware_options* conforms to the `getsubopt`(3C) syntax convention.

`-s` *listing_options*
Supplies listing options to the list (`-l`) command. *listing_options* conforms to the `getsubopt`(3C) syntax convention. The sub-options are used to specify the

attachment point selection criteria (`select=`*select_string*), the type of matching desired (`match=`*match_type*), order of listing (`sort=`*field_spec*), the data that is displayed (`cols=`*field_spec* and `cols2=`*field_spec*), the column delimiter (`delim=`*string*) and whether to suppress column headings (`noheadings`).

When the `select` sub-option is specified, only attachment points which match the specified criteria will be listed. The `select` sub-option has the following syntax:

```
cfgadm -s select=attr1(value1):attr2(value2)...
```

where an *attr* is one of `ap_id`, `class` or `type`. `ap_id` refers to the logical *ap_id* field, `class` refers to attachment point class and `type` refers to the type field. *value1*, *value2*, etc. are the corresponding values to be matched. The type of match can be specified by the `match` sub-option as follows:

```
cfgadm -s match=match_type,select=attr1(value1)...
```

where *match_type* can be either `exact` or `partial`. The default value is `exact`.

Arguments to the `select` sub-option can be quoted to protect them from the shell.

A *field_spec* is one or more *data-field*s concatenated using colon (`:`), as in *data-field*:*data-field*:*data-field*. A *data-field* is one of `ap_id`, `physid`, `r_state`, `o_state`, `condition`, `type`, `busy`, `status_time`, `status_time_p`, `class`, and `info`. The `ap_id` field output is the logical name for the attachment point, while the `physid` field contains the physical name. The `r_state` field can be `empty`, `disconnected` or `connected`. The `o_state` field can be `configured` or `unconfigured`. The `busy` field can be either `y` if the attachment point is busy, or `n` if it is not. The `type` and `info` fields are hardware specific. The `status_time_p` field is a parsable version of the `status_time` field. If an attachment point has an associated class, the `class` field lists the class name. If an attachement point does not have an associated class, the `class` field lists `none`.

The order of the fields in *field_spec* is significant: For the `sort` sub-option, the first field given is the primary sort key. For the `cols` and `cols2` sub-options, the fields are printed in the order requested. The order of sorting on a *data-field* can be reversed by placing a minus (–) before the *data-field* name within the *field_sec* for the `sort` sub-option. The default value for `sort` is `ap_id`. The defaults values for `cols` and `cols2` depend on whether the `-v` option is given: Without it `cols` is `ap_id:r_state:o_state:condition` and `cols2` is not set. With `-v` `cols` is `ap_id:r_state:o_state:condition:info` and `cols2` is `status_time:type:busy:physid:`. The default value for `delim` is a single space. The value of `delim` can be a string of arbitrary length. The delimiter cannot include comma (`,`) character, see `getsubopt`(3C). These listing options can be used to create parsable output. See `NOTES`.

`-t`

Performs a test of one or more attachment points. The test function is used to re-evaluate the condition of the attachment point. Without a test level specifier in *hardware_options*, the fastest test that identifies hard faults is used.

More comprehensive tests are hardware specific and are selected using the *hardware_options*.

The results of the test is used to update the condition of the specified occupant to either `ok` if no faults are found, `failing` if recoverable faults are found or `failed` if any unrecoverable faults are found.

If a test is interrupted, the attachment point's condition can be restored to its previous value or set to `unknown` if no errors were found or `failing` if only recoverable errors were found or to `failed` if any unrecoverable errors were found. The attachment point should only be set to `ok` upon normal completion of testing with no errors.

`-v`
Executes in verbose mode. For the `-c`, `-t` and `-x` options outputs a message giving the results of each attempted operation. Outputs detailed help information for the `-h` option. Outputs verbose information for each attachment point for the `-l` option.

`-x` *hardware_function*
Performs hardware specific functions. Private hardware specific functions can change the state of a receptacle or occupant. Attachment point conditions can change as the result of errors encountered during private hardware specific functions. The format and content of the *hardware_function* string is completely hardware specific. The option string *hardware_function* conforms to the `getsubopt`(3C) syntax convention.

`-y`
Suppresses any interactive confirmation and assume that the answer is `yes`.

USAGE

The required privileges to use this command are hardware dependent. Typically, a default system configuration restricts all but the list option to the superuser.

EXAMPLES

EXAMPLE 1 Listing attachment points in the device tree

The following example lists all attachment points except dynamic attachment points.

```
example# cfgadm

  Ap_Id          Type         Receptacle      Occupant       Cond
system:slot0     cpu/mem      connected       configured     ok
system:slot1     sbus-upa     connected       configured     ok
system:slot2     cpu/mem      connected       configured     ok
system:slot3     unknown      connected       unconfigured   unknown
system:slot4     dual-sbus    connected       configured     failing
system:slot5     cpu/mem      connected       configured     ok
system:slot6     unknown      disconnected    unconfigured   unusable
system:slot7     unknown      empty           unconfigured   ok
c0               scsi-bus     connected       configured     unknown
c1               scsi-bus     connected       configured     unknown
```

EXAMPLE 2 Listing all configurable hardware information

The following example lists all current configurable hardware information, including those represented by dynamic attachment points:

```
example# cfgadm -al
```

EXAMPLE 2 Listing all configurable hardware information *(Continued)*

```
  Ap_Id            Type         Receptacle      Occupant        Cond
system:slot0       cpu/mem      connected       configured      ok
system:slot1       sbus-upa     connected       configured      ok
system:slot2       cpu/mem      connected       configured      ok
system:slot3       unknown      connected       unconfigured    unknown
system:slot4       dual-sbus    connected       configured      failing
system:slot5       cpu/mem      connected       configured      ok
system:slot6       unknown      disconnected    unconfigured    unusable
system:slot7       unknown      empty           unconfigured    ok
c0                 scsi-bus     connected       configured      unknown
c0::dsk/c0t14d0    disk         connected       configured      unknown
c0::dsk/c0t11d0    disk         connected       configured      unknown
c0::dsk/c0t8d0     disk         connected       configured      unknown
c0::rmt/0          tape         connected       configured      unknown
c1                 scsi-bus     connected       configured      unknown
```

EXAMPLE 3 Selective listing based on attachment point attributes

The following example lists all attachment points whose class begins with `scsi`, `ap_id` begins with `c` and `type` field begins with `scsi`. The argument to the `-s` option is quoted to protect it from the shell.

```
example# cfgadm -s "match=partial,select=class(scsi):ap_id(c):type(scsi)"

Ap_Id         Type          Receptacle      Occupant          Cond
 c0          scsi-bus      connected       configured        unknown
 c1          scsi-bus      connected       configured        unknown
```

EXAMPLE 4 Listing current configurable hardware information in verbose mode.

The following example lists current configurable hardware information for *ap-type* `system` in verbose mode:

```
example# cfgadm -v -l system
Ap_Id                     Receptacle Occupant   Condition Information
When         Type      Busy     Phys_Id
system:slot1                 connected  configured ok
Apr  4 23:50 sbus-upa  n        /devices/central/fhc/sysctrl:slot1
system:slot3                 connected  configured ok        non-detachable
Apr 17 11:20 cpu/mem   n        /devices/central/fhc/sysctrl:slot3
system:slot5                 connected  configured ok
Apr  4 23:50 cpu/mem   n        /devices/central/fhc/sysctrl:slot5
system:slot7                 connected  configured ok
Apr  4 23:50 dual-sbus n        /devices/central/fhc/sysctrl:slot7
```

EXAMPLE 5 The hardware specific `extended` test.

The following example tests two occupants using the hardware specific `extended` test:

```
example# cfgadm -v -o extended -t system:slot3 system:slot5
Testing attachment point system:slot3 ...  ok
Testing attachment point system:slot5 ...  ok
```

EXAMPLE 5 The hardware specific `extended` test. *(Continued)*

EXAMPLE 6 The force option.

The following example configures an occupant in the `failing` state to the system using the force option:

```
example# cfgadm -f -c configure system:slot3
```

EXAMPLE 7 Unconfiguring an occupant from the system.

The following example unconfigures an occupant from the system:

```
example# cfgadm -c unconfigure system:slot4
```

EXAMPLE 8 Configuring an occupant at an attachment point

The following example configures an occupant:

```
example# cfgadm -c configure c0::dsk/c0t0d0
```

ENVIRONMENT VARIABLES

See `environ`(5) for descriptions of the following environment variables that affect the execution of `cfgadm`: `LC_TIME`, `LC_MESSAGES`, `NLSPATH` and `TZ`.

`LC_MESSAGES` Determines how `cfgadm` displays column headings and error messages. Listing output data is not affected by the setting of this variable.

`LC_TIME` Determines how `cfgadm` displays human readable status changed time (`status_time`).

`TZ` Specifies the timezone used when converting the status changed time. This applies to both the human readable (`status_time`) and parsable (`status_time_p`) formats.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` An error occurred.

`2` Configuration administration not supported on specified target.

`3` Usage error.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO cfgadm_pci(1M), cfgadm_sbd(1M), cfgadm_usb(1M), cfgadm_scsi(1M), ifconfig(1M), mount(1M), prtdiag(1M), psradm(1M), syslogd(1M), config_admin(3CFGADM), getopt(3C), getsubopt(3C), isatty(3C), attributes(5), environ(5)

DIAGNOSTICS Diagnostic messages appear on the standard error output. Other than options and usage errors, the following are diagnostic messages produced by this utility:

cfgadm: Configuration administration not supported on *ap_id*

cfgadm: No library found for *ap_id*

cfgadm: *ap_id* is ambiguous

cfgadm: *operation*: Insufficient privileges

cfgadm: Attachment point is busy, try again

cfgadm: No attachment points with specified attributes found

cfgadm: System is busy, try again

cfgadm: *operation*: Operation requires a service interruption

cfgadm: *operation*: Data error: *error_text*

cfgadm: *operation*: Hardware specific failure: *error_text*

See config_admin(3CFGADM) for additional details regarding error messages.

NOTES Hardware resources enter the unconfigured pool in a hardware specific manner. This can occur at various times such as: system initialization or as a result of an unconfigure operation. An occupant that is in the unconfigured state is not available for use by the system until specific intervention occurs. This intervention can be manifested as an operator initiated command or it can be by way of an automatic configuring mechanism.

The listing option of the cfgadm command can be used to provide parsable input for another command, for example within a shell script. For parsable output, the -s option must be used to select the fields required. The -s option can also be used to suppress the column headings. The following fields always produce parsable output: ap_id, physid, r_state, o_state, condition, busy status_time_p, class, and type. Parsable output never has white-space characters embedded in the field value.

The following shell script fragment finds the first good unconfigured occupant of type CPU.

```
found=
cfgadm -l -s "noheadings,cols=ap_id:r_state:condition:type" | \
while read ap_id r_state cond type
do
     if [ "$r_state" = unconfigured -a "$cond" = ok -a "$type" = CPU ]
     then
          if [ -z "$found" ]
```

```
          then
               found=$ap_id
          fi
     fi
done
if [ -n "$found" ]
then
         echo "Found CPU $found"
fi
```

The format of the parsable time field (`status_time_p`) is *YYYYMMDDhhmmss*, giving the year, month, day, hour, minute and second in a form suitable for string comparison.

Reference should be made to the hardware specific documentation for details of System Configuration Administration support.

NAME

cfgadm_ac – EXX00 memory system administration

SYNOPSIS

```
/usr/sbin/cfgadm [-c configure] [-f] [-o disable-at-boot |
    enable-at-boot ] ac#:bank# …

/usr/sbin/cfgadm [-c unconfigure] [-o disable-at-bootp |
    enable-at-boot ] ac#:bank# …

/usr/sbin/cfgadm [-v] [-o quick | normal | extended,
    [max_errors=#] ] -t ac#:bank#…

/usr/sbin/cfgadm -x relocate-test ac#:bank# …

/usr/sbin/cfgadm [-l] -o disable-at-boot | enable-at-boot
    ac#:bank# …
```

DESCRIPTION

The `ac` hardware specific library `/usr/platform/sun4u/lib/cfgadm/cfgadm_ac.so.1` provides the functionality for configuring and unconfiguring memory banks on `E6X00`, `E5X00`, `E4X00` and `E3X00` systems as part of the Dynamic Reconfiguration of CPU/Memory boards using `cfgadm_sysctrl`(1M).

Memory banks appear as attachment points in the device tree. For each CPU/Memory board, two attachment points are published, one for each bank on the board: `bank0` and `bank1`. If the bank is unpopulated, the receptacle state is empty. If the bank is populated, the receptacle state is connected. The receptacle state of a memory bank can never be disconnected. The occupant state of a connected memory bank can be configured or unconfigured. If the occupant state is configured, the memory is in use by Solaris, if unconfigured it is not.

OPTIONS

Refer to `cfgadm`(1M) for complete descriptions of the command options.

The following options are supported:

`-c` configure | unconfigure

Change the occupant state. The `configure` argument ensures that the memory is initialized and adds the memory to the Solaris memory pool. The `unconfigure` argument removes the memory from use by Solaris. When a CPU/Memory board is to be removed from a system, both banks of memory must be unconfigured.

`cfgadm` refuses the `configure` operation if the memory on the board is marked `disabled-at-boot` (see `info` field), unless either the `-f` (force) option or the enable at boot flag, (`-o enable-at-boot`), is given. The configure operation takes a short time proportional to the size of memory that must be initialized.

`cfgadm` refuses the `unconfigure` operation if there is not enough uncommitted memory in the system (`VM viability` error) or if the bank to be unconfigured has memory that can't be removed (`non-relocatable pages` error). The presence of non-relocatable pages is indicated by the word `permanent` in the `info` listing field. Removing memory from use by Solaris may take a significant time due to factors such as system load and how much paging to secondary storage is required. The `unconfigure` operation can be cancelled at any time and the memory

returned to the fully configured state by interrupting the command invocation with a signal. The `unconfigure` operation self-cancels if no memory can be removed within a timeout period. The default timeout period of `60` seconds can be changed using the `-o timeout=#` option, with a value of `0` disabling the timeout.

`-f`
Force option. Use this option to override the block on configuring a memory bank marked as disabled at boot in the non-volatile `disabled-memory-list` variable. See *Platform Notes:Sun Enterprise 6x00/5x00/4x00/3x00 Systems*

`-l`
List option. This option is supported as described in `cfgadm`(1M).

The type field is always *memory*.

The *info* field has the following information for empty banks:

```
slot# empty
```
The slot# indicates the system slot into which the CPU/Memory board is inserted. For example, if this were slot11 the attachment point for use with cfgadm to manipulate the associated board would be sysctrl0:slot11. The info field has the following information for connected banks:

```
slot# sizeMb|sizeGb [(sizeMb|sizeGb used)] base 0x###
      [interleaved #-way] [disabled at boot] [permanent]
```
The size of the bank is given in Mb or Gb as appropriate. If the memory is less than completely used, the used size is reported. The physical base address is given in hexadecimal. If the memory bank is interleaved with some other bank, the interleave factor is reported. If the memory on the board is disabled at boot using the non-volatile `disabled-memory-list` variable, this is reported. If the bank has memory that cannot be removed this is reported as permanent.

`-o` disable-at-boot | enable-at-boot
These options allow the state of the non-volatile disabled-memory-list variable to be modified. These options can be used in conjunction with the issuing of a `-c` option or with the explicit or implied listing command, -l, if no command is required. Use of `-o enable-at-boot` with the `configure` command to override the block on configuring memory on a board in the disabled memory list.

`-o` extended | normal | quick
Use with the `-t` option to specify test level.

The `normal` test level ensures that each memory cell stores both a `0` and a `1`, and checks that all cells are separately addressable. The `quick` test level only does the `0`s and `1`s test, and typically misses address line problems. The `extended` test uses patterns to test for adjacent cell interference problems. The default test level is `normal`. See `-t` option.

`-o` max_errors=#
Use with the `-t` option to specify the maximum number of allowed errors. If not specified, a default of `32` is assumed.

`-o` timeout=#
Use with the `unconfigure` command to set the self-cancelling timeout. The default value is `60` and the unit is seconds. A value of `0` means no timeout.

`-t`
Test an unconfigured bank of memory. Specify the test level using the `-o quick | normal | extended` option.

`cfgadm` exits with a `0` (success) if the *test* was able to run on the memory bank. The result of the test is available in the condition for the attachment point.

`-v`
Verbose option. Use this option in combination with the `-t` option to display detailed progress and results of tests.

`-x` relocate-test
For all pages of memory in use on the specified memory bank, a relocation operation as used in the unconfigure command is attempted. The success of this operation does not guarantee that the bank can be unconfigured. Failure indicates that it probably cannot be unconfigured. This option is for test purposes only.

OPERANDS

The following operand is supported:

ac#:*bank*#
The attachment points for memory banks are published by instances of the address controller (ac) driver (*ac*#). One instance of the ac driver is created for each system board, but only those instances associated with CPU/Memory boards publish the two bank attachment points, bank0 and bank1.

This form conforms to the logical `ap_id` specification given in `cfgadm`(1M). The corresponding physical `ap_id`s are listed in the `FILES` section.

The ac driver instance numbering has no relation to the slot number for the corresponding board. The full physical attachment point identifier has the slot number incorporated into it as twice the slot number in hexadecimal directly following the `fhc@` part.

FILES

/devices/fhc@*,f8800000/ac@0,1000000:bank?
attachment points

/usr/platform/sun4u/lib/cfgadm/cfgadm_ac.so.1
hardware specific library file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWkvm.u

SEE ALSO

`cfgadm`(1M), `cfgadm_sysctrl`(1M), `config_admin`(3CFGADM), `attributes`(5)

Sun Enterprise 6x00, 5x00, 4x00 and 3x00 Systems Dynamic Reconfiguration User's Guide

Platform Notes:Sun Enterprise 6x00/5x00/4x00/3x00 Systems

NOTES

Refer to the *Sun Enterprise 6x00, 5x00, 4x00 and 3x00 Systems Dynamic Reconfiguration User's Guide* for additional details regarding dynamic reconfiguration of EXX00 system CPU/Memory boards.

NAME

cfgadm_pci – PCI Hotplug hardware specific commands for cfgadm

SYNOPSIS

/usr/sbin/cfgadm [-f] [-y | -n] [-v] [-o *hardware_options*] -c *function ap_id* [*ap_id*]

/usr/sbin/cfgadm [-f] [-y | -n] [-v] [-o *hardware_options*] -x *hardware_function ap_id* [*ap_id*]

/usr/sbin/cfgadm [-v] [-s *listing_options*] [-o *hardware_options*] [-l [*ap_id* | *ap_type*]]

/usr/sbin/cfgadm [-v] [-o *harware_options*] -t *ap_id* [*ap_id*]

/usr/sbin/cfgadm [-v] [-o *hardware_function*] -h [*ap_id*| *ap_type*]

DESCRIPTION

The PCI hardware specific library `/usr/lib/cfgadm/pci.so.1` provides the support for hot plugging `pci` adapter cards into `pci` hot pluggable slots in a system that is hot plug capable, through `cfgadm`(1M). See `cfgadm`(1M).

For PCI Hot Plug, each hot plug slot on a specific PCI bus is represented by an attachment point of that specific PCI bus.

An attachment point consist of two parts: a receptacle and an occupant. The `receptacle` under PCI hot plug is usually referred to as the physical hot pluggable slot; and the `occupant` is usually referred to as the PCI adapter card that plugs into the slot.

Attachment points are named through `ap_id`s. There are two types of `ap_id`s: logical and physical. The physical `ap_id` is based on the physical pathname, that is, `/devices/pci@1/hpc0_slot3`, whereas the logical ap_id is a shorter, and more user-friendly name. For PCI hot pluggable slots, the logical `ap_id` is usually the corresponding hot plug controller driver name plus the logical slot number, that is, `pci0:hpc0slot1`; `pci` nexus driver, with hot plug controller driver named `hpc` and slot number `1`. The `ap_type` for Hot plug PCI is `pci`.

Note that the `ap_type` is not the same as the information in the `Type` field.

See the *System Administration Guide, Volume I* for a detailed description of the hot plug procedure.

OPTIONS

The following options are supported:

`-c` *function*

The following *function*s are supported for PCI hot pluggable slots:

configure	Configure the PCI device in the slot to be used by Solaris.
connect	Connect the slot to PCI bus.
disconnect	Disconnect the slot from the PCI bus.
insert	Not supported.
remove	Not supported.

unconfigure Logically remove the PCI device's resources from the system.

`-f`
Not supported.

`-h` *ap_id* | *ap_type*
Print out PCI hot plug specific help message.

`-l` *list*
List the values of PCI Hot Plug slots.

`-o` *hardware_options*
No hardware specific options are currently defined.

`-s` *listing_options*
Same as the generic `cfgadm`(1M)..

`-t` *ap_id*
This command is only supported on platforms which support testing capability on the slot.

`-v`
Execute in verbose mode.

When `-v` is used with `-l` option the `cfgadm` command outputs information about the attachment point. For PCI Hot Plug, the `Information` field will be the slot's system label. This string will be obtained from the `slot-name` property of the slot's bus node. The information in the Type field is printed with or without the `-v` option. The occupant `Type` field will describe the contents of the slot. There are 2 possible values:

NULL	The slot is empty. NULL should be unknown. There could be a card in the slot that is simply not configured.
subclass,*board*	The card in the slot is either a single- function or multi-function device. *subclass* is a string representing the subclass code of the device, for example, SCSI, `ethernet`, `pci-isa`, and so forth. If the card is a multi-functional device, `MULT` will get printed instead. *board* is a string representing the board type of the device, for example, hp for PCI Hot Plug adapter, hs for Hot Swap Board, nhs for Non—Hot Swap cPCI Board, bhs for Basic Hot Swap cPCI Board, fhs for Full Hot Swap cPCI Board. Most `pci` cards with more than one device on them are not actually multi-function devices, but are implimented as a pci bridge with arbitraty devices behind it. In that case, the subclass displayed will be

that of the `pci` bridge.

`-x` *hardware_function*
Perform hardware specific function. These hardware specific functions should not normally change the state of a receptacle or occupant.

The following *hardware_functions* are supported:

enable_slot | disable_slot
Change the state of the slot and preserve the state of slot across reboot. Preservation of state across reboot is only supported on select platforms.

`enable_slot` enables the addition of hardware to this slot for hot plugging and at boot time.

`disable_slot` disables the addition of hardware to this slot for hot plugging and at boot time. When a slot is disabled its condition is shown as unusable.

enable_autoconfig | disable_autoconfig
Change the ability to autoconfigure the occupant of the slot. Only platforms that support auto configuration support this feature.

`enable_autoconfig` enables the ability to autoconfigure the slot.

`diable_autoconfig` disables the ability to autoconfigure the slot.

led=[*led_sub_arg*],mode=[*mode_sub_arg*]
Without sub-arguments, print a list of the current LED settings. With sub-arguments, set the mode of a specific LED for a slot.

Specify *led_sub_arg* as `fault`, `power`, `att`, or `active`.

Specify *mode_sub_arg* as `on`, `off` or `blink`.

Changing the state of the LED does not change the state of the receptacle or occupant. Normally, the LEDs are controlled by the hot plug controller, no user intervention is necessary. Use this command for testing purposes.

Caution: Changing the state of the LED can misrepresent the state of occupant or receptacle.

The following command prints the values of LEDs:

```
example#  cfgadm -x led pci0:hpc0_slot1Ap_Id             Led
pci0:hpc0_slot1   power=on,fault=off,active=off,attn=off
```

The following command turns on the Fault LED:

```
example# cfgadm -x led=fault,mode=on pci0:hpc0_slot1
```

The following command turns off the Power LED:

```
example# cfgadm -x led=power,mode=off pci0:hpc0_slot0
```

The following command sets the *active* LED to blink to indicate the location of the slot:

```
example# cfgadm -x led=active,mode=on pci0:hpc0_slot3
```

EXAMPLES

EXAMPLE 1 Printing out the Value of Each Slot

The following command prints out the values of each slot:

```
example# cfgadm -l
Ap_Id            Type         Receptacle   Occupant       Condition
pci1:hpc0_slot0  unknown      empty        unconfigured   unknown
pci1:hpc0_slot1  unknown      empty        unconfigured   unknown
pci1:hpc0_slot2  unknown      empty        unconfigured   unknown
pci1:hpc0_slot3  HP/SCSI      connected    configured     ok
pci1:hpc0_slot4  unknown      empty        unconfigured   unknown
```

EXAMPLE 2 Replacing a Card

The following command lists all DR-capable attachment points:

```
example# cfgadm
Ap_Id                   Type         Receptacle   Occupant     Condition
c0                      scsi-bus     connected   configured   unknown
pci_pci0:cpci_slot1     stpcipci/fhs connected   configured   ok
pci_pci0:cpci_slot      unknown      empty       unconfigured unknown
pci_pci0:cpci_slot4     stpcipci/fhs connected   configured   ok
pci_pci0:cpci_slot5     unknown      empty       unconfigured unknown
```

The following command unconfigures and electrically disconnects the card:

```
example# cfgadm -c disconnect pci_pci0:cpci_slot4
```

The change can be verified by entering the following command:

```
example# cfgadm
Ap_Id                   Type         Receptacle   Occupant     Condition
c0                      scsi-bus     connected    configured   unknown
pci_pci0:cpci_slot1     stpcipci/fhs connected    configured   ok
pci_pci0:cpci_slot      unknown      empty        unconfigured unknown
pci_pci0:cpci_slot4     unknown      disconnected unconfigured unknown
pci_pci0:cpci_slot     unknown       empty        unconfigured unknown
```

Now the card can be swapped. The following command electrically connects and configures the card:

```
example# cfgadm -c configure pci_pci0:cpci_slot4
```

The change can be verifed by entering the following command:

```
# cfgadm
Ap_Id                   Type         Receptacle   Occupant     Condition
c0                      scsi-bus     connected    configured   unknown
```

EXAMPLE 2 Replacing a Card *(Continued)*

```
pci_pci0:cpci_slot1     stpcipci/fhs connected    configured   ok
pci_pci0:cpci_slot2     unknown      empty        unconfigured unknown
pci_pci0:cpci_slot4     stpcipci/fhs connected    configured   ok
pci_pci0:cpci_slot5     unknown      empty        unconfigured unknown
```

FILES

/usr/lib/cfgadm/libpci.so.1	Hardware specific library for PCI hot plugging.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWkvm.u

SEE ALSO

`cfgadm`(1M), `config_admin`(3CFGADM), `libcfgadm`(3LIB)`attributes`(5)

System Administration Guide: Basic Administration

NAME

cfgadm_sbd – `cfgadm` commands for system board administration

SYNOPSIS

`cfgadm` `-l [-a] [-o parsable]` *ap_id...*

`cfgadm` `-c` *function* `[-f] [-y | -n] [-o unassign | nopoweroff] [-v]` *ap_id...*

`cfgadm` `-t [-v]` *ap_id...*

`cfgadm` `-x [-f] [-v]` *function ap_id...*

DESCRIPTION

The `cfgadm_sbd` plugin provides dynamic reconfiguration functionality for connecting, configuring, unconfiguring, and disconnecting class `sbd` system boards. It also enables you to connect or disconnect a system board from a running system without having to reboot the system.

The `cfgadm` command resides in `/usr/sbin`. See `cfgadm`(1M). The `cfgadm_sbd` plugin resides `/usr/platform/sun4u/lib/cfgadm`.

Each board slot appears as a single attachment point in the device tree. Each component appears as a dynamic attachment point. You can view the type, state, and condition of each component, and the states and condition of each board slot by using the `-a` option.

The `cfgadm` options perform differently depending on the platform. Additionally, the form of the attachment points is different depending on the platform. See the `Platform Notes` section for more information.

Component Conditions

The following are the names and descriptions of the component conditions:

failed	The component failed testing.
ok	The component is operational.
unknown	The component has not been tested.

Component States

The following is the name and description of the receptacle state for components:

connected	The component is connected to the board slot.

The following are the names and descriptions of the occupant states for components:

configured	The component is available for use by the Solaris operating environment.
unconfigured	The component is not available for use by the Solaris operating environment.

Board Conditions

The following are the names and descriptions of the board conditions.

failed	The board failed testing.
ok	The board is operational.
unknown	The board has not been tested.

unusable	The board slot is unusable.

Board States

Inserting a board changes the receptacle state from empty to disconnected. Removing a board changes the receptacle state from disconnected to empty.

Caution: Removing a board that is in the connected state or that is powered on and in the disconnected state crashes the operating system and can result in permanent damage to the system.

The following are the names and descriptions of the receptacle states for boards:

connected	The board is powered on and connected to the system bus. You can view the components on a board only after it is in the connected state.
disconnected	The board is disconnected from the system bus. A board can be in the disconnected state without being powered off. However, a board must be powered off and in the disconnected state before you remove it from the slot.
empty	A board is not present.

The occupant state of a disconnected board is always unconfigured. The following table contains the names and descriptions of the occupant states for boards:

configured	At least one component on the board is configured.
unconfigured	All of the components on the board are unconfigured.

Dynamic System Domains

Platforms based on dynamic system domains (DSDs, referred to as domains in this document) divide the slots in the chassis into electrically isolated hardware partitions (that is, DSDs). Platforms that are not based on DSDs assign all slots to the system permanently.

A slot can be empty or populated, and it can be assigned or available to any number of domains. The number of slots available to a given domain is controlled by an available component list (ACL) that is maintained on the system controller. The ACL is not the access control list provided by the Solaris operating environment.

A slot is visible to a domain only if the slot is in the domain's ACL and if it is not assigned to another domain. An unassigned slot is visible to all domains that have the slot in their ACL. After a slot has been assigned to a domain, the slot is no longer visible to any other domain.

A slot that is visible to a domain, but not assigned, must first be assigned to the domain before any other state changing commands are applied. The assign can be done explicitly using `-x assign` or implicitly as part of a connect. A slot must be unassigned from a domain before it can be used by another domain. The unassign is always explicit, either directly using `-x unassign` or as an option to disconnect using `-o unassign`.

State Change Functions

Functions that change the state of a board slot or a component on the board can be issued concurrently against any attachment point. Only one state changing operation is permitted at a given time. A `Y` in the Busy field in the state changing information indicates an operation is in progress.

The following list contains the functions that change the state:

- configure
- unconfigure
- connect
- disconnect

Availability Change Functions

Commands that change the availability of a board can be issued concurrently against any attachment point. Only one availability change operation is permitted at a given time. These functions also change the information string in the `cfgadm -l` output. A `Y` in the Busy field indicates that an operation is in progress.

The following list contains the functions that change the availability:

- `assign`
- `unassign`

Condition Change Functions

Functions that change the condition of a board slot or a component on the board can be issued concurrently against any attachment point. Only one condition change operation is permitted at a given time. These functions also change the information string in the `cfgadm -l` output. A `Y` in the Busy field indicates an operation is in progress.

The following list contains the functions that change the condition:

- `poweron`
- `poweroff`
- `test`

Unconfigure Process

This section contains a description of the unconfigure process, and illustrates the states of source and target boards at different stages during the process of moving permanent memory.

In the following code examples, the permanent memory on board 0 must be moved to another board in the domain. Thus, board 0 is the source, and board 1 is the target.

A status change operation cannot be initiated on a board while it is marked as busy. For brevity, the CPU information has been removed from the code examples.

The process is started with the following command:

```
# cfgadm -c unconfigure -y SB0::memory &
```

First, the memory on board 1 in the same address range as the permanent memory on board 0 must be deleted. During this phase, the source board, the target board, and the memory attachment points are marked as busy. You can display the status with the following command:

```
# cfgadm -a -s cols=ap_id:type:r_state:o_state:busy SB0 SB1

Ap_Id          Type      Receptacle     Occupant       Busy
SB0            CPU       connected      configured     y
SB0::memory    memory    connected      configured     y
SB1            CPU       connected      configured     y
SB1::memory    memory    connected      configured     y
```

After the memory has been deleted on board 1, it is marked as unconfigured. The memory on board 0 remains configured, but it is still marked as busy, as in the following example.

```
Ap_Id          Type      Receptacle     Occupant       Busy
SB0            CPU       connected      configured     y
SB0::memory    memory    connected      configured     y
SB1            CPU       connected      configured     y
SB1::memory    memory    connected      unconfigured   n
```

The memory from board 0 is then copied to board 1. After it has been copied, the occupant state for the memory is switched. The memory on board 0 becomes unconfigured, and the memory on board 1 becomes configured. At this point in the process, only board 0 remains busy, as in the following example.

```
Ap_Id          Type      Receptacle     Occupant       Busy
SB0            CPU       connected      configured     y
SB0::memory    memory    connected      unconfigured   n
SB1            CPU       connected      configured     n
SB1::memory    memory    connected      configured     n
```

After the entire process has been completed, the memory on board 0 remains unconfigured, and the attachment points are not busy, as in the following example.

```
Ap_Id          Type      Receptacle     Occupant       Busy
SB0            CPU       connected      configured     n
SB0::memory    memory    connected      unconfigured   n
SB1            CPU       connected      configured     n
SB1::memory    memory    connected      configured     n
```

The permanent memory has been moved, and the memory on board 0 has been unconfigured. At this point, you can initiate a new state changing operation on either board.

Platform-Specific Options

You can specify platform-specific options that follow the options interpreted by the system board plugin. All platform-specific options must be preceded by the `platform` keyword. The following example contains the general format of a command with platform-specific options:

command `-o` *sbd_options*`,platform=`*platform_options*

OPTIONS

This man page does not include the `-v`, `-a`, `-s`, or `-h` options for the `cfgadm` command. See `cfgadm`(1M) for descriptions of those options. The following options are supported by the `cfgadm_sbd` plugin:

`-c` *function* Performs a state change function. You can use the following functions:

unconfigure
: Changes the occupant state to unconfigured. This function applies to system board slots and to all of the components on the system board.

  The `unconfigure` function removes the CPUs from the CPU list and deletes the physical memory from the system memory pool. If any device is still in use, the `cfgadm` command fails and reports the failure to the user. You can retry the command as soon as the device is no longer busy. If a CPU is in use, you must ensure that it is off line before you proceed. See `pbind`(1M), `psradm`(1M) and `psrinfo`(1M).

  The `unconfigure` function moves the physical memory to another system board before it deletes the memory from the board you want to unconfigure. Depending of the type of memory being moved, the command fails if it cannot find enough memory on another board or if it cannot find an appropriate physical memory range.

  For permanent memory, the operating system must be suspended (that is, quiesced) while the memory is moved and the memory controllers are reprogrammed. If the operating system must be suspended, you will be prompted to proceed with the operation. You can use the `-y` or `-n` options to always answer yes or no respectively.

  Moving memory can take several minutes to complete, depending on the amount of memory and the system load. You can monitor the progress of the operation by issuing a status command against the memory attachment point. You can also interrupt the memory operation by stopping the `cfgadm` command. The deleted memory is returned to the system memory pool.

disconnect
Changes the receptacle state to disconnected. This function applies only to system board slots.

If the occupant state is configured, the `disconnect` function attempts to unconfigure the occupant. It then powers off the system board. At this point, the board can be removed from the slot.

This function leaves the board in the assigned state on platforms that support dynamic system domains.

If you specify `-o nopoweroff`, the `disconnect` function leaves the board powered on. If you specify `-o unassign`, the `disconnect` function unassigns the board from the domain.

If you unassign a board from a domain, you can assign it to another domain. However, if it is assigned to another domain, it is not available to the domain from which is was unassigned.

configure
Changes the occupant state to configured. This function applies to system board slots and to any components on the system board.

If the receptacle state is disconnected, the `configure` function attempts to connect the receptacle. It then walks the tree of devices that is created by the `connect` function, and attaches the devices if necessary. Running this function configures all of the components on the board, except those that have already been configured.

For CPUs, the `configure` function adds the CPUs to the CPU list. For memory, the `configure` function ensures that the memory is initialized then adds the memory to the system memory pool. The CPUs and the memory are ready for use after the `configure` function has been completed successfully.

For I/O devices, you must use the `mount` and the `ifconfig` commands before the devices can be used. See `ifconfig`(1M) and `mount`(1M).

connect
Changes the receptacle state to connected. This function applies only to system board slots.

If the board slot is not assigned to the domain, the `connect` function attempts to assign the slot to the domain. Next, it powers on and tests the board, then it connects the board electronically to the system bus and probes the components.

After the `connect` function is completed successfully, you can use the `-a` option to view the status of the components on the board. The `connect` function leaves all of the components in the unconfigured state.

The assignment step applies only to platforms that support dynamic system domains.

`-f` Overrides software state changing constraints.

The `-f` option never overrides fundamental safety and availability constraints of the hardware and operating system.

`-l` Lists the state and condition of attachment points specified in the format controlled by the `-s`, `-v`, and `-a` options as specified in `cfgadm`(1M). The `cfgadm_sbd` plugin provides specific information in the info field as described below. The format of this information may be altered by the `-o parsable` option.

The parsable `info` field is composed of the following:

cpu
The `cpu` type displays the following information:

cpuid=#	Where # is a number, representing the ID of the CPU.
speed=#	Where # is a number, representing the speed of the CPU in MHz.
ecache=#	Where # is a number, representing the size of the ecache in MBytes.

memory
The `memory` type displays the following information, as appropriate:

address=#	Where # is a number, representing the base physical address.
size=#	Where # is a number, representing the size of the memory in KBytes.
permanent=#	Where # is a number, representing the size of permanent memory in KBytes.
unconfigurable	An operating system setting that prevents

the memory from being unconfigured.

inter-board-interleave	The board is participating in interleaving with other boards.
source=*ap_id*	Represents the source attachment point.
target=*ap_id*	Represents the target attachment point.
deleted=#	Where # is a number, representing the amount of memory that has already been deleted in KBytes.
remaining=#	Where # is a number, representing the amount of memory to be deleted in KBytes.

io

The `io` type displays the following information:

device=*path*	Represents the physical path to the I/O component.
referenced	The I/O component is referenced.

board

The `board` type displays the following boolean names. If they are not present, then the opposite applies.

assigned	The board is assigned to the domain.
powered-on	The board is powered on.

The same items appear in the `info` field in a more readable format if the `-o parsable` option is not specified.

`-o` parsable

Returns the information in the `info` field as a boolean *name* or a set of `name=value` pairs, separated by a space character.

The `-o parsable` option can be used in conjunction with the `-s` option. See the `cfgadm`(1M) man page for more information about the `-s` option.

`-t`

Tests the board.

Before a board can be connected, it must pass the appropriate level of testing.

Use of this option always attempts to test the board, even if it has already passed the appropriate level of testing. Testing is also performed when a `-c connect` state change function is issued, in which case the test step can be skipped if the board already shows an appropriate level of testing. Thus the `-t` option can be used to explicitly request that the board be tested.

`-x` *function* — Performs an sbd-class function. You can use the following functions:

assign
: Assigns a board to a domain.

  The receptacle state must be disconnected or empty. The board must also be listed in the domain available component list. See Dynamic System Domains.

unassign
: Unassigns a board from a domain.

  The receptacle state must be disconnected or empty. The board must also be listed in the domain available component list. See Dynamic System Domains.

poweron
: Powers the system board on.

  The receptacle state must be disconnected.

poweroff
: Powers the system board off.

  The receptacle state must be disconnected.

OPERANDS

The following operands are supported:

Receptacle *ap_id* — For the Sun Fire 15K, the receptacle attachment point ID takes the form `SB`*X* or `IO`*X*, where *X* equals the slot number.

The exact format depends on the platform and typically corresponds to the physical labelling on the machine. See the platform specific information in the `NOTES` section.

Component *ap_id* — The component attachment point ID takes the form *component_typeX*, where *component_type* equals one of the component types described in “Component Types”

and *X* equals the component number. The component number is a board-relative unit number.

The above convention does not apply to memory compontents. Any DR action on a memory attachment point affects all of the memory on the system board.

EXAMPLES The following examples show user input and system output on a Sun Fire 15K system. User input — specifically references to attachment points — and system output may differ on other Sun Fire systems, such as the 6800 and 4810 models. Refer to the Platform Notes for specific information about using the `cfgadm_sbd` plugin on non-15000 models.

EXAMPLE 1 Listing All of the System Board

```
# cfgadm -a -s "select=class(sbd)"

Ap_Id          Type      Receptacle     Occupant       Condition
SB0            CPU       connected      configured     ok
SB0::cpu0      cpu       connected      configured     ok
SB0::memory    memory    connected      configured     ok
IO1            HPCI      connected      configured     ok
IO1::pci0      io        connected      configured     ok
IO1::pci1      io        connected      configured     ok
SB2            CPU       disconnected   unconfigured   failed
SB3            CPU       disconnected   unconfigured   unusable
SB4            unknown   empty          unconfigured   unknown
```

This example demonstrates the mapping of the following conditions:

- The board in Slot 2 failed testing.
- Slot 3 is unusable; thus, you cannot hot plug a board into that slot.

EXAMPLE 2 Listing All of the CPUs on the System Board

```
# cfgadm -a -s "select=class(sbd):type(cpu)"

Ap_Id          Type      Receptacle     Occupant       Condition
SB0::cpu0      cpu       connected      configured     ok
SB0::cpu1      cpu       connected      configured     ok
SB0::cpu2      cpu       connected      configured     ok
SB0::cpu3      cpu       connected      configured     ok
```

EXAMPLE 3 Displaying the CPU Information Field

```
# cfgadm -l -s noheadings,cols=info SB0::cpu0

cpuid 16, speed 400 MHz, ecache 8 Mbytes
```

EXAMPLE 4 Displaying the CPU Information Field in Parsable Format

```
# cfgadm -l -s noheadings,cols=info -o parsable SB0::cpu0
```

EXAMPLE 4 Displaying the CPU Information Field in Parsable Format *(Continued)*

```
cpuid=16 speed=400 ecache=8
```

EXAMPLE 5 Displaying the Devices on an I/O Board

```
# cfgadm -a -s noheadings,cols=ap_id:info -o parsable IO1

IO1       powered-on assigned
IO1::pci0 device=/devices/saf@0/pci@0,2000 referenced
IO1::pci1 device=/devices/saf@0/pci@1,2000 referenced
```

EXAMPLE 6 Monitoring an Unconfigure Operation

In the following example, the memory sizes are displayed in Kbytes.

```
# cfgadm -c unconfigure -y SB0::memory &
# cfgadm -l -s noheadings,cols=info -o parsable SB0::memory SB1::memory

address=0x0 size=2097152 permanent=752592 target=SB1::memory
     deleted=1273680 remaining=823472
address=0x1000000 size=2097152 source=SB0::memory
```

EXAMPLE 7 Assigning a Slot to a Domain

```
# cfgadm -x assign SB2
```

EXAMPLE 8 Unassigning a Slot from a Domain

```
# cfgadm -x unassign SB3
```

ATTRIBUTES

See `attributes`(5) for a description of the following attribute:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	`SUNWkvm.u`

SEE ALSO

`cfgadm`(1M), `devfsadm`(1M), `ifconfig`(1M), `mount`(1M), `pbind`(1M), `psradm`(1M), `psrinfo`(1M), `config_admin`(3CFGADM), `attributes`(5)

NOTES

This section contains information on how to monitor the progress of a memory delete operation. It also contains platform specific information.

Memory Delete Monitoring

The following shell script can be used to monitor the progress of a memory delete operation.

```
# cfgadm -c unconfigure -y SB0::memory &
# watch_memdel SB0
```

```
#!/bin/sh
# This is the watch_memdel script.

if [ -z "$1" ]; then
        printf "usage:  %s board_id\n" `basename $0`
        exit 1
fi

board_id=$1

cfgadm_info='cfgadm -s noheadings,cols=info -o parsable'

eval `$cfgadm_info $board_id::memory`

if [ -z "$remaining" ]; then
        echo no memory delete in progress involving $board_id
        exit 0
fi

echo deleting target $target

while true
do
        eval `$cfgadm_info $board_id::memory`

        if [ -n "$remaining" -a "$remaining" -ne 0 ]
        then
                echo $deleted KBytes deleted, $remaining KBytes remaining
                remaining=
        else
                echo memory delete is done
                exit 0
        fi
        sleep 1
done
exit 0
```

Sun Enterprise 10000 Platform Notes

The following syntax is used to refer to Platform Notes attachment points on the Sun Enterprise 10000 system:

board`::`*component*

where *board* refers to the system board; and *component* refers to the individual component. System boards can range from `SB0` (zero) to `SB15`. A maximum of sixteen system boards are available.

The DR 3.0 model running on a Sun Enterprise 10000 domain supports a limited subset of the functionality provided by the `cfgadm_sbd` plugin. The only supported operation is to view the status of attachment points in the domain. This corresponds to the `-l` option and all of its associated options.

Attempting to perform any other operation from the domain will result in an error that states that the operation is not supported. All operations to add or remove a system board must be initiated from the System Service Processor.

Sun Fire 15K Platform Notes

The following syntax is used to refer to attachment points on the Sun Fire 15K system:

board::*component*

where *board* refers to the system board or I/O board; and *component* refers to the individual component.

Depending on the system's configuration, system boards can range from `SB0` (zero) through `SB17`, and I/O boards can range from `IO0` (IO zero) through `IO17`. (A maximum of eighteen system and I/O boards are available).

The `-t` and `-x` options behave differently on the Sun Fire 15K platform. The following list describes their behavior:

`-t` — The system controller uses a CPU to test system boards by running `LPOST`, sequenced by the `hpost` command. To test I/O boards, the driver starts the testing in response to the `-t` option, and the test runs automatically without user intervention. The driver unconfigures a CPU and a stretch of contiguous physical memory. Then, it sends a command to the system controller to test the board. The system controller uses the CPU and memory to test the I/O board from inside of a transaction/error cage.

`-x assign | unassign` — In the Sun Fire 15K system administration model, the platform administrator controls the platform hardware through the use of an available component list for each domain. This information is maintained on the system controller. Only the platform administrator can modify the available component list for a domain.

The domain administrator is only allowed to assign or unassign a board if it is in the available component list for that domain. The platform administrator does not have this restriction, and can assign or unassign a board even if it is not in the available component list for a domain.

Sun Fire 15K Component Types

The following are the names and descriptions of the component types:

`cpu`	CPU
`io`	I/O device
`memory`	Memory

Note: An operation on a memory component affects all of the memory components on the board.

Sun Fire 6800, 4810, 4800 and 3800 Platform Notes

References to attachment points are slightly different on Sun Fire 6800, 4810, 4800, and 3800 systems than on the Sun Fire 15K system. The following syntax is used to refer to attachment points on Sun Fire systems other than the Sun Fire 15K:

`N#.`*board*`::`*component*

where **`N#`** refers to the node; *board* refers to the system board or I/O board; and *component* refers to the individual component.

Depending on the system's configuration, system boards can range from `SB0` through `SB5`, and I/O boards can range from `IB6` through `IB9`. (A maximum of six system and four I/O boards are available).

Sun Fire 6800, 4810, 4800 and 3800 Component Types

The following are the names and descriptions of the component types:

`cpu`	CPU
`pci`	I/O device
`memory`	Memory

Note: An operation on a memory component affects all of the memory components on the board.

NAME

cfgadm_scsi – SCSI hardware specific commands for cfgadm

SYNOPSIS

/usr/sbin/cfgadm [-f] [-y | -n] [-v] [-o *hardware_option*] -c *function ap_id...*

/usr/sbin/cfgadm [-f] [-y | -n] [-v] [-o *hardware_option*] -x *hardware_function ap_id...*

/usr/sbin/cfgadm [-v] [-a] [-s *listing_option*] [-o *hardware_option*] [-l [*ap_id* | *ap_type ...*]]

/usr/sbin/cfgadm [-v] [-o *hardware_option*] -t *ap_id...*

/usr/sbin/cfgadm [-v] [-o *hardware_option*] -h [*ap_id...*]

DESCRIPTION

The SCSI hardware specific library `/usr/lib/cfgadm/scsi.so.1` provides the functionality for SCSI hot-plugging through the `cfgadm`(1M) command. `cfgadm` operates on attachment points, which are locations in the system where hardware resources can be dynamically reconfigured. Refer to`cfgadm`(1M) for information regarding attachment points.

For SCSI hot-plugging, each SCSI controller is represented by an attachment point in the device tree. In addition, each SCSI device is represented by a dynamic attachment point. Attachment points are named through `ap_ids`. Two types of `ap_ids` are defined: logical and physical. The physical `ap_id` is based on the physical pathname, whereas the logical `ap_id` is a shorter more user-friendly name. For SCSI controllers, the logical `ap_id` is usually the corresponding disk controller number. For example, a typical logical `ap_id` would be `c0`.

SCSI devices are named relative to the controller `ap_id`. Thus if a disk device is attached to controller `c0`, its `ap_id` can be:

```
c0::dsk/c0t0d0
```

where `dsk/c0t0d0` identifies the specific device. In general, the device identifier is derived from the corresponding logical link for the device in `/dev`. For example, a SCSI tape drive logical `ap_id` could be `c0::rmt/0`. Here `c0` is the logical `ap_id` for the `SCSI` controller and `rmt/0` is derived from the logical link for the tape drive in `/dev/rmt`. If an identifier can not be derived from the link in `/dev`, a unique identifier will be assigned to it. For example, if the tape device has no link in `/dev`, it can be assigned an `ap_id` of the form `c0::st3` where `st3` is a unique internally generated identifier.

A simple listing of attachment points in the system will include attachment points at SCSI controllers but not SCSI devices. Use the `-a` flag to the list option (`-l`) to list SCSI devices as well. For example:

```
# cfgadm -l
  Ap_Id             Type        Receptacle     Occupant     Condition
  c0                scsi-bus    connected      configured   unknown
  sysctrl0:slot0    cpu/mem     connected      configured   ok
  sysctrl0:slot1    sbus-upa    connected      configured   ok
```

To list SCSI devices in addition to SCSI controllers:

```
# cfgadm -al

Ap_Id                Type         Receptacle   Occupant     Condition
c0                   scsi-bus     connected    configured   unknown
c0::dsk/c0t14d0      disk         connected    configured   unknown
c0::dsk/c0t11d0      disk         connected    configured   unknown
c0::dsk/c0t8d0       disk         connected    configured   unknown
c0::dsk/c0t0d0       disk         connected    configured   unknown
c0::rmt/0            tape         connected    configured   unknown
sysctrl0:slot0       cpu/mem      connected    configured   ok
sysctrl0:slot1       sbus-upa     connected    configured   ok
```

Refer to `cfgadm`(1M) for more information regarding listing attachment points. The receptacle and occupant state for attachment points at the SCSI controller have the following meanings:

empty
  not applicable

disconnected
  bus quiesced (I/O activity on bus is suspended)

connected
  bus active

configured
  one or more devices on the bus is configured

unconfigured
  no device on the bus is configured

The corresponding states for individual SCSI devices are:

empty
  not applicable

disconnected
  bus to which the device is attached is quiesced

connected
  bus to which device is attached is active

configured
  device is configured

unconfigured
  device is not configured

OPTIONS

`cfgadm` defines several types of operations besides listing (`-l`).These operations include testing, (`-t`), invoking configuration state changes, (`-c`), invoking hardware specific functions (`-x`), and obtaining configuration administration help messages (`-h`).

`-c` *function*

The following generic commands are defined for the SCSI hardware specific library:

For SCSI controller attachment points, the following configuration state change operations are supported:

connect	Unquiesce the SCSI bus.
disconnect	Quiesce the bus (suspend I/O activity on bus). Incorrect use of this command can cause the system to hang. See `NOTES`.
configure	Configure new devices on SCSI bus.
unconfigure	Unconfigure all devices connected to bus.

The following generic commands are defined for SCSI devices:

configure	configure a specific device
unconfigure	unconfigure a specific device

`-f`

When used with the `disconnect` command, forces a quiesce of the SCSI bus, if supported by hardware.

Incorrect use of this command can cause the system to hang. See `NOTES`.

`-h` *ap_id*

SCSI specific help can be obtained by using the help option with any SCSI attachment point.

`-o` *hardware_option*

No hardware specific options are currently defined.

`-s` *listing_option*

Attachment points of class `scsi` can be listed by using the `select` sub-option. Refer to the `cfgadm`(1M) man page for additional information.

`-t` *ap_id*

No test commands are available at present.

`-x` *hardware_function*

Some of the following commands can only be used with SCSI controllers and some only with SCSI devices.

In the following, *controller_ap_id* refers to an `ap_id` for a SCSI controller, for example, `c0`. *device_ap_id* refers to an `ap_id` for a SCSI device, for example: `c0::dsk/c0dt3d0`.

The following hardware specific functions are defined:

insert_device *controller_ap_id*
Add a new device to the SCSI controller, *controller_ap_id*.

This command is intended for interactive use only.

remove_device *device_ap_id*
Remove device *device_ap_id*.

This command is intended for interactive use only.

replace_device *device_ap_id*
Remove device *device_ap_id* and replace it with another device of the same kind.

This command is intended for interactive use only.

reset_device *device_ap_id*
Reset *device_ap_id*.

reset_bus *controller_ap_id*
Reset bus *controller_ap_id* without resetting any devices attached to the bus.

reset_all *controller_ap_id*
Reset bus *controller_ap_id* and all devices on the bus.

EXAMPLES

EXAMPLE 1 Configuring a Disk

The following command configures a disk attached to controller `c0`:

```
# cfgadm -c configure c0::dsk/c0t3d0
```

EXAMPLE 2 Unconfiguring a Disk

The following command unconfigures a disk attached to controller `c0`:

```
# cfgadm -c unconfigure c0::dsk/c0t3d0
```

EXAMPLE 3 Adding a New Device

The following command adds a new device to controller `c0`:

```
# cfgadm -x insert_device c0
```

The system responds with the following:

```
Adding device to SCSI HBA: /devices/sbus@1f,0/SUNW,fas@e,8800000
This operation will suspend activity on SCSI bus c0
Continue (yes/no)?
```

Enter:

```
y
```

EXAMPLE 3 Adding a New Device *(Continued)*

The system responds with the following:

```
SCSI bus quiesced successfully.
It is now safe to proceed with hotplug operation.
Enter y if operation is complete or n to abort (yes/no)?
```

Enter:

```
y
```

EXAMPLE 4 Replacing a Device

The following command replaces a device attached to controller `c0`:

```
# cfgadm -x replace_device c0::dsk/c0t3d0
```

The system responds with the following:

```
Replacing SCSI device: /devices/sbus@1f,0/SUNW,fas@e,8800000/sd@3,0
This operation will suspend activity on SCSI bus: c0
Continue (yes/no)?
```

Enter:

```
y
```

The system responds with the following:

```
SCSI bus quiesced successfully.
It is now safe to proceed with hotplug operation.
Enter y if operation is complete or n to abort (yes/no)?
```

Enter:

```
y
```

EXAMPLE 5 Encountering a Mounted File System While Unconfiguring a Disk

The following command illustrates encountering a mounted file system while unconfiguring a disk:

```
# cfgadm -c unconfigure c1::dsk/c1t0d0
```

The system responds with the following:

```
cfgadm: Component system is busy, try again: failed to offline:
/devices/pci@1f,4000/scsi@3,1/sd@1,0
     Resource              Information
------------------  -------------------------
/dev/dsk/c1t0d0s0   mounted filesystem "/mnt"
```

FILES

`/usr/lib/cfgadm/scsi.so.1`	hardware specific library for generic SCSI hot-plugging

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsl (32-bit)
	SUNWcslx (64-bit)

SEE ALSO

`cfgadm`(1M), `luxadm`(1M), `config_admin`(3CFGADM), `libcfgadm`(3LIB),`attributes`(5)

NOTES

The `disconnect` (quiesce) operation is not supported on controllers which control disks containing critical partitions such as `root` (`/`), `/usr`, `swap`, or `/var`. The disconnect operation should not be attempted on such controllers. Incorrect usage can result in a system hang and require a reboot.

Hotplugging operations are not supported by all SCSI controllers.

WARNINGS

The connectors on some SCSI devices do not confirm to SCSI hotplug specifications. Performing hotplug operations on such devices can cause damage to the hardware on the SCSI bus. Refer to your hardware manual for additional information.

NAME

cfgadm_sysctrl – EXX00 system board administration

SYNOPSIS

```
/usr/sbin/cfgadm -c function [-f] [-o disable-at-boot |
    enable-at-boot] [-n | -y ] sysctrl0:slot# …

/usr/sbin/cfgadm -x quiesce-test sysctrl0:slot#

/usr/sbin/cfgadm -x insert-test | remove-test sysctrl0:slot# …

/usr/sbin/cfgadm -x set-condition-test=# sysctrl0:slot# …

/usr/sbin/cfgadm [-l] -o disable-at-boot | enable-at-boot
    sysctrl0:slot# …
```

DESCRIPTION

The `sysctrl` hardware specific library `/usr/platform/sun4u/lib/cfgadm/sysctrl.so.1` provides dynamic reconfiguration functionality for configuring and disconnecting system boards on `E6X00`, `E5X00`, `E4X00`, and `E3X00` systems. You can insert both I/O and CPU boards into a slot on a running system that is configured for Solaris without rebooting. You can also disconnect and remove both types of boards from a running system without rebooting.

System slots appear as attachment points in the device tree, one attachment point for each actual slot in the system chassis. If a board is not in a slot, the receptacle state is `empty`. If a board is powered-off and ready to remove, the receptacle state is `disconnected`. If a board is powered-on and is connected to the system bus, the receptacle state is `connected`.

The occupant state is `unconfigured` when the receptacle state is `empty` or `disconnected`. The occupant state is either `unconfigured` or `configured` when the receptacle state is `connected`.

In the `configured` state the devices on a board are available for use by Solaris. In the `unconfigured` state, the devices on the board are not.

Inserting a board changes the receptacle state from `empty` to `disconnected`. Removing a board changes the receptacle state from `disconnected` to `empty`. Removing a board that is in the `connected` state crashes the operating system and can result in permanent damage to the system.

OPTIONS

Refer to `cfgadm`(1M) for a more complete description options.

The following options are supported:

`-c` *function*

Perform the state change function. Specify *function* as `connect`, `disconnect`, `configure` or `unconfigure`.

configure

Change the occupant state to `configure`.

If the receptacle state is `disconnected`, the `configure` function first attempts to connect the receptacle. The `configure` function walks the OBP device tree created as part of the `connect` function and creates the Solaris device tree nodes, attaching devices as required. For CPU/Memory boards, `configure` adds CPUs to the CPU list in the `powered-off` state. These are visible to the `psrinfo`(1M) and `psradm`(1M) commands. Two memory attachment points are published for CPU/memory boards. Use `mount`(1M) and`ifconfig`(1M) to use I/O devices on the new board. To use CPUs, use `psradm -n` to on-line the new processors. Use `cfgadm_ac`(1M) to test and configure the memory banks.

connect
Change the receptacle state to `connected`.

Changing the receptacle state requires that the system bus be frozen while the bus signals are connected and the board tested. The bus is frozen by running a `quiesce` operation which stops all process activity and suspends all drivers. Because the `quiesce` operation and the subsequent resume can be time consuming, and are not supported by all drivers, the `-x quiesce-test` is provided. While the system bus is frozen, the board being connected is tested by firmware. This operation takes a short time for I/O boards and a significant time for CPU/Memory boards due to CPU external cache testing. This does not provide memory testing. The user is prompted for confirmation before proceeding with the `quiesce`. Use the `-y` or `-n` option to override the prompt. The `connect` operation is refused if the board is marked as `disabled-at-boot`, unless either the force flag, `-f`, or the enable at boot flag, `-o enable-at-boot`, is given. See `-l`.

disconnect
Change the receptacle state to `disconnected`.

If the occupant state is `configure`, the `disconnect` function first attempts to unconfigure the occupant. The `disconnect` operation does not require a `quiesce` operation and operates quickly. The board is powered-off ready for removal.

unconfigure
Change the occupant state to `unconfigureed`.

Devices on the board are made invisible to Solaris during this process. The I/O devices on an I/O board are removed from the Solaris device tree. Any device that is still in use stops the unconfigure process and be reported as in use. The unconfigure operation must be retried after the device is made non-busy. For CPU/Memory boards, the memory must have been changed to the unconfigured state prior to issuing the board unconfigure operation. The CPUs on the board are off-lined, powered off and removed from the Solaris CPU list. CPUs that have processes bound to them cannot be off-lined. See `psradm`(1M), `psrinfo`(1M), `pbind`(1M), and`p_online`(2) for more information on off-lining CPUs.

`-f`
Force a block on connecting a board marked as `disabled-at-boot` in the non-volatile `disabled-board-list` variable. See *Platform Notes:Sun Enterprise 6x00/5x00/4x00/3x00 Systems*

`-l`
List options. Supported as described in `cfgadm`(1M)cfgadm(1M).

The *type* field can be one of `cpu/mem`, `mem`, `dual-sbus`, `sbus-upa`, `dual-pci`, `soc+sbus`, `soc+upa`, `disk` or `unknown`.

The hardware-specific info field is set as follows: `[disabled at boot] [non-detachable] [100 MHz capable]`

For `sbus-upa` and `soc+upa` type boards, the following additional information appears first: [`single buffered ffb|double buffered ffb|no ffb installed`] For disk type boards, the following additional information appears first: `{target: # | no disk} {target: # | no disk}`

`-o` disable-at-boot | enable-at-boot
Modify the state of the non—volatile `disabled-board-list` variable. Use this the `-o` option in conjunction with the `-c` *function* or `-l` option.

Use `-o enable-at-boot` with the `-c connect` to override a block on connecting a `disabled-at-boot` board.

`-x` insert-test | remove-test
Perform a test.

Specify `remove-test` to change the driver state for the specified slot from `disconnected` to `empty` without the need for physically removing the board during automated test sequences.

Specify `insert-test` to change the driver state of a slot made to appear empty using the `remove-test` command to the `disconnected` state as if it had been inserted.

`-x` quiesce-test sysctrl0:slot1
Perform a test.

Allows the `quiesce` operation required for board connect operations to be exercised. The execution of this test confirms that, with the current software and hardware configuration, it is possible to quiesce the system. If a device or process cannot be quiesced, its name is printed in an error message. Any valid board attachment point can be used with this command, but since all systems have a slot1 the given form is recommended.

`-x` set-condition-test=#
Perform a test.

Allows the the condition of a system board attachment point to be set for testing the policy logic for state change commands. The new setting is given as a number indicating one of the following condition values:

0 unknown
1 ok
2 failing
3 failed
4 unusable

OPERANDS

The following operand is supported:

sysctrl0:slot# The attachment points for boards on `EXX00` systems are published by instance `0` of the `sysctrl` driver (`sysctrl0`). The names of the attachment points are numbered from `slot0` through `slot15`. Specify # as a number between `0` and `15`, indicating the slot number. This form conforms to the logical `ap_id` specification given in `cfgadm`(1M). The corresponding physical `ap_id`s are listed in the `FILES` section.

FILES

/usr/platform/sun4u/lib/cfgadm/sysctrl.so.1
Hardware specific library

/devices/central@1f,0/fhc@0,f8800000/clock-board@0,900000:slot*
Attachment Points

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWkvm.u

SEE ALSO

`cfgadm`(1M), `cfgadm_ac`(1M), `ifconfig`(1M), `mount`(1M), `pbind`(1M), `psradm`(1M), , `psrinfo`(1M), `config_admin`(3CFGADM), `attributes`(5)

Sun Enterprise 6x00, 5x00, 4x00 and 3x00 Systems Dynamic Reconfiguration User's Guide,

Platform Notes:Sun Enterprise 6x00/5x00/4x00/3x00 Systems

NOTES

Refer to the *Sun Enterprise 6x00, 5x00, 4x00 and 3x00 Systems Dynamic Reconfiguration User's Guide* for additional details regarding dynamic reconfiguration of EXX00 system CPU/Memory boards.

NAME

cfgadm_usb – USB hardware-specific commands for cfgadm

SYNOPSIS

/usr/sbin/cfgadm [-f] [-y | -n] [-v] -c *function ap_id...*

/usr/sbin/cfgadm -f [-y | -n] [-v] [-o *hardware_options*] -x *hardware_function ap_id...*

/usr/sbin/cfgadm -v [-a] [-s *listing_option*] [-l [*ap_id* | *ap_type...*]]

/usr/sbin/cfgadm -v -h [*ap_id...*]

DESCRIPTION

The Universal Serial Bus (USB) hardware-specific library `/usr/lib/cfgadm/usb.so.1` provides the functionality for administering USB devices via the `cfgadm`(1M) command. `cfgadm` operates on attachment points. For details regarding attachment points, refer to `cfgadm`(1M).

For USB administration, the only attachment points supported are the ports of hubs attached to the USB bus.

Attachment points are named through attachment point IDs (*ap_ids*). The USB bus is hierarchical, so the *ap_ids* are as well. USB hubs have ports, numbered from `1` to *n*. All USB *ap_ids* consist of a string of the following form:

`usb`*N*`/`*A*`[.`*B*`[.`*C*`[...]]]`

where

N is the *N*th USB host controller on the system,
A is port #*A* on the root (top) hub.
B is port #*B* of the hub plugged into port #*A* of the hub above it.
C is port #*C* of the hub plugged into port #*B* of the hub above it, and so forth.

For example, the first port on the root hub of USB controller `0` (the only controller), has a logical *ap_id*:

```
usb0/1
```

Similarly, the second port on the first external hub plugged into the first port on the root hub of the first USB controller has a logical *ap_id*:

```
usb0/1.2
```

For example, if the *ap_id* is `usb0/1.4.3.4`, it represents port `4` of the hub plugged into port `3` of the hub plugged into port `4` of the hub plugged into port `1` of the root hub of the first USB bus controller on the system.

The following listing of USB attachment points in the system includes all attachment points for USB ports, even if the port is empty (no device plugged in):

```
example# cfgadm -l
Ap_Id               Type         Receptacle   Occupant     Condition
usb0/1              USB-hub      connected    configured   ok
usb0/2              unknown      empty        unconfigured ok
```

```
usb0/1.1             USB-storage  connected    configured   ok
usb0/1.2             unknown      empty        unconfigured ok
usb0/1.3             unknown      empty        unconfigured ok
usb0/1.4             USB-composit connected    configured   ok
```

The receptacle states for attachment points at the USB port have the following meanings:

`connected` USB port is powered on and enabled. A USB device is plugged in to the port. The device is logically connected to the USB bus.

`disconnected` USB port is powered on and enabled. A USB device is plugged into the port. The device has been logically disconnected from the USB bus (using the `cfgadm -c disconnect` command).

`empty` USB port is powered on, but no device is plugged in to it.

The occupant states for devices at USB port attachment points at the USB port have the following meanings:

`configured` The USB device at the USB port is configured and usable by Solaris.

`unconfigured` The USB device at the USB port was explicitly off-lined using `cfgadm -c unconfigure`, or was not successfully configured for use with Solaris, for example, having no driver or a device problem.

The attachment point conditions are:

`ok` Normal state - ready for use.

`failing` Not used.

`failed` Not used.

`unusable` The user has physically removed a device while an application had the device open (there may be outstanding `I/O`). Users need to reinsert the same physical device and close the application properly before removing the device again. The port cannot configure other inserted devices until this is done.

If the original device cannot be reinserted into the port, see the *System Administration Guide, Volume 1* for instructions for clearing this attachment point condition.

`unknown` Not used.

A USB device can be hotplugged or hotunplugged at any time, and the system detects the event and takes the appropriate action.

It is not necessary to transition a receptacle to the `disconnected` state before removing its device from the USB. However, it is not recommended to hot-remove devices currently in use (such as removable disks currently opened by volume manager (see `vold`(1M)) or some other application).

OPTIONS

`cfgadm` defines several types of operations. These operations include invoking configuration state changes (`-c`), invoking hardware-specific functions (`-x`), and obtaining configuration administration help messages (`-h`).

If any of these operations fail, the device and attachment point may not be in the expected state. Use the `cfgadm -l` command to display the device's current status.

All other options have the same meaning as defined in `cfgadm`(1M).

The following options are supported:

`-c` *function*

The following generic commands are defined for the USB hardware specific library. The following configuration state change operations are supported:

`configure`
If there is a USB device plugged into the port, this command attempts to configure it and set everything up so that it is usable by Solaris. This command does an implied `connect` (reverse of `disconnect`) if necessary. This command accomplishes nothing, and returns an error message, if the device at that port is already configured. After successful execution of this command, the device is ready for use under Solaris.

`disconnect`
Performs an `unconfigure` on the *ap_id* (if it is not already `unconfigured`), and then transitions the receptacle to the `disconnected` state, even though a device will still be plugged into the port. Issuing a `cfgadm -c configure`, or physically hotplugging the device, will bring the device back to the `connected` receptacle state, and to the `configured` occupant state, assuming a driver can be found and there are no problems enumerating and configuring the device.

`unconfigure`
Makes the device plugged into the port unusable by Solaris (offline it). If successful, `cfgadm` will now report this *ap_id*'s occupant state as *unconfigured*. Issuing a `configure` to the *ap_id* (if successful) will bring its occupant back to the `configured` (online) condition, as will physically hotplugging the device

on the port.

`-f` — Not supported.

`-h` *ap_id* — USB specific help may be obtained by using the help option with any USB attachment point.

`-l`[`v`] — The `-l` option works as described in `cfgadm`(1M). When paired with the `-v` option, the `Information` field contains the following USB-specific information:

- `Mfg`: manufacturer string (`iManufacturer`)
- `Product`: product string (`iProduct`)
- `NConfigs`: total number of configurations the device supports (`bNumConfigurations`).
- `Config`: current configuration setting in decimal (configuration index, not configuration value).
- The configuration string descriptor for the current configuration (`iConfiguration`)

See the Universal Serial Bus specification for a description of these fields.

`-o` *hardware_options* — Hardware options are only supported for the hardware-specific command, `-x usb_config`. See the description of that command below for an explanation of the options available.

`-s` *listing_options* — Attachment points of class USB may be listed by using the `select` sub-option. See `cfgadm`(1M).

`-x` *hardware_function* — The following hardware-specific functions are defined:

`usb_config -o config=`*n*

This command requires the mandatory `config` value to be specified using the `-o` option.

Sets the USB configuration of a multi-configuration USB device at *ap_id* to configuration index *n*. The device is set to this configuration henceforth and this setting persists across reboots, hot-removes, and unconfigure/configure of the device.

Valid values of *n* range from `0` to (`Nconfigs -1`). The device is reset by a `disconnect` followed by a `configure`. The `configure` causes the device to be configured to the new configuration setting.

If any of these steps fail, the configuration file and the device are restored to their previous state and an error message is issued.

usb_reset
Performs a software reset (re-enumeration) of the device. This is the equivalent of removing the device and inserting it back again. The port on the hub will be power cycled if the hub supports power cycling of individual ports.

If the connected device is a hub, this function has the effect of resetting that hub and any devices down the tree of which it is the root.

If any of these steps fail, the device is restored to its previous state and an error message is issued.

State table: attachment points state versus commands:

```
Valid states:
    empty/unconfigured         → no device connected

    disconnected/unconfigured  → logically disconnected,
                                 unavailable,
                                 devinfo node removed,
                                 device physically connected

    connected/unconfigured     → logically connected,
                                 unavailable,
                                 devinfo node present

    connected/configured       → connected, available
```

The table below clarifies the state transitions resulting from actions or commands:

```
current state      operation           new state
-------------      ---------           ---------
empty/
unconfigured:
               device plugged in:     connected/configured or
                                      connected/unconfigured
                                      (if enumeration failed)
               device removed:        n/a
               cfgadm -c unconfigure: empty/unconfigured
               cfgadm -c configure:   empty/unconfigured
               cfgadm -c disconnect:  empty/unconfigured
                                      (no-op and error)

disconnected/
unconfigured:
               device plugged in:     n/a
               device removed:        empty/unconfigured
               cfgadm -c unconfigure: disconnected/unconfigured
               cfgadm -c configure:   connected/configured, or
                                      connected/unconfigured
                                      (if reenumeration failed)
             cfgadm -c disconnect:    disconnected/unconfigured

connected/unconfigured:
```

```
             device plugged in:      n/a
             device removed:         empty/unconfigured
             cfgadm -c unconfigure:  connected/unconfigured
             cfgadm -c configure:    connected/configured, or
                                     connected/unconfigured
                                     (if reenumeration failed)
             cfgadm -c disconnect:   disconnected/unconfigured

connected/configured:
             device plugged in:      n/a
             device removed:         empty/unconfigured or
                                     connected/configured,
                                     but with ap condition
                                     'unusable' if device
                                     was open when removed
             cfgadm -c unconfigure:  connected/unconfigured
             cfgadm -c configure:    connected/configured
             cfgadm -c disconnect:   disconnected/unconfigured
```

EXAMPLES

EXAMPLE 1 Listing the status of all USB devices

The following command lists the status of all USB devices on the system:

```
# cfgadm
Ap_Id           Type         Receptacle   Occupant     Condition
usb0/1          USB-hub      connected    configured   ok
usb0/2          unknown      empty        unconfigured ok
usb0/1.1        USB-storage  connected    configured   ok
usb0/1.2        unknown      empty        unconfigured ok
usb0/1.3        unknown      empty        unconfigured ok
usb0/1.4        USB-composit connected    configured   ok
```

Notice that `cfgadm` treats the *composite* device at `ap_id` `usb0/1.4` as a single unit, since it cannot currently control individual interfaces.

EXAMPLE 2 Listing the status of a port with no device plugged in

The following command lists the status of a port with no device plugged in:

```
example# cfgadm -l usb0/1.3
Ap_Id           Type         Receptacle   Occupant     Condition
usb0/1.3        unknown      empty        unconfigured ok
```

EXAMPLE 3 Listing the status of the same port with a device plugged in

The following command lists the status of the same port after physically plugging in a device that configures without problems:

```
example# cfgadm -l usb0/1.3
Ap_Id           Type         Receptacle   Occupant     Condition
usb0/1.3        USB-hub      connected    configured   ok
```

EXAMPLE 4 Unconfiguring an existing USB device

The following command unconfigures the USB device attached to `usb0/1.3`, then displays the status of the `ap_id`:

EXAMPLE 4 Unconfiguring an existing USB device *(Continued)*

```
example# cfgadm -c unconfigure usb0/1.3
Unconfigure the device: /devices/pci@0,0/pci8086,7112@7,2/hub@2:2.3
This operation will suspend activity on the USB device
Continue (yes/no)?

Enter:

y

example# cfgadm -l usb0/1.3
Ap_Id           Type         Receptacle   Occupant     Condition
usb0/1.3        unknown      connected    unconfigured ok
```

EXAMPLE 5 Unconfiguring and logically disconnecting an existing USB device

The following command unconfigures and logically disconnects a USB device attached to `usb0/1.3`:

```
example# cfgadm -c disconnect usb0/1.3
Disconnect the device: /devices/pci@0,0/pci8086,7112@7,2/hub@2:2.3
This operation will suspend activity on the USB device
Continue (yes/no)?

Enter:

y

example# cfgadm -l usb0/1.3
Ap_Id         Type         Receptacle     Occupant        Condition
usb0/1.3      unknown      disconnected   unconfigured    ok
```

A `disconnect` implies that `cfgadm` does an `unconfigure` first. The receptacle status now shows `disconnected`, even though the device is still physically connected. In this case, a physical hotplug or using the `cfgadm -c configure` on the `ap_id` will bring it back on-line.

EXAMPLE 6 Configuring a previously unconfigured USB device

The following command configures a USB device that was previously attached to `usb0/1.3`:

```
example # cfgadm -yc configure usb0/1.3
example# cfgadm -l usb0/1.3
Ap_Id           Type         Receptacle   Occupant     Condition
usb0/1.3        unknown      connected    configured   ok
```

EXAMPLE 7 Resetting a USB device

The following command resets a USB device:

```
example# cfgadm -x usb_reset usb0/1.3
Reset the device: /devices/pci@0,0/pci8086,7112@7,2/hub@2:2.3
This operation will suspend activity on the USB device
```

EXAMPLE 7 Resetting a USB device *(Continued)*

```
Continue (yes/no)?

Enter:

y
```

EXAMPLE 8 Displaying detailed information about a USB device

The following command displays detailed information about a USB device. This device shows the following USB-specific information in the 'Information' field:

- Manufacturer string: Iomega
- Product string: USB Zip 250
- Number of configurations supported: 1
- Configuration currently active: 0
- Configuration string descriptor for configuration 0: Default

```
example# cfgadm -lv  usb0/1.5
Ap_Id                Receptacle   Occupant     Condition  Information
When         Type         Busy         Phys_Id
usb0/1.5     connected    configured   ok         Mfg:"Io
mega"  Product:"USB Zip 250"  NConfigs:1  Config:0 : Default

example# cfgadm -l -s "cols=ap_id:info" usb0/1.5
Ap_Id                        Information
usb0/1.5                     Mfg:"Iomega"  Product:"USB Zip 250"
NConfigs:1  Config:0 : Default
```

EXAMPLE 9 Displaying detailed information about all USB devices on the system

The following command displays detailed information about all USB devices on the system:

```
example# cfgadm -l -s "select=class(usb),cols=ap_id:info"
Ap_Id                        Information
usb0/1                       Mfg:<undefined>  Product:<undefined>
NConfigs:1  Config:0 <no cfg str descr>
usb0/2
usb0/1.1                     Mfg:<undefined>  Product:<undefined>
NConfigs:1  Config:0 <no cfg str descr>
usb0/1.2
usb0/1.3
usb0/1.4                     Mfg:"Wizard"  Product:"Modem/ISDN"
NConfigs:3  Config:1 : V.90 Analog Modem
usb0/1.5                     Mfg:"Iomega"  Product:"USB Zip 250"
NConfigs:1  Config:0 : Default
usb0/1.6                     Mfg:"SOLID YEAR"  Product:"SOLID YEAR
USB"NConfigs:1  Config:0 <no cfg str descr>
usb0/1.7
```

Lines containing only an ap_id are empty ports. These can be filtered out. This example only lists USB ap_ids with connected devices, and information about those devices.

EXAMPLE 9 Displaying detailed information about all USB devices on the system *(Continued)*

```
example# cfgadm -l -s "select=class(usb),cols=ap_id:info" | grep Mfg
usb0/1                        Mfg:<undefined>  Product:<undefined>
NConfigs:1  Config:0 <no cfg str descr>
usb0/1.1                      Mfg:<undefined>  Product:<undefined>
NConfigs:1  Config:0 <no cfg str descr>
usb0/1.4                      Mfg:"Wizard"  Product:"Modem/ISDN"
NConfigs:3  Config:1 : V.90 Analog Modem
usb0/1.5                      Mfg:"Iomega"  Product:"USB Zip 250"
NConfigs:1  Config:0 : Default
usb0/1.6                      Mfg:"SOLID YEAR"  Product:"SOLID YEAR USB"
Config:0 <no cfg str descr>
```

EXAMPLE 10 Listing information about a multi-configuration USB device

The following example lists information about a multi-configuration USB device.

Notice the `NConfigs` field: the configurations available for this device are `0`, `1`, and `2` (`0` to (*N*`Configs-1`)).

```
example# cfgadm -l -s "cols=ap_id:info" usb0/1.4
Ap_Id                         Information
usb0/1.4                      Mfg:"Wizard"  Product:"Modem/ISDN"
NConfigs:3  Config:1 V.90 Analog Modem"
```

EXAMPLE 11 Setting the current configuration of a multi-configuration USB device

The following example sets the current configuration of a multi-configuration USB device:

```
example# cfgadm -o config=2 -x usb_config usb0/1.4
Setting the device: /devices/pci@1f,2000/usb@1/device@3
to USB configuration 2
This operation will suspend activity on the USB device
Continue (yes/no)?

Enter:

y

USB configuration changed successfully.
```

The device path should be checked to ensure that the right instance of a device is being referred to, in the case where multiple devices of the exact same type are on the same bus. This information is available in the '`Information`' field.

FILES

`/usr/lib/cfgadm/usb.so.1`
Hardware specific library for generic USB device administration

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWusb (32-bit)
	SUNWusbx (64-bit)

SEE ALSO

`cfgadm`(1M), `vold`(1M), `config_admin`(3CFGADM), `attributes`(5), `scsa2usb`(7D), `usba`(7D)

Universal Serial Bus 1.1 Specification (`www.usb.org`)

System Administration Guide, Volume 1

NOTES

`cfgadm`(1M) can not unconfigure, disconnect, reset, or change the configuration of any USB device currently opened by `vold`(1M) or any other application. These operations also fail on a hub if a device in its hierarchy is opened by an application. See `scsa2usb`(7D) for unconfiguring a USB mass-storage device that is being used by `vold`(1M).

Only super-users can execute any functions on an attachment point. However, one need not be a super-user to list the attachment points.

NAME

cfsadmin – administer disk space used for caching file systems with the Cache File-System (CacheFS)

SYNOPSIS

cfsadmin -c [-o *cacheFS-parameters*] *cache_directory*

cfsadmin -d {*cache_ID* | all} *cache_directory*

cfsadmin -l *cache_directory*

cfsadmin -s {*mntpt1 ....*} | all

cfsadmin -u [-o *cacheFS-parameters*] *cache_directory*

DESCRIPTION

The `cfsadmin` command provides the following functions:

- cache creation
- deletion of cached file systems
- listing of cache contents and statistics
- resource parameter adjustment when the file system is unmounted.

You must always supply an option for `cfsadmin`. For each form of the command except `-s`, you must specify a cache directory, that is, the directory under which the cache is actually stored. A path name in the front file system identifies the cache directory. For the `-s` form of the command, you must specify a mount point.

You can specify a cache ID when you mount a file system with CacheFS, or you can let the system generate one for you. The `-l` option includes the cache ID in its listing of information. You must know the cache ID to delete a cached file system.

OPTIONS

`-c` [`-o` *cacheFS-parameters*] *cache_directory*
Create a cache under the directory specified by *cache_directory*. This directory must not exist prior to cache creation.

`-d` { *cache_ID* | all } *cache_directory*
Remove the file system whose cache ID you specify and release its resources, or remove all file systems in the cache by specifying `all`. After deleting a file system from the cache, you must run the `fsck_cachefs`(1M) command to correct the resource counts for the cache.

As indicated by the syntax above, you must supply either a *cache_ID* or `all`, in addition to *cache_directory*.

`-l` *cache_directory*
List file systems stored in the specified cache, as well as statistics about them. Each cached file system is listed by cache ID. The statistics document resource utilization and cache resource parameters.

`-s` { *mntpt1* ... } | all
Request a consistency check on the specified file system (or all `cachefs` mounted file systems). The `-s` option will only work if the cache file system was mounted with `demandconst` enabled (see `mount_cachefs`(1M)). Each file in the specified cache file system is checked for consistency with its corresponding file in the back file system. Note that the consistency check is performed file by file as files are

accessed. If no files are accessed, no checks are performed. Use of this option does not result in a sudden "storm" of consistency checks.

As indicated by the syntax above, you must supply one or more mount points, or `all`.

`-u` [`-o` *cacheFS-parameters*] *cache_directory*
Update resource parameters of the specified cache directory. Parameter values can only be increased. To decrease the values, you must remove the cache and recreate it. All file systems in the cache directory must be unmounted when you use this option. Changes will take effect the next time you mount any file system in the specified cache directory. The `-u` option with no `-o` option sets all parameters to their default values.

CacheFS Resource Parameters

You can specify the following CacheFS resource parameters as arguments to the `-o` option. Separate multiple parameters with commas.

`maxblocks=`*n*
Maximum amount of storage space that CacheFS can use, expressed as a percentage of the total number of blocks in the front file system. If CacheFS does not have exclusive use of the front file system, there is no guarantee that all the space the `maxblocks` parameter allows will be available. The default is `90`.

`minblocks=`*n*
Minimum amount of storage space, expressed as a percentage of the total number of blocks in the front file system, that CacheFS is always allowed to use without limitation by its internal control mechanisms. If CacheFS does not have exclusive use of the front file system, there is no guarantee that all the space the `minblocks` parameter attempts to reserve will be available. The default is `0`.

`threshblocks=`*n*
A percentage of the total blocks in the front file system beyond which CacheFS cannot claim resources once its block usage has reached the level specified by `minblocks`. The default is `85`.

`maxfiles=`*n*
Maximum number of files that CacheFS can use, expressed as a percentage of the total number of inodes in the front file system. If CacheFS does not have exclusive use of the front file system, there is no guarantee that all the inodes the `maxfiles` parameter allows will be available. The default is `90`.

`minfiles=`*n*
Minimum number of files, expressed as a percentage of the total number of inodes in the front file system, that CacheFS is always allowed to use without limitation by its internal control mechanisms. If CacheFS does not have exclusive use of the front file system, there is no

		guarantee that all the inodes the `minfiles` parameter attempts to reserve will be available. The default is `0`.
	`threshfiles=`*n*	A percentage of the total inodes in the front file system beyond which CacheFS cannot claim inodes once its usage has reached the level specified by `minfiles`. The default is `85`.
	`maxfilesize=`*n*	Largest file size, expressed in megabytes, that CacheFS is allowed to cache. The default is `3`. You cannot decrease the block or inode allotment for a cache. To decrease the size of a cache, you must remove it and create it again with different parameters. Currently `maxfilesize` is ignored by `cachefs`, therefore, setting it will have no effect.
OPERANDS	*cache_directory*	The directory under which the cache is actually stored.
	mntpt1	The directory where the CacheFS is mounted.

USAGE

See `largefile`(5) for the description of the behavior of `cfsadmin` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXAMPLES

EXAMPLE 1 Creating a cache directory.

The following example creates a cache directory named `/cache`:

```
example# cfsadmin -c /cache
```

EXAMPLE 2 Creating a cache specifying maxblocks, minblocks and threshblocks.

The following example creates a cache named `/cache1` that can claim a maximum of 60 percent of the blocks in the front file system, can use 40 percent of the front file system blocks without interference by CacheFS internal control mechanisms, and has a threshold value of 50 percent. The threshold value indicates that after CacheFS reaches its guaranteed minimum, it cannot claim more space if 50 percent of the blocks in the front file system are already used.

```
example# cfsadmin -c -o maxblocks=60,minblocks=40, threshblocks=50 /cache1
```

EXAMPLE 3 Changing the `maxfilesize` parameter.

The following example changes the `maxfilesize` parameter for the cache directory `/cache2` to 2 megabytes:

```
example# cfsadmin -u -o maxfilesize=2 /cache2
```

EXAMPLE 3 Changing the `maxfilesize` parameter. *(Continued)*

EXAMPLE 4 Listing the contents of a cache directory.

The following example lists the contents of a cache directory named `/cache3` and provides statistics about resource utilization:

```
example# cfsadmin -l /cache3
```

EXAMPLE 5 Removing a cached file system.

The following example removes the cached file system with cache ID `23` from the cache directory `/cache3` and frees its resources (the cache ID is part of the information returned by `cfsadmin -l`):

```
example# cfsadmin -d 23 /cache3
```

EXAMPLE 6 Removeing all cached file systems.

The following example removes all cached file systems from the cache directory `/cache3`:

```
example# cfsadmin -d all /cache3
```

EXAMPLE 7 Checking for consistency in file systems.

The following example checks for consistency all file systems mounted with `demandconst` enabled. No errors will be reported if no `demandconst` file systems were found.

```
example# cfsadmin -s all
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`cachefslog`(1M), `cachefsstat`(1M), `cachefswssize`(1M), `fsck_cachefs`(1M), `mount_cachefs`(1M), `attributes`(5), `largefile`(5)

NAME

cg14config – configure the cgfourteen device

SYNOPSIS

/usr/platform/ *platform-name* /sbin/cg14config [-d *device*] [-r *resolution*] [-g *gammavalue*] [-G *gammafile*] [-u *degammavalue*] [-U *degammafile*]

DESCRIPTION

cg14config sets up state on the selected cgfourteen device. *platform-name* can be found using the -i option of uname(1).

cg14config is supported only on Desktop SPARCsystems with SX graphics option.

The interface, output, and command location are uncommitted and subject to change in future releases.

OPTIONS

-d *device*
: Use *device* as the cgfourteen device to configure. Default is /dev/fb.

-r *resolution*
: Use *resolution* as the desired screen resolution. Resolution is specified in terms of screen width and height (in pixels), and vertical refresh (in hz). Available resolutions are:

```
1024x768@60
1024x768@66
1024x768@70
1152x900@66
1152x900@76
1280x1024@66
1280x1024@76
1600x1280@66
1920x1080@72
```

The default is the value read from the monitor sense codes. Note that some or all of the resolutions above may not be supported by any given monitor. If a programmed resolution is outside of the range of allowable values for a monitor, unpredictable results can occur, including damage to the monitor. Thus, care should be taken when programming the resolution. See *Openboot Command Reference* for a description of how to reset the console device to the default value if it becomes unusable from programming an unsupported resolution.

The -r option is not available when the window system is running.

-g *gammavalue*
: Each entry of the gamma lookup table will be loaded with entry^(1/*gammavalue*). The gamma lookup table has 256 entries. Default *gammavalue* is 2.2.

-G *filename*
: Initialize the gamma lookup table with the contents of *filename*. The format of *filename* is 256 triplets (red green

blue) of non-negative integers separated by NEWLINE characters. The integers must be in the range 0 to 1023, inclusive.

`-u` *degammavalue* — Each entry of the degamma lookup table will be loaded with entry^(*degammavalue*). The degamma lookup table has 256 entries. Default *degammavalue* is `2.2`.

`-U` *filename* — Initialize the degamma lookup table with the contents of *filename*. The format of *filename* is 256 entries of non-negative integers separated by NEWLINE characters. The integers must be in the range 0 to 255, inclusive.

EXIT STATUS

`cg14config` returns `0` on success and a positive integer on failure.

`1` Selected device is not a cgfourteen device.

`2` Requested action failed.

`3` Unsupported resolution.

`4` Gamma or degamma value out of range.

FILES

`/platform/`*platform-name*`/kernel/drv/cgfourteen` `cgfourteen` device driver

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWkvm

SEE ALSO

`uname`(1), `init`(1M), `mmap`(2), `attributes`(5)

Platform Notes: SPARCstation 10SX System Configuration Guide

Openboot Command Reference

NAME

chat – automated conversational exchange tool

SYNOPSIS

chat [*options*] *script*

DESCRIPTION

The `chat` program implements a conversational text-based exchange between the computer and any serial device, including (but not limited to) a modem, an ISDN TA, and the remote peer itself, establishing a connection between the Point-To-Point Protocol daemon (`pppd`) and the remote `pppd` process.

OPTIONS

The `chat` command supports the following options:

`-f` <*chat file*>
: Read the `chat` script from the `chat` file. This option is mutually exclusive with the `chat` script parameters. You must have `read` access to use the file. Multiple lines are permitted in the file. Use the space or horizontal tab characters to separate the strings.

`-t` <*timeout*>
: Set the timeout for the expected string to be received. If the string is not received within the time limit, the reply string is not sent. If specified, a 'subexpect' (alternate reply) string can be sent. Otherwise, if no alternate reply strings remain, the `chat` script fails.. A failed script will cause the `chat` program to terminate with a non-zero error code.

`-r` <*report file*>
: Set the file for output of the report strings. If you use the keyword `REPORT`, the resulting strings are written to this file. If the `-r` option is not used and you use the `REPORT` keyword, the `stderr` file is used for the report strings.

`-e`
: Start with the echo option turned on. You turn echo on or off at specific points in the `chat` script using the `ECHO` keyword. When echoing is enabled, all output from the modem is echoed to `stderr`.

`-E`
: Enables environment variable substitution within `chat` scripts using the standard *$xxx* syntax.

`-v`
: Request that the `chat` script execute in a verbose mode. The `chat` program logs the execution state of the `chat` script as well as all text received from the modem and output strings sent to the modem. The default is to log through `syslog`(3C) with facility `local2`; the logging method is alterable using the `-S` and `-s` options.

`-V`
: Request that the `chat` script be executed in a `stderr` verbose mode. The `chat` program logs all text received from the modem and output strings sent to the modem

	to `stderr`. `stderr` is usually the local console at the station running the `chat` or `pppd` program.
`-s`	Use `stderr`. Log messages from `-v` and error messages are sent to `stderr`.
`-S`	Do not use syslog. By default, error messages are set to syslog. This option prevents log messages from `-v` and error messages from being sent to syslog.
`-T` <*phone number*>	Pass in an arbitrary string (usually a telephone number) that will be substituted for the `\T` substitution metacharacter in a send string.
`-U` <*phone number 2*>	Pass in a second string (usually a telephone number) that will be substituted for the `\U` substitution metacharacter in a send string. This is useful when dialing an ISDN terminal adapter that requires two numbers.
`script`	If the script is not specified in a file with the `-f` option, the script is included as parameters to the `chat` program.

Chat Script

The `chat` script defines communications. A script consists of one or more "expect-send" pairs of strings separated by spaces, with an optional "subexpect-subsend" string pair, separated by a dash (as in the following example:)

ogin:-BREAK-ogin: ppp ssword: hello2u2

The example indicates that the `chat` program should expect the string "ogin:". If it fails to receive a login prompt within the time interval allotted, it sends a break sequence to the remote and then expects the string "ogin:". If the first "ogin:" is received, the break sequence is not generated.

Upon receiving the login prompt, the `chat` program sends the string ”ppp” and then expects the prompt "ssword:". When the password prompt is received, it sends the password hello2u2.

A carriage return is normally sent following the reply string. It is not expected in the "expect" string unless it is specifically requested by using the `\r` character sequence.

The expect sequence should contain only what is needed to identify the received data. Because it’s stored on a disk file, it should not contain variable information. Generally it is not acceptable to look for time strings, network identification strings, or other variable pieces of data as an expect string.

To correct for characters that are corrupted during the initial sequence, look for the string "ogin:" rather than "login:". The leading "l" character may be received in error, creating problems in finding the string. For this reason, scripts look for "ogin:" rather than "login:" and "ssword:" rather than "password:".

An example of a simple script follows:

```
ogin: ppp ssword: hello2u2
```

The example can be intrepreted as: expect ogin:, send ppp, expect ...ssword:, send hello2u2.

When login to a remote peer is necessary, simple scripts are rare. At minimum, you should include sub-expect sequences in case the original string is not received. For example, consider the following script:

```
ogin:--ogin: ppp ssword: hello2u2
```

This script is more effective than the simple one used earlier. The string looks for the same login prompt; however, if one is not received, a single return sequence is sent and then the script looks for login: again. If line noise obscures the first login prompt, send the empty line to generate a login prompt again.

Comments

Comments can be embedded in the `chat` script. Comment lines are ignored by the chat program. A comment starts with the hash ("#") character in column one. If a # character is expected as the first character of the expect sequence, quote the expect string. If you want to wait for a prompt that starts with a # character, write something like this:

```
# Now wait for the prompt and send logout string
'# ' logout
```

Sending Data From A File

If the string to send begins with an at sign ("@"), the remainder of the string is interpreted as the name of the file that contains the string. If the last character of the data read is a newline, it is removed. The file can be a named pipe (or fifo) instead of a regular file. This enables `chat` to communicate with another program, for example, a program to prompt the user and receive a password typed in.

Abort

Many modems report the status of a call as a string. These status strings are often "CONNECTED" or "NO CARRIER" or "BUSY." If the modem fails to connect to the remote, you can terminate the script. Abort strings may be specified in the script using the ABORT sequence. For example:

```
ABORT BUSY ABORT 'NO CARRIER' '' ATZ OK ATDT5551212 CONNECT
```

This sequence expects nothing and sends the string ATZ. The expected response is the string OK. When OK is received, the string ATDT5551212 dials the telephone. The expected string is CONNECT. If CONNECT is received, the remainder of the script is

executed. When the modem finds a busy telephone, it sends the string BUSY, causing the string to match the abort character sequence. The script fails because it found a match to the abort string. If the NO CARRIER string is received, it aborts for the same reason.

Clr_Abort The CLR_ABORT sequence clears previously set ABORT strings. ABORT strings are kept in an array of a pre-determined size; CLR_ABORT reclaims the space for cleared entries, enabling new strings to use that space.

Say The SAY string enables the script to send strings to a user at a terminal via standard error. If `chat` is being run by `pppd` and `pppd` is running as a daemon (detached from its controlling terminal), standard error is normally redirected to the `/etc/ppp/connect-errors` file.

SAY strings must be enclosed in single or double quotes. If carriage return and line feed are required for the output, you must explicitly add them to your string.

The SAY string can provide progress messages to users even with "ECHO OFF." For example, add a line similar to the following to the script:

```
ABORT BUSY
ECHO OFF
SAY "Dialing your ISP...\n"
'' ATDT5551212
TIMEOUT 120
SAY "Waiting up to 2 minutes for connection ..."
CONNECT ''
SAY "Connected, now logging in ...\n"
ogin: account
ssword: pass
$ \c
SAY "Logged in OK ... \n"
```

This sequence hides script detail while presenting the SAY string to the user. In this case, you will see:

```
Dialing your ISP...
Waiting up to 2 minutes for connection...Connected, now logging in...
Logged in OK ...
```

Report REPORT is similar to the ABORT string. With REPORT, however, strings and all characters to the next control character (such as a carriage return), are written to the report file.

REPORT strings can be used to isolate a modem's transmission rate from its CONNECT string and return the value to the `chat` user. Analysis of the REPORT string logic occurs in conjunction with other string processing, such as looking for the expect string. It's possible to use the same string for a REPORT and ABORT sequence, but probably not useful.

Report strings may be specified in the script using the REPORT sequence. For example:

```
REPORT CONNECT
ABORT BUSY
ATDT5551212 CONNECT
ogin: account
```

The above sequence expects nothing, then sends the string ATDT5551212 to dial the telephone. The expected string is CONNECT. If CONNECT is received, the remainder of the script is executed. In addition, the program writes the string CONNECT to the report file (specified by `-r`) in addition to any characters that follow.

Clr_Report CLR_REPORT clears previously set REPORT strings. REPORT strings are kept in an array of a pre-determined size; CLR_REPORT reclaims the space for cleared entries so that new strings can use that space.

Echo ECHO determines if modem output is echoed to `stderr`. This option may be set with the `-e` option, but can also be controlled by the ECHO keyword. The "expect-send" pair ECHO ON enables echoing, and ECHO OFF disables it. With ECHO, you can select which parts of the conversation should be visible. In the following script:

```
ABORT   'BUSY'
ABORT   'NO CARRIER'
""    AT&F
OK\r\n  ATD1234567
\r\n    \c
ECHO    ON
CONNECT \c
ogin:   account
```

All output resulting from modem configuration and dialing is not visible, but output is echoed beginning with the CONNECT (or BUSY) message.

Hangup The HANGUP option determines if a modem hangup is considered as an error. HANGUP is useful for dialing systems that hang up and call your system back. HANGUP can be ON or OFF. When HANGUP is set to OFF and the modem hangs up (for example, following the first stage of logging in to a callback system), `chat` continues running the script (for example, waiting for the incoming call and second stage login prompt). When the incoming call is connected, use the HANGUP ON string to reinstall normal hang up signal behavior. An example of a simple script follows:

```
ABORT   'BUSY'
 ""  AT&F
 OK\r\n  ATD1234567
 \r\n    \c
 CONNECT \c
 'Callback login:' call_back_ID
 HANGUP OFF
 ABORT "Bad Login"
 'Callback Password:' Call_back_password
 TIMEOUT 120
```

```
CONNECT \c
HANGUP ON
ABORT "NO CARRIER"
ogin:--BREAK--ogin: real_account
```

Timeout The initial timeout value is 45 seconds. Use the `-t` parameter to change the intial timeout value.

To change the timeout value for the next expect string, the following example can be used:

```
''"AT&F
 OK ATDT5551212
 CONNECT \c
 TIMEOUT 10
 ogin:--ogin: username
 TIMEOUT 5
 assword: hello2u2
```

The example changes the timeout to ten seconds when it expects the login: prompt. The timeout is changed to five seconds when it looks for the password prompt.

Once changed, the timeout value remains in effect until it is changed again.

EOT The EOT special reply string instructs the `chat` program to send an EOT character to the remote. This is equivalent to using ^D\c as the reply string. The EOT string normally indicates the end-of-file character sequence. A return character is not sent following the EOT. The EOT sequence can embedded into the send string using the sequence ^D.

BREAK The BREAK special reply string sends a break condition. The break is a special transmitter signal. Many UNIX systems handle break by cycling through available bit rates, and sending break is often needed when the remote system does not support autobaud. BREAK is equivalent to using \K\c as the reply string. You embed the break sequence into the send string using the \K sequence.

Escape Sequences Expect and reply strings can contain escape sequences. Reply strings accept all escape sequences, while expect strings accept most sequences. A list of escape sequences is presented below. Sequences that are not accepted by expect strings are indicated.

`''` Expects or sends a null string. If you send a null string, `chat` sends the return character. If you expect a null string, `chat` proceeds to the reply string without waiting. This sequence can be a pair of apostrophes or quote mark characters.

`\b` Represents a backspace character.

`\c` Suppresses the newline at the end of the reply string. This is the only method to send a string without a trailing return character. This sequence must be at the end of the send string. For example, the sequence hello\c will simply send the characters h, e, l, l, o. (Not valid in expect.)

`\d`	Delay for one second. The program uses `sleep`(1) which delays to a maximum of one second. (Not valid in expect.)
`\K`	Insert a BREAK. (Not valid in expect.)
`\n`	Send a newline or linefeed character.
`\N`	Send a null character. The same sequence may be represented by `\0`. (Not valid in expect.)
`\p`	Pause for 1/10th of a second. (Not valid in expect.)
`\q`	Suppress writing the string to syslog. The string ?????? is written to the log in its place. (Not valid in expect.)
`\r`	Send or expect a carriage return.
`\s`	Represents a space character in the string. Can be used when it is not desirable to quote the strings which contains spaces. The sequence 'HI TIM' and HI\sTIM are the same.
`\t`	Send or expect a tab character.
`\T`	Send the phone number string as specified with the `-T` option. (Not valid in expect.)
`\U`	Send the phone number 2 string as specified with the `-U` option. (Not valid in expect.)
`\\`	Send or expect a backslash character.
`\ddd`	Collapse the octal digits (ddd) into a single ASCII character and send that character. (\000 is not valid in an expect string.)
`^C`	Substitute the sequence with the control character represented by C. For example, the character DC1 (17) is shown as ^Q. (Some characters are not valid in expect.)

ENVIRONMENT VARIABLES

Environment variables are available within `chat` scripts if the `-E` option is specified on the command line. The metacharacter `$` introduces the name of the environment variable to substitute. If the substition fails because the requested environment variable is not set, nothing is replaced for the variable.

EXIT STATUS

The `chat` program terminates with the following completion codes:

`0`	Normal program termination. Indicates that the script was executed without error to normal conclusion.
`1`	One or more of the parameters are invalid or an expect string was too large for the internal buffers. Indicates that the program was not properly executed.
`2`	An error occurred during the execution of the program. This may be due to a read or write operation failing or `chat` receiving a signal such as SIGINT.

3	A timeout event occurred when there was an expect string without having a "-subsend" string. This indicates that you may not have programmed the script correctly for the condition or that an unexpected event occurred and the expected string could not be found.
4	The first string marked as an ABORT condition occurred.
5	The second string marked as an ABORT condition occurred.
6	The third string marked as an ABORT condition occurred.
7	The fourth string marked as an ABORT condition occurred.
. . .	The other termination codes are also strings marked as an ABORT condition.

To determine which event terminated the script, use the termination code. It is possible to decide if the string "BUSY" was received from the modem versus "NO DIALTONE." While the first event may be retried, the second probably will not succeed during a retry.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpppdu
Interface Stability	Evolving

SEE ALSO

`sleep`(1), `uucp`(1C), `pppd`(1M), `uucico`(1M), `syslog`(3C), `attributes`(5)

Additional information on `chat` scripts are available with UUCP documentation. The `chat` script format was taken from scripts used by the `uucico` program.

NAME

check-hostname – check if sendmail can determine the system's fully-qualified host name

SYNOPSIS

/usr/lib/mail/sh/check-hostname

DESCRIPTION

The `check-hostname` script is a migration aid for `sendmail`(1M). This script tries to determine the local host's fully-qualified host name (FQHN) in a manner similar to `sendmail`(1M). If `check-hostname` is able to determine the FQHN of the local host, it reports success. Otherwise, `check-hostname` reports how to reconfigure the system so that the FQHN can be properly determined.

FILES

`/etc/hosts`	host name database
`/etc/nsswitch.conf`	name service switch configuration file
`/etc/resolv.conf`	configuration file for name server routines

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsndmu

SEE ALSO

`sendmail`(1M), `hosts`(4), `attributes`(5)

NAME

check-permissions – check permissions on mail rerouting files

SYNOPSIS

/usr/lib/mail/sh/check-permissions [*login*]

DESCRIPTION

The `check-permissions` script is intended as a migration aid for `sendmail`(1M). It checks the `/etc/mail/sendmail.cf` file for all configured alias files, and checks the alias files for `:include:` files. It also checks for certain `.forward` files. For each file that `check-permissions` checks, it verifies that none of the parent directories are group- or world-writable. If any directories are overly permissive, it is reported. Otherwise it reports that no unsafe directories were found.

As to which `.forward` files are checked, it depends on the arguments included on the command line. If no argument is given, the current user's home directory is checked for the presence of a `.forward` file. If any arguments are given, they are assumed to be valid logins, and the home directory of each one is checked.

If the special argument `ALL` is given, the `passwd` entry in the `/etc/nsswitch.conf` file is checked, and all password entries that can be obtained through the switch file are checked. In large domains, this can be time-consuming.

OPERANDS

The following operands are supported:

login Where *login* is a valid user name, checks the home directory for *login*.

`ALL` Checks the home directory of *all* users.

FILES

`/etc/mail/sendmail.cf` defines enviornment for `sendmail`

`/etc/mail/aliases` ascii mail aliases file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsndmu

SEE ALSO

`getent`(1M), `sendmail`(1M), `aliases`(4), `attributes`(5)

NAME

chown – change owner

SYNOPSIS

/usr/ucb/chown [-f] [-R] *owner* [*group*] *filename...*

DESCRIPTION

chown changes the owner of the *filename*s to *owner*. The owner may be either a decimal user ID (UID) or a login name found in the password file. An optional *group* may also be specified. The group may be either a decimal group ID (GID) or a group name found in the GID file.

Only the super-user of the machine where the file is physically located can change owner, in order to simplify accounting procedures.

OPTIONS

-f Do not report errors.

-R Recursively descend into directories setting the ownership of all files in each directory encountered. When symbolic links are encountered, their ownership is changed, but they are not traversed.

FILES

/etc/passwd password file

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

chgrp(1), chown(2), group(4), passwd(4), attributes(5)

NAME

chroot – change root directory for a command

SYNOPSIS

/usr/sbin/chroot *newroot command*

DESCRIPTION

The `chroot` utility causes *command* to be executed relative to *newroot*. The meaning of any initial slashes (|) in the path names is changed to *newroot* for *command* and any of its child processes. Upon execution, the initial working directory is *newroot*.

Notice that redirecting the output of *command* to a file,

`chroot` *newroot command* `>x`

will create the file `x` relative to the original root of *command*, not the new one.

The new root path name is always relative to the current root. Even if a `chroot` is currently in effect, the *newroot* argument is relative to the current root of the running process.

This command can be run only by the super-user.

RETURN VALUES

The exit status of `chroot` is the return value of *command*.

EXAMPLES

EXAMPLE 1 Using the `chroot` utility.

The `chroot` utility provides an easy way to extract `tar` files (see `tar`(1)) written with absolute filenames to a different location:

```
example# cp /usr/sbin/static/tar /tmp
example# dd if=/dev/nrst0 | chroot /tmp tar xvf -
```

Note that `tar` is statically linked, so it is not necessary to copy any shared libraries to the *newroot* filesystem.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`cd`(1), `tar`(1), `chroot`(2), `ttyname`(3C), `attributes`(5)

NOTES

Exercise extreme caution when referencing device files in the new root file system.

References by routines such as `ttyname`(3C) to stdin, stdout, and stderr will find that the device associated with the file descriptor is unknown after `chroot` is run.

NAME

cimworkshop – start the Sun WBEM CIM WorkShop application

SYNOPSIS

/usr/sadm/bin/cimworkshop

DESCRIPTION

The `cimworkshop` command starts Sun WBEM CIM WorkShop, a graphical user interface that enables you to create, modify, and view the classes and instances that describe the managed resources on your system.

Managed resources are described using a standard information model called Common Information Model (CIM). A CIM class is a computer representation, or model, of a type of managed resource, such as a printer, disk drive, or CPU. A CIM instance is a particular managed resource that belongs to a particular class. Instances contain actual data. Objects can be shared by any WBEM-enabled system, device, or application. CIM objects are grouped into meaningful collections called schema. One or more schemas can be stored in directory-like structures called namespaces.

The CIM WorkShop application displays a Login dialog box. Context help is displayed on the left side of the CIM WorkShop dialog boxes. When you click on a field, the help content changes to describe the selected field.

By default, CIM WorkShop uses the RMI protocol to connect to the CIM Object Manager on the local host, in the default namespace, `root\cimv2`. You can select HTTP if you want to communicate to a CIM Object Manager using the standard XML/HTTP protocol from the Desktop Management Task Force. When a connection is established, all classes contained in the default namespace are displayed in the left side of the CIM WorkShop window.

The name of the current namespace is listed in the tool bar. All programming operations are performed within a namespace. Four namespaces are created in a `root` namespace during installation:

cimv2	Contains the default CIM classes that represent managed resources on your system.
security	Contains the security classes used by the CIM Object Manager to represent access rights for users and namespaces.
system	Contains properties for configuring the CIM Object Manager.
snmp	Contains pre-defined SNMP-related classes and all SNMP MOF files that are compiled.

The `cimworkshop` application allows you to perform the following tasks:

Create, view, and change namespaces.
: Use the CIM WorkShop application to view all namespaces. A namespace is a directory-like structure that can store CIM classes and instances.

Create, delete, and view CIM classes.
: You cannot modify the unique attributes of the classes that make up the CIM and Solaris Schema. You can create a new instance or subclass of the class and modify the desired attributes in that instance or subclass.

Create, modify, delete, and view CIM instances.

You can add instances to a class and modify its inherited properties or create new properties. You can also change the property values of a CIM instance.

Invoke methods.

You can set input values for a parameter of a method and invoke the method.

When CIM WorkShop connects to the CIM Object Manager in a particular namespace, all subsequent operations occur within that namespace. When you connect to a namespace, you can access the classes and instances in that namespace (if they exist) and in any namespaces contained in that namespace.

When you use CIM WorkShop to view CIM data, the WBEM system validates your login information on the current host. By default, a validated WBEM user is granted read access to the CIM Schema. The CIM Schema describes managed objects on your system in a standard format that all WBEM-enabled systems and applications can interpret.

Read Only	Allows read-only access to CIM Schema objects. Users with this privilege can retrieve instances and classes, but cannot create, delete, or modify CIM objects.
Read/Write	Allows full read, write, and delete access to all CIM classes and instances.
Write	Allows write and delete, but not read access to all CIM classes and instances.
None	Allows no access to CIM classes and instances.

USAGE

The `cimworkshop` command is not a tool for a distributed environment. Rather, this command is used for local administration on the machine on which the CIM Object Manager is running.

EXIT STATUS

The `cimworkshop` utility terminates with exit status `0`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWwbdev

SEE ALSO

`mofcomp`(1M), `wbemlogviewer`(1M), `init.wbem`(1M), `attributes`(5)

NAME

clear_locks – clear locks held on behalf of an NFS client

SYNOPSIS

`/usr/sbin/clear_locks` `[-s]` *hostname*

DESCRIPTION

The `clear_locks` command removes all file, record, and share locks created by the *hostname*and held on the current host, regardless of which process created or owns the locks.

This command can be run only by the super-user.

This command should only be used to repair the rare case of a client crashing and failing to clear held locks. Clearing locks held by an active client may cause applications to fail in an unexpected manner.

OPTIONS

`-s` Remove all locks created by the current machine and held by the server *hostname*.

OPERANDS

The following operands are supported:

hostname name of host server

EXIT STATUS

`0` Successful operation.

`1` If not root.

`2` Usage error.

`3` If unable to contact server (RPC).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`fcntl`(2), `attributes`(5)

NAME

clinfo – display cluster information

SYNOPSIS

```
clinfo [-nh]
```

DESCRIPTION

The `clinfo` command displays cluster configuration information about the node from which the command is executed.

Without arguments, `clinfo` returns an exit status of `0` if the node is configured and booted as part of a cluster. Otherwise, `clinfo` returns an exit status of `1`.

OPTIONS

The following options are supported:

`-h` Displays the highest node number allowed in the cluster. This is different from the maximum number of nodes supported in a given cluster. For example, `clinfo -h` might return 64, meaning that the highest number you can use to identify a node is 64. See the *Sun Cluster 3.0 System Administration Guide* for a description of utilities you can use to determine the number of nodes in a cluster.

`-n` Prints the number of the node from which `clinfo` is executed.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` An error occurred.

This is usually because the node is not configured or booted as part of a cluster.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`attributes`(5)

NAME

clri, dcopy – clear inode

SYNOPSIS

clri [-F *FSType*] [-V] *special i-number*

dcopy [-F *FSType*] [-V] *special i-number*

DESCRIPTION

clri writes zeros on the inodes with the decimal *i-number* on the file system stored on *special*. After clri, any blocks in the affected file show up as missing in an fsck(1M) of *special*.

Read and write permission is required on the specified file system device. The inode becomes allocatable.

The primary purpose of this routine is to remove a file that for some reason appears in no directory. If it is used to zap an inode that does appear in a directory, care should be taken to track down the entry and remove it. Otherwise, when the inode is reallocated to some new file, the old entry will still point to that file. At that point, removing the old entry will destroy the new file. The new entry will again point to an unallocated inode, so the whole cycle is likely to be repeated again and again.

dcopy is a symbolic link to clri.

OPTIONS

-F *FSType* — Specify the *FSType* on which to operate. The *FSType* should either be specified here or be determinable from /etc/vfstab by matching *special* with an entry in the table, or by consulting /etc/default/fs.

-V — Echo the complete command line, but do not execute the command. The command line is generated by using the options and arguments provided by the user and adding to them information derived from /etc/vfstab. This option should be used to verify and validate the command line.

USAGE

See largefile(5) for the description of the behavior of clri and dcopy when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

FILES

/etc/default/fs — Default local file system type

/etc/vfstab — List of default parameters for each file system

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

fsck(1M), vfstab(4), attributes(5), largefile(5)

NOTES

This command might not be supported for all *FSTypes*.

NAME

consadm – select or display devices used as auxiliary console devices

SYNOPSIS

/usr/sbin/consadm

/usr/sbin/consadm [-a *device* . . .] [-p]

/usr/sbin/consadm [-d *device* . . .] [-p]

/usr/sbin/consadm [-p]

DESCRIPTION

`consadm` selects the hardware *device* or devices to be used as auxiliary console devices, or displays the current device. Only superusers are allowed to make or display auxiliary console device selections.

Auxiliary console devices receive copies of `console` messages, and can be used as the console during `single user` mode. In particular, they receive kernel messages and messages directed to `/dev/sysmsg`. On Solaris or IA based systems they can also be used for interaction with the bootstrap.

By default, selecting a display device to be used as an auxiliary console device selects that device for the duration the system remains up. If the administrator needs the selection to persist across reboots the `-p` option can be specified.

`consadm` runs a daemon in the background, monitoring auxiliary console devices. Any devices that are disconnected (hang up, lose carrier) are removed from the auxiliary console device list, though not from the persistent list. While auxiliary console devices may have been removed from the device list receiving copies of `console` messages, those messages will always continue to be displayed by the default console device.

The daemon will not run if it finds there are not any auxiliary devices configured to monitor. Likewise, after the last auxiliary console is removed, the daemon will shut itself down. Therefore the daemon persists for only as long as auxiliary console devices remain active.

OPTIONS

The following options are supported:

`-a` *device*
: Adds *device* to the list of auxiliary console devices. Specify *device* as the path name to the device or devices to be added to the auxiliary console device list.

`-d` *device*
: Removes *device* from the list of auxiliary console devices. Specify *device* as the path name to the device or devices to be removed from the auxiliary console device list.

`-p`
: Prints the list of auxiliary consoles that will be auxiliary across reboots.

  When invoked with the `-a` or `-d` options , tells the application to make the change persist across reboot.

EXAMPLES

EXAMPLE 1 Adding to the list of devices that will receive console messages

The following command adds `/dev/term/a` to the list of devices that will receive console messages.

```
example# consadm -a /dev/term/a
```

EXAMPLE 2 Removing from the list of devices that will receive console messages

The following command removes `/dev/term/a` from the list of devices that will receive console messages. This includes removal from the persistent list.

```
example# consadm -d -p /dev/term/a
```

EXAMPLE 3 Printing the list of devices selected as auxiliary console devices

The following command prints the name or names of the device or devices currently selected as auxiliary console devices.

```
example# consadm
```

ENVIRONMENT VARIABLES

See `environ`(5) for descriptions of the following environment variables that affect the execution of `consadm`: `LC_CTYPE`, `LC_MESSAGES`, and `NLSPATH`.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu
Stability Level	Evolving

SEE ALSO

`eeprom`(1M), `syslogd`(1M), `kadb`(1M), `environ`(5), `attributes`(5), `sysmsg`(7d), `console`(7d)

NOTES

Auxiliary console devices are not usable for `kadb` or firmware I/O, do not receive panic messages, and do not receive output directed to `/dev/console`.

NAME

conv_lp – convert LP configuration

SYNOPSIS

conv_lp [-d *dir*] [-f *file*]

DESCRIPTION

conv_lp reads LP printer configuration information from a directory and converts it to an output file for use with print client software.

OPTIONS

The following options are supported:

-d *dir* The root (‘ / ’) directory from which LP configuration information is read. The default is root (‘ / ’).

-f *file* The output file to which conv_lp writes the converted LP configuration information. The default is /etc/printers.conf.

EXAMPLES

EXAMPLE 1 Converting LP Configuration Information from the Default Directory and File

The following example converts LP configuration information from directory root (/) to file /etc/printers.conf.

```
% conv_lp
```

EXAMPLE 2 Converting LP Configuration Information From a Specified Directory and File

The following example converts LP configuration information from directory /export/root/client to file /export/root/client/etc/printers.conf.

```
% conv_lp -d /export/root/client -f /export/root/client/etc/printers.conf
```

EXIT STATUS

The following exit values are returned:

0 Successful completion.

non-zero An error occurred.

FILES

/etc/printers.conf System printer configuration database.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpcu

SEE ALSO

lpset(1M), printers.conf(4), attributes(5)

NAME

conv_lpd – convert LPD configuration

SYNOPSIS

conv_lpd [-c printers | -c printcap] [-n] *file*

DESCRIPTION

conv_lpd converts LPD printer configuration information from *file* to a printers.conf or a printcap file (see printers.conf(4)). *file* specifies the name of the input file, and can be either in printers.conf or printcap format. If *file* is in printers.conf format, it converts it to a printcap file. If *file* is in printcap format, it converts it to a printers.conf file.

OPTIONS

The following options are supported:

-c printers | -c printcap	Specifies the type of output file produced by the conversion. -c printers converts to a printers.conf file. -c printcap converts to a printcap file. -c printers is the default.
-n	Preserves the namelist during the conversion.

OPERANDS

The following operands are supported:

file	The file to be converted.

EXAMPLES

EXAMPLE 1 Converting a printcap file to a printers.conf file.

The following example converts a printcap file to a printers.conf file.

```
example% conv_lpd /etc/printcap
```

EXAMPLE 2 Converting a printcap file to a printers.conf file and preserving the namelist.

The following example converts a printcap file to a printers.conf file and preserves the namelist.

```
example% conv_lpd -c printers -n /etc/printcap
```

EXAMPLE 3 Converting a printers.conf file to a printcap file and preserving the namelist.

The following example converts a printers.conf file to a printcap file and preserves the namelist.

```
example% conv_lpd -c printcap -n /etc/printers.conf
```

EXIT STATUS

The following exit values are returned:

0	Successful completion.
non-zero	An error occurred.

FILES

`/etc/printers.conf`	System printer configuration database.
`/etc/printcap`	SunOS 4.*x* printer capability database.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpcu

SEE ALSO

`lpset`(1M), `printers.conf`(4), `attributes`(5)

NAME

coreadm – core file administration

SYNOPSIS

coreadm [-g *pattern*] [-i *pattern*] [-d *option*...] [-e *option*...]

coreadm [-p *pattern*] [*pid*...]

coreadm -u

DESCRIPTION

The coreadm command is used to specify the name and location of core files produced by abnormally-terminating processes. See core(4).

The first form shown in the synopsis can be executed only by the super-user and is used to configure system-wide core file options, including a global core file name pattern and a per-process core file name pattern for the init(1M) process. All such settings are saved in coreadm's configuration file /etc/coreadm.conf for setting on reboot. See init(1M)

The second form can be executed by non-privileged users and is used to specify the file name pattern to be used by the operating system when generating a per-process core file.

The third form can be executed only by the super-user and is used to update all system-wide core file options based on the contents of /etc/coreadm.conf. Normally this option is used only on reboot by the startup script /etc/init.d/coreadm.

A core file name pattern is a normal file system path name with embedded variables, specified with a leading % character, that are expanded from values in effect when a core file is generated by the operating system. The possible variables are:

%p	process-ID
%u	effective user-ID
%g	effective group-ID
%f	executable file name, up to a maximum of MAXCOMLEN characters
%n	system node name (uname -n)
%m	machine name (uname -m)
%t	decimal value of time(2)
%%	literal %

For example, the core file name pattern:

```
/var/core/core.%f.%p
```

would result, for command foo with process-ID 1234, in the core file name: /var/core/core.foo.1234

The coreadm command with no arguments reports the current system configuration, for example:

```
$ coreadm
    global core file pattern: /var/core/core.%f.%p
      init core file pattern: core
           global core dumps: enabled
      per-process core dumps: enabled
     global setid core dumps: enabled
per-process setid core dumps: disabled
    global core dump logging: disabled
```

The `coreadm` command with only a list of process-IDs reports each process's per-process core file name pattern, for example:

```
$ coreadm 278 5678
  278:   core.%f.%p
  5678:  /home/george/cores/%f.%p.%t
```

Only the owner of a process or the super-user can interrogate a process in this manner.

When a process is dumping core, the operating system will generate two possible core files, the global core file and the per-process core file. Both files, one or the other, or no file will be generated, based on the system options in effect at the time.

When generated, a global core file will be created mode `600` and will be owned by the super-user. Non-privileged users cannot examine such files.

Ordinary per-process core files are created mode `600` under the credentials of the process. The owner of the process can examine such files.

A process that is or ever has been `setuid` or `setgid` since its last `exec`(2), including a process that began life with super-user privileges and gave up that privilege by way of `setuid`(2), presents security issues with respect to dumping core, as it may contain sensitive information in its address space to which the current non-privileged owner of the process should not have access. If setid core files are enabled, they will be created mode `600` and will be owned by the super-user.

OPTIONS

The following options are supported:

`-d` *option*...
: Disable the specified core file option. See the `-e` *option* for descriptions of possible options.

  Multiple `-e` and `-d` options can be specified on the command line. Only super-users can use this option.

`-e` *option*...
: Enable the specified core file option. Specify *option* as one of the following:

global	Allow core dumps using global core pattern
process	Allow core dumps using per-process core pattern
global-setid	Allow set-id core dumps using global core pattern
proc-setid	Allow set-id core dumps using per-process core pattern

log — Generate a `syslog`(3C) message when generation of a global core file is attempted. Multiple `-e` and `-d` options can be specified on the command line. Only super-users can use this option.

`-g` *pattern* — Set the global core file name pattern to *pattern*. The pattern must start with a `/` and can contain any of the special `%` variables described in the `DESCRIPTION`.

Only super-users can use this option.

`-i` *pattern* — Set the per-process core file name *pattern* for `init`(1M) to *pattern*. This is the same as `coreadm -p pattern 1` except that the setting will be persistent across reboot.

Only super-users can use this option.

`-p` *pattern* — Set the per-process core file name pattern to *pattern* for each of the specified process-IDs. The pattern can contain any of the special `%` variables described in the `DESCRIPTION` and need not begin with `/`. If it does not begin with `/`, it will be evaluated relative to the current directory in effect when the process generates a core file.

A non-privileged user can apply the `-p` option only to processes owned by that user. The super-user can apply it to any process. The per-process core file name pattern will be inherited by future child processes of the affected processes. See `fork`(2).

`-u` — Update system-wide core file options from the contents of the configuration file `/etc/coreadm.conf`. If the configuration file is missing or contains invalid values, default values are substituted. Following the update, the configuration file is resynchronized with the system core file configuration. Only super-users can use this option.

OPERANDS

The following operands are supported:

pid — process-ID

EXIT STATUS

The following exit values are returned:

0 — Successful completion.

1 — A fatal error occurred while either obtaining or modifying the system core file configuration.

2 — Invalid command line options were specified.

EXAMPLES

EXAMPLE 1 Setting the core file name pattern

When executed from a user's `$HOME/.profile` or `$HOME/.login`, the following command sets the core file name pattern for all processes run during the login session:

EXAMPLE 1 Setting the core file name pattern *(Continued)*

```
example$  coreadm -p core.%f.%p $$
```

`$$` is the process-id of the currently running shell. The per-process core file name pattern is inherited by all child processes.

EXAMPLE 2 Dumping user's files into a subdirectory

The following command dumps all of the user's core dumps into the `corefiles` subdirectory of the home directory, discriminated by the system node name. This is useful for users who use many different machines but have a shared home directory.

```
example$  coreadm -p $HOME/corefiles/%n.%f.%p $$
```

FILES

`/etc/init.d/coreadm`

`/etc/coreadm.conf`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`gcore`(1), `init`(1M), `exec`(2), `fork`(2), `setuid`(2), `time`(2), `syslog`(3C), `core`(4), `attributes`(5)

NAME

cpustat – monitor system behavior using CPU performance counters

SYNOPSIS

cpustat -c *eventspec* [-c *eventspec*]... [-ntD] [*interval* [*count*]]

cpustat -h

DESCRIPTION

The `cpustat` utility allows CPU performance counters to be used to monitor the overall behavior of the CPUs in the system.

If *interval* is specified, `cpustat` samples activity every *interval* seconds, repeating forever. If a *count* is specified, the statistics are repeated *count* times. If neither are specified, an interval of five seconds is used, and there is no limit to the number of samples that will be taken.

OPTIONS

The following options are supported:

`-c` *eventspec* — Specifies a set of events for the CPU performance counters to monitor. The syntax of these event specification can be determined using the `-h` option to cause the usage message to be generated. The semantics of these event specifications can be determined by reading the CPU manufacturers documentation for the events. See `cpc_strtoevent`(3CPC) for a description of the syntax.

Multiple `-c` options may be specified, in which case the command cycles between the different event settings on each sample.

`-D` — Enables debug mode.

`-h` — Prints an extensive help message on how to use the utility and how to program the processor-dependent counters.

`-n` — Omits all header output (useful if `cpustat` is the beginning of a pipeline).

`-t` — Prints an additional column of processor cycle counts, if available on the current architecture.

USAGE

A closely related utility, `cputrack`(1), can be used to monitor the behavior of individual applications with little or no interference from other activities on the system.

The `cpustat` utility must be run by the super-user, as there is an intrinsic conflict between the use of the CPU performance counters system-wide by `cpustat` and the use of the CPU performance counters to monitor an individual process (for example, by `cputrack`.)

Once any instance of this utility has started, no further per-process or per-LWP use of the counters is allowed until the last instance of the utility terminates.

The times printed by the command correspond to the wallclock time when the hardware counters were actually sampled, instead of when the program told the kernel to sample them. The time is derived from the same timebase as `gethrtime`(3C).

The processor cycle counts enabled by the `-t` option always apply to both user and system modes, regardless of the settings applied to the performance counter registers.

On some hardware platforms, the counters are implemented using 32-bit registers. While the kernel attempts to catch all overflows to synthesize 64-bit counters, because of hardware implementation restrictions, overflows may be lost unless the sampling interval is kept short enough. The events most prone to wrap are those that count processor clock cycles. If such an event is of interest, sampling should occur frequently so that less than 4 billion clock cycles can occur between samples.

The output of cpustat is designed to be readily parseable by `nawk`(1) and `perl`(1), thereby allowing performance tools to be composed by embedding `cpustat` in scripts. Alternatively, tools may be constructed directly using the same APIs that `cpustat` is built upon using the facilities of `libcpc`(3LIB). See `cpc`(3CPC).

The `cpustat` utility only monitors the CPUs that are accessible to it in the current processor set. Thus, several instances of the utility can be running on the CPUs in different processor sets. See `psrset`(1M) for more information about processor sets.

Because `cpustat` uses LWPs bound to CPUs, the utility may have to be terminated before the configuration of the relevant processor can be changed.

WARNINGS

By running the `cpustat` command, the super-user will forcibly invalidate all existing performance counter context. This may in turn cause all invocations of the `cputrack` command, and other users of performance counter context, to exit prematurely with unspecified errors.

If `cpustat` is invoked on a system that has CPU performance counters, but on which the packages containing the kernel support for those counters is not installed, the following message appears:

```
cpustat: CPU performance counters are inaccessible on this machine
```

This error message implies that `cpc_access()` has failed and is documented in `cpc_access`(3CPC). Review this documentation for more information about the problem and possible solutions.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcpcu
Interface Stability	Evolving

SEE ALSO `cputrack`(1), `nawk`(1), `perl`(1), `iostat`(1M), `prstat`(1M), `psrset`(1M), `vmstat`(1M), `cpc`(3CPC), `cpc_access`(3CPC), `cpc_strtoevent`(3CPC), `gethrtime`(3C), `libcpc`(3LIB), `attributes`(5)

Sun Microelectronics UltraSPARC I&II User's Manual, January 1997, STP1031, http://www.sun.com/sparc

Intel Architecture Software Developer's Manual, Volume 3: System Programmers Guide, 243192, http://developer.intel.com

NAME

cron – clock daemon

SYNOPSIS

/usr/sbin/cron

DESCRIPTION

The cron command starts a process that executes commands at specified dates and times. Regularly scheduled commands can be specified according to instructions found in crontab files in the directory /var/spool/cron/crontabs. Users can submit their own crontab file using the crontab(1) command. Commands which are to be executed only once may be submitted using the at(1) command.

cron only examines crontab or at command files during its own process initialization phase and when the crontab or at command is run. This reduces the overhead of checking for new or changed files at regularly scheduled intervals.

Since cron never exits, it should be executed only once. This is done routinely through /etc/rc2.d/S75cron at system boot time. The file /etc/cron.d/FIFO is used (among other things) as a lock file to prevent the execution of more than one instance of cron.

cron captures the output of the job's stdout and stderr streams, and, if it is non-empty, mails the output to the user. If the job does not produce output, no mail is sent to the user (unless the job is an at(1) job and the -m option was specified when the job was submitted).

cron and at jobs will be not be executed if the user's account is locked. Only accounts which are not locked as defined in shadow(4) will have their job or process executed.

Setting cron Defaults

To keep a log of all actions taken by cron, CRONLOG=YES (by default) must be specified in the /etc/default/cron file. If CRONLOG=NO is specified, no logging is done. Keeping the log is a user configurable option since cron usually creates huge log files.

The PATH for user cron jobs can be set using PATH= in /etc/default/cron. The PATH for root cron jobs can be set using SUPATH= in /etc/default/cron. The security implications of setting PATH and SUPATH should be carefully considered.

Example /etc/default/cron file:

```
CRONLOG=YES
PATH=/usr/bin:/usr/ucb:
```

This example enables logging and sets the default PATH used by non-root jobs to /usr/bin:/usr/ucb:. Root jobs will continue to use /usr/sbin:/usr/bin.

/etc/cron.d/logchecker is a script that checks to see if the log file has exceeded the system ulimit. If so, the log file is moved to /var/cron/olog.

FILES

/etc/cron.d	main cron directory
/etc/cron.d/FIFO	used as a lock file
/etc/default/cron	contains cron default settings

`/var/cron/log`	`cron` history information
`/var/spool/cron`	spool area
`/etc/cron.d/logchecker`	moves log file to `/var/cron/olog` if log file exceeds system ulimit.
`/etc/cron.d/queuedefs`	queue description file for `at`, `batch`, and `cron`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`at`(1), `crontab`(1), `sh`(1), `queuedefs`(4), `shadow`(4), `attributes`(5)

DIAGNOSTICS

A history of all actions taken by `cron` is stored in `/var/cron/log` and (possibly) `/var/cron/olog`.

NAME

cvcd – virtual console daemon

SYNOPSIS

/platform/*platform_name***/cvcd**

DESCRIPTION

The virtual console daemon, `cvcd`, is a server process that supports the network console provided on some platforms. The `cvcd` daemon accepts network console connections from a remote host (only one host at any given time). Console input is read from this connection and forwarded to `cvc`(7D) by way of `cvcredir`(7D).

Similarly, console output is read from `cvcredir`(7D) and forwarded across the network console connection. If `cvcd` dies, console traffic is automatically rerouted through an internal hardware interface.

The `cvcd` daemon normally starts at system boot time. Each domain supports only one `cvcd` process at a time.

`Caution:` On Sun Enterprise 10000 domains, `cvcd` uses a configuration file (`/etc/ssphostname`) to determine the name of the host from which network console connections are allowed. If the remote console host is renamed, you must edit the configuration file to reflect that change.

OPERANDS

The following operands are supported:

platform_name The official Sun platform name used in packaging and code. For example, for Sun Fire 15K servers, the *platform_name* would be `SUNW,Sun-Fire-15000`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Architecture	Sun Enterprise 10000 servers Sun Fire 15K servers
Availability	SUNWcvc.u

SEE ALSO

`services`(4), `attributes`(5), `cvc`(7D), `cvcredir`(7D)

Sun Enterprise 10000 SSP Reference Manual

Sun System Management Services (SMS) Reference Manual

NAME

dcs – domain configuration server

SYNOPSIS

/usr/lib/dcs [-s *sessions*]

DESCRIPTION

The domain configuration server (DCS) is a daemon process that runs on Sun servers that support remote dynamic reconfiguration (DR) clients. It is started by inetd(1M) when the first DR request is received from a client connecting to the network service sun-dr. After the DCS accepts a DR request, it uses the libcfgadm(3LIB) interface to execute the DR operation. After the operation is performed, the results are returned to the client.

The DCS listens on the network service labeled sun-dr. Its underlying protocol is TCP, and it is invoked as an inetd server using the TCP transport. The entries for the DCS in the /etc/inet/inetd.conf file are as follows:

```
sun-dr stream tcp wait root /usr/lib/dcs dcs
sun-dr stream tcp6 wait root /usr/lib/dcs dcs
```

These entries enable remote DR operations. Removing them does not negatively impact the server; however, all DR operations initiated from a remote host would fail.

OPTIONS

-s *sessions* — Sets the number of active sessions that the DCS allows at any one time. When the limit is reached, the DCS stops accepting connections until active sessions complete the execution of their DR operation. If this option is not specified, a default value of 128 is used.

ERRORS

The DCS uses syslog(3C) to report status and error messages. All of the messages are logged with the LOG_DAEMON facility. Error messages are logged with the LOG_ERR and LOG_NOTICE priorities, and informational messages are logged with the LOG_INFO priority. The default entries in the /etc/syslog.conf file log all of the DCS error messages to the /var/adm/messages log.

SEE ALSO

cfgadm_sbd(1M), inetd(1M), syslog(3C), config_admin(3CFGADM), libcfgadm(3LIB), inetd.conf(4), syslog.conf(4), dr(7D)

NAME

dd – convert and copy a file

SYNOPSIS

`/usr/bin/dd` [*operand=value*...]

DESCRIPTION

`dd` copies the specified input file to the specified output with possible conversions. The standard input and output are used by default. The input and output block sizes may be specified to take advantage of raw physical I/O. Sizes are specified in bytes; a number may end with `k`, `b`, or `w` to specify multiplication by 1024, 512, or 2, respectively. Numbers may also be separated by `x` to indicate multiplication.

`dd` will read the input one block at a time, using the specified input block size; it then will process the block of data actually returned, which could be smaller than the requested block size. `dd` will apply any conversions that have been specified and write the resulting data to the output in blocks of the specified output block size.

`cbs` is used only if `ascii`, `asciib`, `unblock`, `ebcdic`, `ebcdicb`, `ibm`, `ibmb`, or `block` conversion is specified. In the first two cases, `cbs` characters are copied into the conversion buffer, any specified character mapping is done, trailing blanks are trimmed, and a NEWLINE is added before sending the line to output. In the last three cases, characters up to NEWLINE are read into the conversion buffer and blanks are added to make up an output record of size `cbs`. ASCII files are presumed to contain NEWLINE characters. If `cbs` is unspecified or `0`, the `ascii`, `asciib`, `ebcdic`, `ebcdicb`, `ibm`, and `ibmb` options convert the character set without changing the input file's block structure; the `unblock` and `block` options become a simple file copy.

After completion, `dd` reports the number of whole and partial input and output blocks.

OPERANDS

The following operands are supported:

`if=file`
Specify the input path; standard input is the default.

`of=file`
Specify the output path; standard output is the default. If the `seek=expr` conversion is not also specified, the output file will be truncated before the copy begins, unless `conv=notrunc` is specified. If `seek=expr` is specified, but `conv=notrunc` is not, the effect of the copy will be to preserve the blocks in the output file over which `dd` seeks, but no other portion of the output file will be preserved. (If the size of the seek plus the size of the input file is less than the previous size of the output file, the output file will be shortened by the copy.)

`ibs=`*n*
Specify the input block size in *n* bytes (default is `512`).

`obs=`*n*
Specify the output block size in *n* bytes (default is `512`).

`bs=`*n*
Set both input and output block sizes to *n* bytes, superseding `ibs=` and `obs=`. If no conversion other than `sync`, `noerror`, and `notrunc` is specified, each input block will be copied to the output as a single block without aggregating short blocks.

`cbs=`*n*
Specify the conversion block size for `block` and `unblock` in bytes by *n* (default is `0`). If `cbs=` is omitted or given a value of `0`, using `block` or `unblock` produces unspecified results.

This option is used only if ASCII or EBCDIC conversion is specified. For the `ascii` and `asciib` operands, the input is handled as described for the `unblock` operand except that characters are converted to ASCII before the trailing SPACE characters are deleted. For the `ebcdic`, `ebcdicb`, `ibm`, and `ibmb` operands, the input is handled as described for the `block` operand except that the characters are converted to EBCDIC or IBM EBCDIC after the trailing SPACE characters are added.

`files=`*n*
Copy and concatenate *n* input files before terminating (makes sense only where input is a magnetic tape or similar device).

`skip=`*n*
Skip *n* input blocks (using the specified input block size) before starting to copy. On seekable files, the implementation will read the blocks or seek past them; on non-seekable files, the blocks will be read and the data will be discarded.

`iseek=`*n*
Seek *n* blocks from beginning of input file before copying (appropriate for disk files, where `skip` can be incredibly slow).

`oseek=`*n*
Seek *n* blocks from beginning of output file before copying.

`seek=`*n*
Skip *n* blocks (using the specified output block size) from beginning of output file before copying. On non-seekable files, existing blocks will be read and space from the current end-of-file to the specified offset, if any, filled with null bytes; on seekable files, the implementation will seek to the specified offset or read the blocks as described for non-seekable files.

`count=`*n*
Copy only *n* input blocks.

`conv=`*value*[`,`*value*`...`]
Where *value*s are comma-separated symbols from the following list:

`ascii`	Convert EBCDIC to ASCII.
`asciib`	Convert EBCDIC to ASCII using BSD-compatible character translations.

`ebcdic`	Convert ASCII to EBCDIC. If converting fixed-length ASCII records without NEWLINEs, set up a pipeline with `dd conv=unblock` beforehand.
`ebcdicb`	Convert ASCII to EBCDIC using BSD-compatible character translations. If converting fixed-length ASCII records without NEWLINEs, set up a pipeline with `dd conv=unblock` beforehand.
`ibm`	Slightly different map of ASCII to EBCDIC. If converting fixed-length ASCII records without NEWLINEs, set up a pipeline with `dd conv=unblock` beforehand.
`ibmb`	Slightly different map of ASCII to EBCDIC using BSD-compatible character translations. If converting fixed-length ASCII records without NEWLINEs, set up a pipeline with `dd conv=unblock` beforehand.

The `ascii` (or `asciib`), `ebcdic` (or `ebcdicb`), and `ibm` (or `ibmb`) values are mutually exclusive.

`block`	Treat the input as a sequence of NEWLINE-terminated or EOF-terminated variable-length records independent of the input block boundaries. Each record is converted to a record with a fixed length specified by the conversion block size. Any NEWLINE character is removed from the input line; SPACE characters are appended to lines that are shorter than their conversion block size to fill the block. Lines that are longer than the conversion block size are truncated to the largest number of characters that will fit into that size; the number of truncated lines is reported.
`unblock`	Convert fixed-length records to variable length. Read a number of bytes equal to the conversion block size (or the number of bytes remaining in the input, if less than the conversion block size), delete all trailing SPACE characters, and append a NEWLINE character.

The `block` and `unblock` values are mutually exclusive.

`lcase`	Map upper-case characters specified by the `LC_CTYPE` keyword `tolower` to the corresponding lower-case character. Characters for which no mapping is specified will not be modified by this conversion.
`ucase`	Map lower-case characters specified by the `LC_CTYPE` keyword `toupper` to the corresponding upper-case character. Characters for which no mapping is specified will not be modified by this conversion.

The `lcase` and `ucase` symbols are mutually exclusive.

`swab`	Swap every pair of input bytes. If the current input record is an odd number of bytes, the last byte in the input record is ignored.
`noerror`	Do not stop processing on an input error. When an input error occurs, a diagnostic message will be written on standard error, followed by the current input and output block counts in the same format as used at completion. If the `sync` conversion is specified, the missing input will be replaced with null bytes and processed normally; otherwise, the input block will be omitted from the output.
`notrunc`	Do not truncate the output file. Preserve blocks in the output file not explicitly written by this invocation of `dd`. (See also the preceding `of=file` operand.)
`sync`	Pad every input block to the size of the `ibs=` buffer, appending null bytes. (If either `block` or `unblock` is also specified, append SPACE characters, rather than null bytes.)

If operands other than `conv=` are specified more than once, the last specified `operand=`*value* will be used.

For the `bs=`, `cbs=`, `ibs=`, and `obs=` operands, the application must supply an expression specifying a size in bytes. The expression, `expr`, can be:

1. a positive decimal number
2. a positive decimal number followed by `k`, specifying multiplication by 1024
3. a positive decimal number followed by `b`, specifying multiplication by 512
4. two or more positive decimal numbers (with or without `k` or `b`) separated by `x`, specifying the product of the indicated values.

All of the operands will be processed before any input is read.

USAGE

See `largefile`(5) for the description of the behavior of `dd` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXAMPLES

EXAMPLE 1 Copying From Tape Drive 0 to Tape Drive 1:

The following example copies from tape drive 0 to tape drive 1, using a common historical device naming convention.

```
example% dd if=/dev/rmt/0h  of=/dev/rmt/1h
```

EXAMPLE 2 Stripping the First 10 bytes From Standard Input

The following example strips the first 10 bytes from standard input.

```
example% dd ibs=10  skip=1
```

EXAMPLE 2 Stripping the First 10 bytes From Standard Input *(Continued)*

EXAMPLE 3 Reading a Tape Into an ASCII File

This example reads an EBCDIC tape blocked ten 80-byte EBCDIC card images per block into the ASCII file `x` :

```
example% dd if=/dev/tape of=x ibs=800 cbs=80 conv=ascii,lcase
```

EXAMPLE 4 Using `conv=sync` to Write to Tape

The following example uses `conv=sync` when writing to a tape:

```
example% tar cvf - . | compress | dd obs=1024k of=/dev/rmt/0 conv=sync
```

ENVIRONMENT VARIABLES

See `environ`(5) for descriptions of the following environment variables that affect the execution of `dd`: `LC_CTYPE`, `LC_MESSAGES`, and `NLSPATH`.

EXIT STATUS

The following exit values are returned:

`0` The input file was copied successfully.

`>0` An error occurred.

If an input error is detected and the `noerror` conversion has not been specified, any partial output block will be written to the output file, a diagnostic message will be written, and the copy operation will be discontinued. If some other error is detected, a diagnostic message will be written and the copy operation will be discontinued.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`cp`(1), `sed`(1), `tr`(1), `attributes`(5), `environ`(5), `largefile`(5)

DIAGNOSTICS

`f+p records in(out)` numbers of full and partial blocks read(written)

NOTES

Do not use `dd` to copy files between file systems having different block sizes.

Using a blocked device to copy a file will result in extra nulls being added to the file to pad the final block to the block boundary.

When `dd` reads from a pipe, using the `ibs=X` and `obs=Y` operands, the output will always be blocked in chunks of size Y. When `bs=Z` is used, the output blocks will be whatever was available to be read from the pipe at the time.

When using `dd` to copy files to a tape device, the file size must be a multiple of the device sector size (for example, 512 Kbyte). To copy files of arbitrary size to a tape device, use `tar`(1) or `cpio`(1).

For `SIGINT, dd` will write status information to standard error before exiting. It will take the standard action for all other signals.

NAME

devattr – display device attributes

SYNOPSIS

devattr [-v] *device* [*attribute*...]

DESCRIPTION

devattr displays the values for a device's attributes. The display can be presented in two formats. Used without the -v option, only the attribute values are shown. Used with the -v option, the attributes are shown in an *attribute=value* format. When no attributes are given on the command line, all attributes for the specified device are displayed in alphabetical order by attribute name. If attributes are given on the command line, only those attributes are shown, displayed in command line order.

OPTIONS

The following options are supported:

-v Specifies verbose format. Attribute values are displayed in an *attribute=value* format.

OPERANDS

The following operands are supported:

attribute Defines which attribute, or attributes, should be shown. Default is to show all attributes for a device. See the putdev(1M) manual page for a complete listing and description of available attributes.

device Defines the device whose attributes should be displayed. Can be the pathname of the device or the device alias.

EXIT STATUS

The following exit values are returned:

0 successful completion.

1 Command syntax was incorrect, invalid option was used, or an internal error occurred.

2 Device table could not be opened for reading.

3 Requested device could not be found in the device table.

4 Requested attribute was not defined for the specified device.

FILES

/etc/device.tab

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

getdev(1M), putdev(1M), attributes(5)

NAME

devfree – release devices from exclusive use

SYNOPSIS

devfree *key* [*device*...]

DESCRIPTION

devfree releases devices from exclusive use. Exclusive use is requested with the command devreserv.

When devfree is invoked with only the *key* argument, it releases all devices that have been reserved for that *key*. When called with *key* and *device* arguments, devfree releases the specified devices that have been reserved with that *key*.

OPERANDS

The following operands are supported:

device Defines device that this command will release from exclusive use. *device* can be the pathname of the device or the device alias.

key Designates the unique key on which the device was reserved.

EXIT STATUS

The following exit values are returned:

0 Successful completion.

1 Command syntax was incorrect, an invalid option was used, or an internal error occurred.

2 Device table or device reservation table could not be opened for reading.

3 Reservation release could not be completely fulfilled because one or more of the devices was not reserved or was not reserved on the specified key.

FILES

/etc/device.tab

/etc/devlkfile

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

devreserv(1M), attributes(5)

NOTES

The commands devreserv and devfree are used to manage the availability of devices on a system. These commands do not place any constraints on the access to the device. They serve only as a centralized bookkeeping point for those who wish to use them. Processes that do not use devreserv may concurrently use a device with a process that has reserved that device.

NAME

devfsadm, devfsadmd – administration command for /dev and /devices

SYNOPSIS

/usr/sbin/devfsadm [-C] [-c *device_class*] [-i *driver_name*] [-n] [-r *root_dir*] [-s] [-t *table_file*] [-v]

/usr/lib/devfsadm/devfsadmd

DESCRIPTION

devfsadm(1M) maintains the /dev and /devices namespaces. It replaces the previous suite of devfs administration tools including drvconfig(1M), disks(1M), tapes(1M), ports(1M), audlinks(1M), and devlinks(1M).

The default operation is to attempt to load every driver in the system and attach to all possible device instances. devfsadm then creates device special files in /devices and logical links in /dev.

devfsadmd(1M) is the daemon version of devfsadm(1M). The daemon is started by the /etc/rc* scripts during system startup and is responsible for handling both reconfiguration boot processing and updating /dev and /devices in response to dynamic reconfiguration event notifications from the kernel.

For compatibility purposes, drvconfig(1M), disks(1M), tapes(1M), ports(1M), audlinks(1M), and devlinks(1M) are implemented as links to devfsadm.

In addition to managing /dev and /devices, devfsadm also maintains the path_to_inst(4) database.

OPTIONS

The following options are supported:

-C	Cleanup mode. Prompts devfsadm to invoke cleanup routines that are not normally invoked to remove dangling logical links. If -c is also used, devfsadm only cleans up for the listed devices' classes.
-c *device_class*	Restrict operations to devices of class *device_class*. Solaris defines the following values for *device_class*: disk, tape, port, audio, and pseudo. This option may be specified more than once to specify multiple device classes.
-i *driver_name*	Configure only the devices for the named driver, *driver_name*.
-n	Do not attempt to load drivers or add new nodes to the kernel device tree.
-s	Suppress any changes to /dev or /devices. This is useful with the -v option for debugging.
-t *table_file*	Read an alternate devlink.tab file. devfsadm normally reads /etc/devlink.tab.

`-r` *root_dir*	Presume that the `/dev` and `/devices` directory trees are found under *root_dir*, not directly under `root` (`/`). No other use or assumptions are made about *root_dir*.
`-v`	Print changes to `/dev` and `/devices` in verbose mode.

EXIT STATUS

The following exit values are returned:

`0`	Successful completion.
`1`	An error occurred.

FILES

`/devices`	device nodes directory
`/dev`	logical symbolic links to `/devices`
`/usr/lib/devfsadm/devfsadmd`	devfsadm daemon
`/etc/init.d/devfsadm`	daemon start/stop script
`/etc/rcS.d/S50devfsadm`	link to init.d script
`/etc/rc0.d/K83devfsadm`	link to init.d script
`/dev/.devfsadm_dev.lock`	update lock file
`/dev/.devfsadm_daemon.lock`	daemon lock file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`add_drv`(1M), `devlinks`(1M), `disks`(1M), `drvconfig`(1M), `modinfo`(1M), `modload`(1M), `modunload`(1M), `ports`(1M), `rem_drv`(1M), `tapes`(1M), `path_to_inst`(4), `attributes`(5)

NOTES

This document does not constitute an API. The `/devices` directory might not exist or might have different contents or interpretations in a future release. The existence of this notice does not imply that any other documentation that lacks this notice constitutes an API.

NAME

devinfo – print device specific information

SYNOPSIS

/usr/sbin/devinfo -i *device*

/usr/sbin/devinfo -p *device*

DESCRIPTION

The `devinfo` command is used to print device specific information about disk devices on standard out. The command can only be used by the superuser.

OPTIONS

`-i` Prints the following device information:

- Device name
- Software version (not supported and prints as `0`)
- Drive id number (not supported and prints as `0`)
- Device blocks per cylinder
- Device bytes per block
- Number of device partitions with a block size greater than zero

`-p` Prints the following device partition information:

- Device name
- Device major and minor numbers (in hexadecimal)
- Partition start block
- Number of blocks allocated to the partition
- Partition flag
- Partition tag

This command is used by various other commands to obtain device specific information for the making of file systems and determining partition information. If the device cannot be opened, an error message is reported.

OPERANDS

device Device name.

EXIT STATUS

`0` Successful operation.

`2` Operation failed.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`prtvtoc`(1M), `attributes`(5)

NAME

devlinks – adds /dev entries for miscellaneous devices and pseudo-devices

SYNOPSIS

/usr/sbin/devlinks [-d] [-r *rootdir*] [-t *table-file*]

DESCRIPTION

devfsadm(1M) is now the preferred command for /dev and /devices and should be used instead of devlinks.

devlinks creates symbolic links from the /dev directory tree to the actual block- and character-special device nodes under the /devices directory tree. The links are created according to specifications found in the *table-file* (by default /etc/devlink.tab).

devlinks is called each time the system is reconfiguration-booted, and can only be run after drvconfig(1M) is run, since drvconfig(1M) builds the kernel data structures and the /devices tree.

The *table-file* (normally /etc/devlink.tab) is an ASCII file, with one line per record. Comment lines, which must contain a hash character ('#') as their first character, are allowed. Each entry must contain at least two fields, but may contain three fields. Fields are separated by single TAB characters.

The fields are:

devfs-spec

Specification of devinfo nodes that will have links created for them. This specification consists of one or more keyword-value pairs, where the keyword is separated from the value by an equal-sign ('='), and keyword-value pairs are separated from one another by semicolons.

The possible keywords are:

type

The devinfo device type. Possible values are specified in ddi_create_minor_node(9F)

name

The name of the node. This is the portion of the /devices tree entry name that occurs before the first '@' or ':' character.

addr[*n*]

The address portion of a node name. This is the portion of a node name that occurs between the '@' and the ':' characters. It is possible that a node may have a name without an address part, which is the case for many of the pseudo-device nodes. If a number is given after the *addr* it specifies a match of a particular

comma-separated subfield of the address field: *addr1* matches the first subfield, *addr2* matches the second, and so on. *addr0* is the same as *addr* and matches the whole field.

minor[*n*] — The minor portion of a node name – the portion of the name after the ':'. As with *addr* above, a number after the *minor* keyword specifies a subfield to match.

Of these four specifications, only the *type* specification must always be present.

name — Specification of the `/dev` links that correspond to the devinfo nodes. This field allows `devlinks` to determine matching `/dev` names for the `/devices` nodes it has found. The specification of this field uses escape-sequences to allow portions of the `/devices` name to be included in the `/dev` name, *or* to allow a counter to be used in creating node names. If a counter is used to create a name, the portion of the name before the counter must be specified absolutely, and all names in the `/dev/`-subdirectory that match (up to and including the counter) are considered to be subdevices of the same device. This means that they should all point to the same directory, `name` and `address` under the `/devices/`-tree

The possible escape-sequences are:

`\D` — Substitute the `device-name` (name) portion of the corresponding devinfo node-name.

`\A`*n* — Substitute the *n*th component of the `address` component of the corresponding devinfo node name. Sub-components are separated by commas, and sub-component `0` is the whole `address` component.

`\M`*n* — Substitute the *n*th sub-component of the `minor` component of the corresponding devinfo node name. Sub-components are separated by commas, and sub-component `0` is the whole `minor` component.

`\N`*n* — Substitute the value of a 'counter' starting at *n*. There can be only one counter for each dev-spec, and counter-values will be selected so they are as low as possible while not colliding with already-existing link names.

In a dev-spec the `counter` sequence should not be followed by a digit, either explicitly or as a result of another escape-sequence expansion. If this occurs, it would not be possible to correctly match already-existing links to their counter entries, since it would not be possible to unambiguously parse the already-existing `/dev`-name.

extra-dev-link — Optional specification of an extra `/dev` link that points to the initial `/dev` link (specified in field 2). This field may contain a `counter` escape-sequence (as described for the *dev-spec* field) but may not contain any of the other escape-sequences. It provides a way to specify an alias of a particular `/dev` name.

OPTIONS

`-d` — Debugging mode – print out all `devinfo` nodes found, and indicate what links would be created, but do not do anything.

`-r` *rootdir* — Use *rootdir* as the root of the `/dev` and `/devices` directories under which the device nodes and links are created. Changing the root directory does not change the location of the `/etc/devlink.tab` default table, nor is the root directory applied to the filename supplied to the `-t` option.

`-t` *table-file* — Set the table file used by `devlinks` to specify the links that must be created. If this option is not given, `/etc/devlink.tab` is used. This option gives a way to instruct `devlinks` just to perform a particular piece of work, since just the links-types that `devlinks` is supposed to create can be specified in a command-file and fed to `devlinks`.

ERRORS

If `devlinks` finds an error in a line of the *table-file* it prints a warning message on its standard output and goes on to the next line in the *table-file* without performing any of the actions specified by the erroneous rule.

If it cannot create a link for some filesystem-related reason it prints an error-message and continues with the current rule.

If it cannot read necessary data it prints an error message and continues with the next *table-file* line.

EXAMPLES

EXAMPLE 1 Examples of `/etc/devlink.tab` fields

Example `/etc/devlink.tab` fields are:

```
type=pseudo;name=win    win\M0
type=ddi_display    framebuffer/\M0    fb\N0
```

The first example states that all devices of type `pseudo` with a name component of `win` will be linked to `/dev/win`*x*, where *x* is the minor-component of the *devinfo-name* (this is always a single-digit number for the `win` driver).

The second example states that all devinfo nodes of type `ddi_display` will be linked to entries under the `/dev/framebuffer` directory, with names identical to the entire minor component of the `/devices` name. In addition an extra link will be created pointing from `/dev/fb`*n* to the entry under `/dev/framebuffer`. This entry will use a counter to end the name.

FILES

`/dev`	entries for the miscellaneous devices for general use
`/devices`	device nodes
`/etc/devlink.tab`	the default rule-file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`devfsadm`(1M), `ddi_create_minor_node`(9F), `disks`(1M), `drvconfig`(1M), `ports`(1M), `tapes`(1M), `attributes`(5)

BUGS

It is very easy to construct mutually-contradictory link specifications, or specifications that can never be matched. The program does not check for these conditions.

NAME

devnm – device name

SYNOPSIS

/usr/sbin/devnm *name* [*name...*]

DESCRIPTION

The `devnm` command identifies the special file associated with the mounted file system where the argument *name* resides. One or more *name* can be specified.

EXAMPLES

EXAMPLE 1 Using the devnm Command

Assuming that `/usr` is mounted on `/dev/dsk/c0t3d0s6`, the following command :

```
/usr/sbin/devnm /usr
```

produces:

```
/dev/dsk/c0t3d0s6 /usr
```

FILES

`/dev/dsk/*`

`/etc/mnttab`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`mnttab`(4), `attributes`(5)

NAME

devreserv – reserve devices for exclusive use

SYNOPSIS

devreserv [*key* [*device-list...*]]

DESCRIPTION

`devreserv` reserves devices for exclusive use. When the device is no longer required, use `devfree` to release it.

`devreserv` reserves at most one device per *device-list*. Each list is searched in linear order until the first available device is found. If a device cannot be reserved from each list, the entire reservation fails.

When `devreserv` is invoked without arguments, it lists the devices that are currently reserved and shows to which key it was reserved. When `devreserv` is invoked with only the *key* argument, it lists the devices that are currently reserved to that key.

OPERANDS

The following operands are supported:

device-list — Defines a list of devices that `devreserv` will search to find an available device. The list must be formatted as a single argument to the shell.

key — Designates a unique key on which the device will be reserved. The key must be a positive integer.

EXAMPLES

EXAMPLE 1 Reserving a Floppy Disk and a Cartridge Tape

The following example reserves a floppy disk and a cartridge tape:

```
$ key=$$
$ echo "The current Process ID is equal to: $key"
  The Current Process ID is equal to: 10658
$ devreserv $key diskette1 ctape1
```

EXAMPLE 2 Listing All Devices Currently Reserved

The following example lists all devices currently reserved:

```
$ devreserv
  disk1         2423
  diskette1     10658
  ctape1        10658
```

EXAMPLE 3 Listing All Devices Currently Reserved to a Particular Key

The following example lists all devices currently reserved to a particular key:

```
$ devreserv $key
  diskette1
  ctape1
```

EXIT STATUS

The following exit values are returned:

0	Successful completion.
1	Command syntax was incorrect, an invalid was option used, or an internal error occurred.
2	Device table or device reservation table could not be opened for reading.
3	Device reservation request could not be fulfilled.

FILES

`/etc/device.tab`

`/etc/devlkfile`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`devfree`(1M), `attributes`(5)

NOTES

The commands `devreserv` and `devfree` are used to manage the availability of devices on a system. Their use is on a participatory basis and they do not place any constraints on the actual access to the device. They serve as a centralized bookkeeping point for those who wish to use them. Devices which have been reserved cannot be used by processes which utilize the device reservation functions until the reservation has been canceled. However, processes that do not use device reservation may use a device that has been reserved since such a process would not have checked for its reservation status.

NAME

df – displays number of free disk blocks and files

SYNOPSIS

/usr/bin/df [-F *FSType*] [-abeghklntV] [-o *FSType-specific_options*] [*block_device* | *directory* | *file* | *resource* ...]

/usr/xpg4/bin/df [-F *FSType*] [-abeghklnPtV] [-o *FSType-specific_options*] [*block_device* | *directory* | *file* | *resource* ...]

DESCRIPTION

The `df` command displays the amount of disk space occupied by mounted or unmounted file systems, the amount of used and available space, and how much of the file system's total capacity has been used. The file system is specified by device, or by referring to a file or directory on the specified file system.

Used without operands or options, `df` reports on all mounted file systems.

`df` may not be supported for all *FSTypes*.

If `df` is run on a networked mount point that the automounter has not yet mounted, the file system size will be reported as zero. As soon as the automounter mounts the file system, the sizes will be reported correctly.

OPTIONS

The following options are supported for both `/usr/bin/df` and `/usr/xpg4/bin/df`:

`-a` Report on all file systems including ones whose entries in `/etc/mnttab` (see `mnttab`(4)) have the `ignore` option set.

`-b` Print the total number of kilobytes free.

`-e` Print only the number of files free.

`-F` *FSType* Specify the *FSType* on which to operate. The `-F` option is intended for use with unmounted file systems. The *FSType* should be specified here or be determinable from `/etc/vfstab` (see `vfstab`(4)) have the by matching the *directory*, *block_device*, or *resource* with an entry in the table, or by consulting `/etc/default/fs`. See `default_fs`(4).

`-g` Print the entire `statvfs`(2) structure. This option is used only for mounted file systems. It can not be used with the `-o` option. This option overrides the `-b`, `-e`, `-k`, `-n`, `-P`, and `-t` options.

`-h` Like `-k`, except that sizes are in a more human readable format. The output consists of one line of information for each specified file system. This information includes the file system name, the total space allocated in the file system, the amount of space allocated to existing files, the total amount of space available for the creation of new files by unpriviledged users, and the percentage of

normally available space that is currently allocated to all files on the file system. All sizes are scaled to a human readable format, for example, `14K`, `234M`, `2.7G`, or `3.0T`. Scaling is done by repetitively dividing by `1024`.

This option overrides the `-b`, `-e`, `-g`, `-k`, `-n`, `-t`, and `-V` options. This option only works on mounted filesystems and can not be used together with `-o` option.

`-k` Print the allocation in kbytes. The output consists of one line of information for each specified file system. This information includes the file system name, the total space allocated in the file system, the amount of space allocated to existing files, the total amount of space available for the creation of new files by unpriviledged users, and the percentage of normally available space that is currently allocated to all files on the file system. This option overrides the `-b`, `-e`, `-n`, and `-t` options.

`-l` Report on local file systems only. This option is used only for mounted file systems. It can not be used with the `-o` option.

`-n` Print only the *FSType* name. Invoked with no operands, this option prints a list of mounted file system types. This option is used only for mounted file systems. It can not be used with the `-o` option.

`-o` *FSType-specific_options* Specify *FSType-specific* options. These options are comma-separated, with no intervening spaces. See the manual page for the *FSType-specific* command for details.

`-t` Print full listings with totals. This option overrides the `-b`, `-e`, and `-n` options.

`-V` Echo the complete set of file system specific command lines, but do not execute them. The command line is generated by using the options and operands provided by the user and adding to them information derived from `/etc/mnttab`, `/etc/vfstab`, or `/etc/default/fs`. This option may be used to verify and validate the command line.

/usr/xpg4/bin/df

The following option is supported for `/usr/xpg4/bin/df` only:

`-P` Same as `-k` except in 512-byte units.

OPERANDS

`df` interprets operands according to the following precedence: *block_device*, *directory*, *file*. The following operands are supported:

block_device — represents a block special device (for example, `/dev/dsk/c1d0s7`); the corresponding file system need not be mounted.

directory — represents a valid directory name. `df` reports on the file system that contains *directory*.

file — represents a valid file name. `df` reports on the file system that contains *file*.

resource — represents an NFS resource name.

USAGE

See `largefile`(5) for the description of the behavior of `df` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXAMPLES

EXAMPLE 1 Writing Portable Information About the `/usr` File System

The following example writes portable information about the `/usr` file system:

```
example% /usr/xpg4/bin/df -P /usr
```

EXAMPLE 2 Writing Portable Information About the `/usr` File System

Assuming that `/usr/src` is part of the `/usr` file system, the following example writes portable information :

```
example% /usr/xpg4/bin/df -P /usr/src
```

EXAMPLE 3 Using `df` to Display Inode Usage

The following example displays inode usage on all `ufs` file systems:

```
example% /usr/bin/df -F ufs -o i
```

ENVIRONMENT VARIABLES

`SYSV3` — This variable is used to override the default behavior of `df` and provide compatibility with INTERACTIVE UNIX System and SCO UNIX installation scripts. As the `SYSV3` variable is provided for compatibility purposes only, it should not be used in new scripts.

When set, any header which normally displays “files” will now display “nodes”. See `environ`(5) for descriptions of the following environment variables that affect the execution of `df`: `LC_CTYPE`, `LC_MESSAGES`, and `NLSPATH`.

EXIT STATUS

The following exit values are returned:

`0` — Successful completion.

`>0` — An error occurred.

FILES

`/dev/dsk/*`	disk devices
`/etc/default/fs`	default local file system type. Default values can be set for the following flags in `/etc/default/fs`. For example: `LOCAL=ufs`, where `LOCAL` is the default partition for a command if no FSType is specified.
`/etc/mnttab`	mount table
`/etc/vfstab`	list of default parameters for each file system

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

/usr/bin/df

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

/usr/xpg4/bin/df

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWxcu4

SEE ALSO

`find`(1), `df_ufs`(1M), `mount`(1M), `statvfs`(2), `default_fs`(4), `mnttab`(4), `vfstab`(4), `attributes`(5), `environ`(5), `largefile`(5), `XPG4`(5)

NOTES

If UFS logging is enabled on a file system, the disk space used for the log is reflected in the `df` report. The log is allocated from free blocks on the file system, and it is sized approximately `1` Mbyte per `1` Gbyte of file system, up to a maximum of `64` Mbytes.

NAME

dfmounts – display mounted resource information

SYNOPSIS

dfmounts [-F *FSType*] [-h] [-o *specific_options*] [*restriction*...]

DESCRIPTION

dfmounts shows the local resources shared through a distributed file system *FSType* along with a list of clients that have the resource mounted. If *restriction* is not specified, dfmounts shows file systems that are currently shared on any NFS server. *specific_options* as well as the availability and semantics of *restriction* are specific to particular distributed file system types.

If dfmounts is entered without arguments, all remote resources currently mounted on the local system are displayed, regardless of file system type.

dfmounts Output

The output of dfmounts consists of an optional header line (suppressed with the -h flag) followed by a list of lines containing whitespace-separated fields. For each resource, the fields are:

*resource server pathname clients ...*where:

resource	Specifies the resource name that must be given to the mount(1M) command.
server	Specifies the system from which the resource was mounted.
pathname	Specifies the pathname that must be given to the share(1M) command.
clients	Is a comma-separated list of systems that have mounted the resource. Clients are listed in the form *domain.*, *domain.system*, or *system*, depending on the file system type.

A field may be null. Each null field is indicated by a hyphen (–) unless the remainder of the fields on the line are also null; in which case, the hyphen may be omitted.

Fields with whitespace are enclosed in quotation marks (" ").

OPTIONS

-F *FSType*	Specify filesystem type. Defaults to the first entry in /etc/dfs/fstypes. Note: currently the only valid *FSType* is nfs.
-h	Suppress header line in output.
-o *specific_options*	Specify options specific to the filesystem provided by the -F option. Note: currently no options are supported.

FILES

/etc/dfs/fstypes	file system types

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO `dfshares`(1M), `mount`(1M), `share`(1M), `unshare`(1M), `attributes`(5)

NAME

dfmounts_nfs – display mounted NFS resource information

SYNOPSIS

dfmounts [-F nfs] [-h] [*server*...]

DESCRIPTION

dfmounts shows the local resources shared through NFS, along with the list of clients that have mounted the resource. The -F flag may be omitted if NFS is the only file system type listed in the file /etc/dfs/fstypes.

dfmounts without options, displays all remote resources mounted on the local system, regardless of file system type.

The output of dfmounts consists of an optional header line (suppressed with the -h flag) followed by a list of lines containing whitespace-separated fields. For each resource, the fields are:

*resource server pathname clients ...*where

resource Does not apply to NFS. Printed as a hyphen (-).

server Specifies the system from which the resource was mounted.

pathname Specifies the pathname that must be given to the share(1M) command.

clients Is a comma-separated list of systems that have mounted the resource.

OPTIONS

-F nfs Specifies the nfs-FSType.

-h Suppress header line in output.

server Displays information about the resources mounted from each server, where *server* can be any system on the network. If no server is specified, the *server* is assumed to be the local system.

FILES

/etc/dfs/fstypes

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnfscu

SEE ALSO

mount(1M), share(1M), unshare(1M), attributes(5)

NAME

dfshares – list available resources from remote or local systems

SYNOPSIS

dfshares [-F *FSType*] [-h] [-o *specific_options*] [*server*...]

DESCRIPTION

dfshares provides information about resources available to the host through a distributed file system of type *FSType*. *specific_options* as well as the semantics of *server* are specific to particular distributed file systems.

If dfshares is entered without arguments, all resources currently shared on the local system are displayed, regardless of file system type.

The output of dfshares consists of an optional header line (suppressed with the -h flag) followed by a list of lines containing whitespace-separated fields. For each resource, the fields are:

resource server access transport

where

resource	Specifies the resource name that must be given to the mount(1M) command.
server	Specifies the name of the system that is making the resource available.
access	Specifies the access permissions granted to the client systems, either ro (for read-only) or rw (for read/write). If dfshares cannot determine access permissions, a hyphen (–) is displayed.
transport	Specifies the transport provider over which the resource is shared.

A field may be null. Each null field is indicated by a hyphen (–) unless the remainder of the fields on the line are also null; in which case, the hyphen may be omitted.

OPTIONS

-F *FSType*	Specify filesystem type. Defaults to the first entry in /etc/dfs/fstypes.
-h	Suppress header line in output.
-o *specific_options*	Specify options specific to the filesystem provided by the -F option.

FILES

/etc/dfs/fstypes

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

dfmounts(1M), mount(1M), share(1M), unshare(1M), attributes(5)

NAME

dfshares_nfs – list available NFS resources from remote systems

SYNOPSIS

dfshares [-F nfs] [-h] [*server*...]

DESCRIPTION

`dfshares` provides information about resources available to the host through NFS. The `-F` flag may be omitted if NFS is the first file system type listed in the file `/etc/dfs/fstypes`.

The query may be restricted to the output of resources available from one or more servers.

`dfshares` without arguments displays all resources shared on the local system, regardless of file system type.

Specifying *server* displays information about the resources shared by each server. *Server* can be any system on the network. If no server is specified, then *server* is assumed to be the local system.

The output of `dfshares` consists of an optional header line (suppressed with the `-h` flag) followed by a list of lines containing whitespace-separated fields. For each resource, the fields are:

resource server access transport

where

resource	Specifies the resource name that must be given to the `mount`(1M) command.
server	Specifies the system that is making the resource available.
access	Specifies the access permissions granted to the client systems; however, `dfshares` cannot determine this information for an NFS resource and populates the field with a hyphen (-).
transport	Specifies the transport provider over which the *resource* is shared; however, `dfshares` cannot determine this information for an NFS resource and populates the field with a hyphen (-).

A field may be null. Each null field is indicated by a hyphen (-) unless the remainder of the fields on the line are also null; in which case, the hyphen may be omitted.

OPTIONS

`-F nfs`	Specify the NFS file system type
`-h`	Suppress header line in output.

FILES

`/etc/dfs/fstypes`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnfscu

SEE ALSO

`mount`(1M), `share`(1M), `unshare`(1M), `attributes`(5)

NAME

df_ufs – report free disk space on ufs file systems

SYNOPSIS

df `-F ufs [`*generic_options*`] [-o i] [`*directory* | *special*`]`

DESCRIPTION

`df` displays the amount of disk space occupied by `ufs` file systems, the amount of used and available space, and how much of the file system's total capacity has been used.The amount of space reported as used and available is less than the amount of space in the file system; this is because the system reserves a fraction of the space in the file system to allow its file system allocation routines to work well. The amount reserved is typically about 10%; this may be adjusted using `tunefs`(1M). When all the space on the file system except for this reserve is in use, only the superuser can allocate new files and data blocks to existing files. When the file system is overallocated in this way, `df` may report that the file system is more than 100% utilized.If neither *directory* nor *special* is specified, `df` displays information for all mounted ufs file systems.

OPTIONS

The following options are supported:

generic_options Options supported by the generic `df` command. See `df`(1M) for a description of these options.

`-o` Specify `ufs` file system specific options. The available option is:

`i` Report the number of used and free inodes. This option may not be used with *generic_options*.

FILES

`/etc/mnttab` list of file systems currently mounted

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu, SUNWxcu4

SEE ALSO

`df`(1M), `tunefs`(1M), `fs_ufs`(4), `mnttab`(4), `attributes`(5)

NOTES

`df` calculates its results differently for mounted and unmounted file systems. For unmounted systems, the numbers reflect the 10% reservation mentioned above; this reservation is not reflected in `df` output for mounted file systems. For this reason, the available space reported by the generic command may differ from the available space reported by this module.

NAME

dhcpagent – Dynamic Host Configuration Protocol (DHCP) client daemon

SYNOPSIS

dhcpagent [-a] [-d *n*] [-f] [-v]

DESCRIPTION

dhcpagent implements the client half of the Dynamic Host Configuration Protocol (DHCP) for machines running Solaris software.

The dhcpagent daemon obtains configuration parameters for the client (local) machine's network interfaces from a DHCP server. These parameters may include a lease on an IP address, which gives the client machine use of the address for the period of the lease, which may be infinite. If the client wishes to use the IP address for a period longer than the lease, it must negotiate an extension using DHCP. For this reason, dhcpagent must run as a daemon, terminating only when the client machine powers down.

The dhcpagent daemon is controlled through ifconfig(1M) in much the same way that the init(1M) daemon is controlled by telinit(1M). dhcpagent may be invoked as a user process, albeit one requiring root privileges, but this is not necessary, as ifconfig(1M) will start it automatically.

When invoked, dhcpagent enters a passive state while it awaits instructions fromifconfig(1M). When it receives a command to configure an interface, it starts DHCP. Once DHCP is complete, dhcpagent may be queried for the values of the various network parameters. In addition, if DHCP was used to obtain a lease on an address for an interface, the interface is configured and brought up. When a lease is obtained, it is automatically renewed as necessary. If the lease cannot be renewed, dhcpagent will take the interface down at the end of the lease. If the configured interface is found to have a different IP address, subnet mask or broadcast address from those obtained from DHCP, the interface is abandoned from DHCP control.

In addition to DHCP, dhcpagent also supports BOOTP. See *RFC 951, Bootstrap Protocol*. Configuration parameters obtained from a BOOTP server are treated identically to those received from a DHCP server, except that the IP address received from a BOOTP server always has an infinite lease.

DHCP also acts as a mechanism to configure other information needed by the client, for example, the domain name and addresses of routers. Aside from the IP address, netmask, broadcast address and default router, the agent does not directly configure the workstation, but instead acts as a database which may be interrogated by other programs, and in particular by dhcpinfo(1).

On clients with a single interface, this is quite straightforward. Clients with multiple interfaces may present difficulties, as it is possible that some information arriving on different interfaces may need to be merged, or may be inconsistent. Furthermore, the configuration of the interfaces is asynchronous, so requests may arrive while some or all of the interfaces are still unconfigured. To handle these cases, one interface may be designated as primary, which makes it the authoritative source for the values of DHCP parameters in the case where no specific interface is requested. See dhcpinfo(1) and ifconfig(1M) for details.

The dhcpagent daemon can be configured to request a particular host name. See the REQUEST_HOSTNAME description in the FILES section. When first configuring a client to request a host name, you must perform the following steps as root to ensure that the full DHCP negotiation takes place:

```
# pkill dhcpagent
# rm /etc/dhcp/interface.dhc
# reboot
```

Messages

The dhcpagent daemon writes information and error messages in five categories:

critical	Critical messages indicate severe conditions that prevent proper operation.
errors	Error messages are important, sometimes unrecoverable events due to resource exhaustion and other unexpected failure of system calls; ignoring errors may lead to degraded functionality.
warnings	Warnings indicate less severe problems, and in most cases, describe unusual or incorrect datagrams received from servers, or requests for service that cannot be provided.
informational	Informational messages provide key pieces of information that can be useful to debugging a DHCP configuration at a site. Informational messages are generally controlled by the -v option. However, certain critical pieces of information, such as the IP address obtained, are always provided.
debug	Debugging messages, which may be generated at two different levels of verbosity, are chiefly of benefit to persons having access to source code, but may be useful as well in debugging difficult DHCP configuration problems. Debugging messages are only generated when using the -d option.

When dhcpagent is run without the -f option, all messages are sent to the system logger syslog(3C) at the appropriate matching priority and with a facility identifier LOG_DAEMON. When dhcpagent is run with the -f option, all messages are directed to standard error.

OPTIONS

The following options are supported:

-a	Adopt a configured interface. This option is for use with diskless DHCP clients. In the case of diskless DHCP, DHCP has already been performed on the network interface providing the operating system image prior to running dhcpagent. This option instructs the agent to take over control of the interface. It is intended primarily for use in boot scripts.
-d *n*	Set debug level to *n*. Two levels of debugging are currently available, 1 and 2; the latter is more verbose.
-f	Run in the foreground instead of as a daemon process. When this option is used, messages are sent to standard error instead of to syslog(3C).

`-v` Provide verbose output useful for debugging site configuration problems.

FILES

`/etc/dhcp/`*if*`.dhc`
Contains the configuration for interface. The mere existence of this file does not imply that the configuration is correct, since the lease may have expired.

`/etc/default/dhcpagent`
Contains default values for tunable parameters. All values may be qualified with the interface they apply to by prepending the interface name and a period (".") to the interface parameter name. The parameters include:

`RELEASE_ON_SIGTERM`
Indicates that a `RELEASE` rather than a `DROP` should be performed on managed interfaces when the agent terminates.

`OFFER_WAIT`
Indicates how long to wait between checking for valid `OFFER`s after sending a `DISCOVER`.

`ARP_WAIT`
Indicates how long to wait for clients to respond to an `ARP` request before concluding the address in the `ARP` request is unused.

`IGNORE_FAILED_ARP`
Specifies whether or not the agent should assume an address is available, in the unlikely event that `ARP` cannot be performed on that address.

`CLIENT_ID`
Indicates the value that should be used to uniquely identify the client to the server.

`PARAM_REQUEST_LIST`
Specifies a list of comma-separated integer values of options for which the client would like values.

`REQUEST_HOSTNAME`
Indicates the client requests the DHCP server to map the client's leased IP address to the host name associated with the network interface that performs DHCP on the client. The host name must be specified in the `/etc/hostname.`*interface* file for the relevant interface on a line of the form

`inet` *hostname*
where *hostname* is the host name requested.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsr
Interface Stability	Evolving

SEE ALSO `dhcpinfo`(1), `ifconfig`(1M), `init`(1M), `syslog`(3C), `attributes`(5), `dhcp`(5)

System Administration Guide: IP Services

Croft, B. and Gilmore, J.,*Bootstrap Protocol (BOOTP)*RFC 951, Network Working Group, September 1985.

Droms, R., *Dynamic Host Configuration Protocol*, RFC 2131, Network Working Group, March 1997.

NOTES Currently, configurations where more than one interface is attached to the same physical network are unsupported. This precludes use of virtual interfaces.

NAME

dhcpconfig – DHCP service configuration utility

SYNOPSIS

dhcpconfig -D -r *resource* -p *path* [-u *uninterpreted*] [-l *lease_length*] [-n] [-d *DNS_domain*] [-a *DNS_server_addresses*] [-h *hosts_resource*] [-y *hosts_domain*]

dhcpconfig -R *server_addresses*

dhcpconfig -U [-f] [-x] [-h]

dhcpconfig -N *network_address* [-m *subnet_mask*] [-b] [-t *router_addresses*] [-y *NIS-domain*] [-a *NIS_server_addresses*]

dhcpconfig -C -r *resource* -p *path* [-f] [-k] [-u *uninterpreted*]

dhcpconfig -X *filename* [-m *macro_list*] [-o *option_list*] [-a *network_addresses*] [-f] [-x]

dhcpconfig -I *filename* [-f]

DESCRIPTION

The dhcpconfig command is used to configure and manage the Dynamic Host Configuration Protocol (DHCP) service or BOOTP relay services. It is intended for use by experienced Solaris system administrators and is designed for ease of use in scripts. The dhcpmgr utility is recommended for less experienced administrators or those preferring a graphical utility to configure and manage the DHCP service or BOOTP relay service.

The dhcpconfig command can be run by root, or by other users assigned to the DHCP Management profile. See rbac(5) and user_attr(4).

dhcpconfig requires one of the following function flags: -D, -R, -U, -N, -C, -X, or -I.

The dhcpconfig menu driven mode is supported in Solaris 8 and previous versions of Solaris.

Where dhcpconfig Obtains Configuration Information

dhcpconfig scans various configuration files on your Solaris machine for information it can use to assign values to options contained in macros it adds to the dhcptab configuration table. The following table lists information dhcpconfig needs, the source used, and how the information is used:

Information	*Source*	*Where Used*
Timezone	System date, timezone settings	Locale macro
DNS parameters	nsswitch.conf, /etc/resolv.conf	Server macro
NIS parameters	System domainname, nsswitch.conf, NIS	Network macros

Subnetmask	Network interface, `netmasks` table in nameservice	Network macros

If you have not set these parameters on your server machine, you should do so before configuring the DHCP server with `dhcpconfig`. Note that if you specify options with the `dhcpconfig -D` command line, the values you supply override the values obtained from the system files.

OPTIONS

The following options are supported:

`-C`

Convert to using a new data store, recreating the DHCP data tables in a format appropriate to the new data store, and setting up the DHCP server to use the new data store.

The following sub-options are required:

`-p` *path_to_data*

The paths for `SUNWfiles` and `SUNWbinfiles` must be absolute UNIX pathnames. The path for `SUNWnisplus` must be a fully specified NIS+ directory (including the tailing period.) See `dhcp_modules`(5).

`-r` *data_resource*

New data store resource. One of the following must be specified: `SUNWfiles`, `SUNWbinfiles`, or `SUNWnisplus`. See `dhcp_modules`(5).

The following sub-options are optional:

`-f`

Do not prompt for confirmation. If `-f` is not used, a warning and confirmation prompt are issued before the conversion starts.

`-k`

Keep the old DHCP data tables after successful conversion. If any problem occurs during conversion, tables will not be deleted even if `-k` sub-option is not specified.

`-u` *uninterpreted*

Data which will be ignored by `dhcpconfig`, but passed on to the datastore for interpretation. This might be used for a database account name or other authentication or authorization

parameters required by a particular data store. The `-u` sub-option is not used with the `SUNWfiles`, `SUNWbinfiles`, and `SUNWnisplus` data stores. See `dhcp_modules`(5).

`-D` Configure the DHCP service.

The following sub-options are required:

`-r` *data_resource*
One of the following must be specified: `SUNWfiles`, `SUNWbinfiles`, or `SUNWnisplus`. Other data stores may be available.See `dhcp_modules`(5).

`-p` *path*
The paths for `SUNWfiles` and `SUNWbinfiles` must be absolute UNIX pathnames. The path for `SUNWnisplus` must be a fully specified NIS+ directory (including the tailing period.) . See `dhcp_modules`(5).

The following sub-options are optional:

`-a` *DNS_servers*
IP addresses of DNS servers, separated with commas.

`-d` *DNS_domain*
DNS domain name.

`-h` *hosts_resource*
Resource in which to place hosts data. Usually, the name service in use on the server. Valid values are `nisplus`, `files`, or `dns`.

`-l` *seconds*
Lease length used for addresses not having a specified lease length, in seconds.

`-n`
Non-negotiable leases

`-y` *hosts_domain*
DNS or NIS+ domain name to be used for hosts data. Valid only if `dns` or `nisplus` is specified for `-h` sub-option.

`-u` *uninterpreted*
Data which will be ignored by `dhcpconfig`, but passed on to the datastore for interpretation. This might be used for a database account name or other

authentication or authorization parameters required by a particular data store. The `-u` sub-option is not used with the `SUNWfiles`, `SUNWbinfiles`, and `SUNWnisplus` data stores. See `dhcp_modules`(5).

`-I` *filename*

Import data from *filename*, containing data previously exported from a Solaris DHCP server. Note that after importing, you may have to edit macros to specify the correct domain names, and edit network tables to change the owning server of addresses in imported networks. Use `dhtadm` and `pntadm` to do this.

The following sub-option is supported:

`-f` Replace any conflicting data with the data being imported.

`-N` *net_address*

Configure an additional network for DHCP service.

The following sub-options are supported:

`-a` *NIS_server_addresses*	List of IP addresses of NIS servers.
`-b`	Network is a point-to-point (PPP) network, therefore no broadcast address should be configured. If `-b` is not used, the network is assumed to be a LAN, and the broadcast address is determined using the network address and subnet mask.
`-m` *xxx.xxx.xxx.xxx*	Subnet mask for the network; if `-m` is not used, subnet mask is obtained from netmasks.
`-t` *router_addresses*	List of router IP addresses; if not specified, router discovery flag will be set.
`-y` *NIS_domain_name*	If NIS is used on this network, specify the NIS domain name.

`-R` *server_addresses*

Configure the BOOTP relay service. BOOTP or DHCP requests are forwarded to the list of servers specified.

server_addresses is a comma separated list of hostnames and/or IP addresses.

`-U` Unconfigure the DHCP service or BOOTP relay service.

The following sub-options are supported:

`-f` Do not prompt for confirmation. If `-f` is not used, a warning and confirmation prompt is issued.

`-h` Delete hosts entries from name service.

`-x` Delete the `dhcptab` and network tables.

`-X` *filename* Export data from the DHCP data tables, saving to *filename*, to move the data to another Solaris DHCP server.

The following sub-options are optional:

`-a` *networks_to_export* List of networks whose addresses should be exported, or the keyword ALL to specify all networks. If `-a` is not specified, no networks are exported.

`-m` *macros_to_export* List of macros to export, or the keyword `ALL` to specify all macros. If `-m` is not specified, no macros are exported.

`-o` *options_to_export* List of options to export, or the keyword `ALL` to specify all options. If `-o` is not specified, no options are exported.

`-x` Delete the data from this server after it is exported. If `-x` is not specified you are in effect copying the data.

EXAMPLES

EXAMPLE 1 Configuring DHCP Service with Binary Files Data Store

The following command configures DHCP service, using the binary files data store, in the DNS domain `acme.eng`, with a lease time of 28800 seconds (8 hours),

```
example# dhcpconfig -D -r SUNWbinfiles -p /var/dhcp -l 28800 -d acme.eng
     -a 120.30.33.4 -h dns -y acme.eng
```

EXAMPLE 2 Configuring BOOTP Relay Agent

The following command configures the DHCP daemon as a BOOTP relay agent, which will forward BOOTP and DHCP requests to the servers having the IP addresses 120.30.33.7 and 120.30.42.132:

```
example# dhcpconfig -R 120.30.33.7,120.30.42.132
```

EXAMPLE 3 Unconfiguring DHCP Service

The following command unconfigures the DHCP service, with confirmation, and deletes the DHCP data tables and host table entries:

```
example# dhcpconfig -U -x -h
```

EXAMPLE 4 Configuring a Network for DHCP Service

The following command configures an additional LAN network for DHCP service, specifying that clients should use router discovery and providing the NIS domain name and NIS server address:

```
example# dhcpconfig -N 120.30.171.0 -y east.acme.eng.com -a 120.30.33.4
```

EXAMPLE 5 Converting to SUNWnisplus Data Store

The following command converts a DHCP server from using a text or binary files data store to a NIS+ data store, deleting the old data store's DHCP tables:

```
example# dhcpconfig -C -r SUNWnisplus -p whatever.com.
```

EXAMPLE 6 Exporting a Network, Macros, and Options from a DHCP Server

The following command exports one network (`120.30.171.0`) and its addresses, the macro `120.30.171.0`, and the options `motd` and `PSptr`from a DHCP server, saves the exported data in file `/export/var/120301710_data`, and deletes the exported data from the server.

```
example# dhcpconfig -X /var/dhcp/120301710_export
     -a 120.30.171.0 -m 120.30.171.0 -o motd,PSptr
```

EXAMPLE 7 Importing Data on a DHCP Server

The following command imports DHCP data from a file, `/net/golduck/export/var/120301710_data`, containing data previously exported from a Solaris DHCP server, and overwrites any conflicting data on the importing server:

```
example# dhcpconfig -I /net/golduck/export/var/120301710_data -f
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWdhcsu
Interface Stability	Evolving

SEE ALSO

`dhcpmgr`(1M), `dhtadm`(1M), `in.dhcpd`(1M), `pntadm`(1M), `dhcp_network`(4), `dhcptab`(4), `dhcpsvc.conf`(4), `nsswitch.conf`(4), `resolv.conf`(4), `user_attr`(4), `attributes`(5), `dhcp`(5), `dhcp_modules`(5), `rbac`(5)

System Administration Guide: IP Services

NAME

dhcpmgr – graphical interface for managing DHCP service

SYNOPSIS

/usr/sadm/admin/bin/dhcpmgr

DESCRIPTION

`dhcpmgr` is a graphical user interface which enables you to manage the Dynamic Host Configuration Protocol (DHCP) service on the local system. It performs the functions of the `dhcpconfig`, `dhtadm`, and `pntadm` command line utilities. You must be `root` to use `dhcpmgr`. The `dhcpmgr` Help, available from the `Help` menu, contains detailed information about using the tool.

USAGE

You can perform the following tasks using `dhcpmgr`:

Configure DHCP service
Use `dhcpmgr` to configure the DHCP daemon as a DHCP server, and select the data store to use for storing network configuration tables..

Configure BOOTP relay service
Use `dhcpmgr` to configure the DHCP daemon as a BOOTP relay.

Manage DHCP or BOOTP relay service
Use `dhcpmgr` to start, stop, enable, disable or unconfigure the DHCP service or BOOTP relay service, or change DHCP server parameters.

Manage DHCP addresses
Use `dhcpmgr` to add, modify, or delete IP addresses leased by the DHCP service.

Manage DHCP macros
Use `dhcpmgr` to add, modify or delete macros used to supply configuration parameters to DHCP clients.

Manage DHCP options
Use `dhcpmgr` to add, modify or delete options used to define parameters deliverable through DHCP.

Convert to a new DHCP data store
Use `dhcpmgr` to configure the DHCP server to use a different data store, and convert the DHCP data to the format used by the new data store.

Move DHCP data to another server
Use `dhcpmgr` to export data from one Solaris DHCP server and import data onto another Solaris DHCP server.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`non-zero` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE

Availability	SUNWdhcm
Interface Stability	Evolving

SEE ALSO

`dhcpconfig`(1M), `dhtadm`(1M), `pntadm`(1M), `in.dhcpd`(1M), `dhcpsvc.conf`(4), `dhcp_network`(4), `dhcptab`(4), `attributes`(5), `dhcp`(5), `dhcp_modules`(5)

Solaris DHCP Service Developer's Guide

System Administration Guide: IP Services

NAME

dhtadm – DHCP configuration table management utility

SYNOPSIS

dhtadm -C [-r *resource*] [-p *path*] [-u *uninterpreted*]

dhtadm -A -s *symbol_name* -d *definition* [-r *resource*] [-p *path*] [-u *uninterpreted*]

dhtadm -A -m *macro_name* -d *definition* [-r *resource*] [-p *path*] [-u *uninterpreted*]

dhtadm -M -s *symbol_name* -d *definition* [-r *resource*] [-p *path*] [-u *uninterpreted*]

dhtadm -M -s *symbol_name* -n *new_name* [-r *resource*] [-p *path*] [-u *uninterpreted*]

dhtadm -M -m *macro_name* -n *new_name* [-r *resource*] [-p *path*] [-u *uninterpreted*]

dhtadm -M -m *macro_name* -d *definition* [-r *resource*] [-p *path*] [-u *uninterpreted*]

dhtadm -M -m *macro_name* -e *symbol=value* [-r *resource*] [-p *path*] [-u *uninterpreted*]

dhtadm -D -s *symbol_name* [-r *resource*] [-p *path*] [-u *uninterpreted*]

dhtadm -D -m *macro_name* [-r *resource*] [-p *path*] [-u *uninterpreted*]

dhtadm -P [-r *resource*] [-p *path*] [-u *uninterpreted*]

dhtadm -R [-r *resource*] [-p *path*] [-u *uninterpreted*]

dhtadm -B [-v] [*batchfile*]

DESCRIPTION

`dhtadm` manages the Dynamic Host Configuration Protocol (DHCP) service configuration table, `dhcptab`. You can use it to add, delete, or modify DHCP configuration macros or options or view the table. For a description of the table format, see `dhcptab`(4).)

The `dhtadm` command can be run by root, or by other users assigned to the DHCP Management profile. See `rbac`(5) and `user_attr`(4).

After you make changes with `dhtadm`, you should issue a `SIGHUP` to the DHCP server, causing it to read the `dhcptab` and pick up the changes. Do this using the command using the `pkill -HUP in.dhcpd` command. See `in.dhcpd`(1M).

OPTIONS

One of the following function flags must be specified with the `dhtadm` command: `-A`, `-B`, `-C`, `-D`, `-M`, `-P` or `-R`.

The following options are supported:

`-A` Add a symbol or macro definition to the `dhcptab` table.

The following sub-options are required:

`-d` *definition*
Specify a macro or symbol definition.

definition must be enclosed in single quotation marks. For macros, use the form `-d '`*symbol*`=`*value*`:`*symbol*`=`*value*`:'`. For symbols, the definition is a series of fields that define a symbol's characteristics. The fields are separated by commas. Use the form `-d '`*context*`,`*code*`,`*type*`,`*granularity*`,`*maximum*`'`. See `dhcptab`(4) for information about these fields.

`-m` *macro_name*
Specify the name of the macro to be added.

The `-d` option must be used with the `-m` option. The `-s` option cannot be used with the `-m` option.

`-s` *symbol_name*
Specify the name of the symbol to be added.

The `-d` option must be used with the `-s` option. The `-m` option cannot be used with the `-s` option.

`-B`
Batch process `dhtadm` commands. `dhtadm` will read from the specified file or from standard input a series of `dhtadm` commands and execute them within the same process. Processing many `dhtadm` commands using this method is much faster than running an executable batchfile itself. Batch mode is recommended for using `dhtadm` in scripts.

The following sub-option is optional:

`-v`
Display commands to standard output as they are processed.

`-C`
Create the DHCP service configuration table, `dhcptab`.

`-D`
Delete a symbol or macro definition.

The following sub-options are required:

`-m` *macro_name*
Delete the specified macro.

`-s` *symbol_name*
Delete the specified symbol.

`-M`
Modify an existing symbol or macro definition.

The following sub-options are required:

`-d` *definition*
Specify a macro or symbol definition to modify.

The definition must be enclosed in single quotation marks. For macros, use the form `-d '`*symbol*`=`*value*`:`*symbol*`=`*value*`:'`. For symbols, the definition is a series of fields that define a symbol's characteristics. The fields are separated by commas. Use the form `-d '`*context*`,`*code*`,`*type*`,`*granularity*`,`*maximum*`'`. See `dhcptab`(4) for information about these fields.

`-e`
This sub-option uses the *symbol =value* argument. Use it to edit a *symbol/value* pair within a macro. To add a symbol which does not have an associate value, enter:

symbol`=NULL_VALUE`_To delete a symbol definition from a macro, enter:

symbol=

`-m`
This sub-option uses the *macro_name* argument. The `-n`, `-d`, or `-e` sub-options are legal companions for this sub-option..

`-n`
This sub-option uses the *new_name* argument and modifies the name of the object specified by the `-m` or `-s` sub-option. It is not limited to macros. . Use it to specify a new macro name or symbol name.

`-s`
This sub-option uses the *symbol_name* argument. Use it to specify a symbol. The `-d` sub-option is a legal companion.

`-p` *path*
Override the `dhcpsvc.conf`(4) configuration value for `PATH=` with *path*. See `dhcpsvc.conf`(4) for more details regarding *path*. See `dhcp_modules`(5) for information regarding data storage modules for the DHCP service.

`-P`
Print (display) the `dhcptab` table.

`-r` *data_store_resource*
Override the `dhcpsvc.conf`(4) configuration value for `RESOURCE=` with the *data_store_resource* specified. See `dhcpsvc.conf`(4) for more details on resource type. See*Solaris DHCP Service Developer's Guide* for

	more information about adding support for other data stores. See `dhcp_modules`(5) for information regarding data storage modules for the DHCP service.
`-R`	Remove the `dhcptab` table.
`-u` *uninterpreted*	Data which will be ignored by `dhtadm`, but passed to currently configured public module, to be interpreted by the data store. This might be used for a database account name or other authentication or authorization parameters required by a particular data store. Uninterpreted data is stored within `RESOURCE_CONFIG` keyword of `dhcpsvc.conf`(4). See `dhcp_modules`(5) for information regarding data storage modules for the DHCP service.

EXAMPLES

EXAMPLE 1 Creating the DHCP Service Configuration Table

The following command creates the DHCP service configuration table, `dhcptab`:

```
# dhtadm -C
```

EXAMPLE 2 Adding a Symbol Definition

The following command adds a `Vendor` option symbol definition for a new symbol called `MySym` to the `dhcptab` table in the `SUNWfiles` resource in the `/var/mydhcp` directory:

```
# dhtadm -A -s MySym
     -d 'Vendor=SUNW.PCW.LAN,20,IP,1,0'
          -r SUNWfiles -p /var/mydhcp
```

EXAMPLE 3 Adding a Macro Definition

The following command adds the `aruba` macro definition to the `dhcptab` table. Note that symbol/value pairs are bracketed with colons (`:`).

```
# dhtadm -A -m aruba
     -d ':Timeserv=10.0.0.10 10.0.0.11:DNSserv=10.0.0.1:'
```

EXAMPLE 4 Modifying a Macro Definition

The following command modifies the `Locale` macro definition, setting the value of the `UTCOffst` symbol to 18000 seconds. Note that any macro definition which includes the definition of the `Locale` macro will inherit this change.

```
# dhtadm -M -m Locale -e 'UTCOffst=18000'
```

EXAMPLE 4 Modifying a Macro Definition *(Continued)*

EXAMPLE 5 Deleting a Symbol

The following command deletes the `Timeserv` symbol from the `aruba` macro. Note that any macro definition which includes the definition of the `aruba` macro will inherit this change.

```
# dhtadm -M -m aruba -e 'Timeserv='
```

EXAMPLE 6 Adding a Symbol to a Macro

The following command adds the `Hostname` symbol to the `aruba` macro. Note that the `Hostname` symbol takes no value, and thus requires the special value `_NULL_VALUE_`. Note also that any macro definition which includes the definition of the `aruba` macro will inherit this change.

```
# dhtadm -M -m aruba -e 'Hostname=_NULL_VALUE_'
```

EXAMPLE 7 Renaming a Macro

The following command renames the `Locale` macro to `MyLocale`. Note that any `Include` statements in macro definitions which include the `Locale` macro will also need to be changed.

```
# dhtadm -M -m Locale -n MyLocale
```

EXAMPLE 8 Deleting a Symbol Definition

The following command deletes the `MySym` symbol definition. Note that any macro definitions which use `MySym` will need to be modified.

```
# dhtadm -D -s MySym
```

EXAMPLE 9 Removing a dhcptab

The following command removes the `dhcptab` table in the NIS+ directory specified.

```
# dhtadm -R -r SUNWnisplus -p Test.Nis.Plus.
```

EXAMPLE 10 Printing a dhcptab

The following command prints to standard output the contents of the `dhcptab` that is located in the data store and path indicated in the `dhcpsvc.conf` file:.

```
# dhtadm -P
```

EXAMPLE 10 Printing a dhcptab *(Continued)*

EXAMPLE 11 Executing dhtadm in Batch Mode

The following command runs a series of `dhtadm` commands contained in a batch file:

```
# dhtadm -B addmacros
```

EXIT STATUS

`0`	Successful completion.
`1`	Object already exists.
`2`	Object does not exist.
`3`	Non-critical error.
`4`	Critical error.

FILES

`/etc/inet/dhcpsvc.conf`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWdhcsu
Interface Stability	Evolving

SEE ALSO

`dhcpconfig`(1M), `dhcpmgr`(1M), `in.dhcpd`(1M), `dhcpsvc.conf`(4), `dhcp_network`(4), `dhcptab`(4), `hosts`(4), `user_attr`(4), `attributes`(5), `dhcp`(5), `dhcp_modules`(5)`rbac`(5)

Solaris DHCP Service Developer's Guide

System Administration Guide: IP Services

Alexander, S., and R. Droms, *DHCP Options and BOOTP Vendor Extensions*, RFC 1533, Lachman Technology, Inc., Bucknell University, October 1993.

Droms, R., *Interoperation Between DHCP and BOOTP*, RFC 1534, Bucknell University, October 1993.

Droms, R., *Dynamic Host Configuration Protocol*, RFC 1541, Bucknell University, October 1993.

Wimer, W., *Clarifications and Extensions for the Bootstrap Protocol*, RFC 1542, Carnegie Mellon University, October 1993.

NAME

dig – send domain name query packets to name servers

SYNOPSIS

```
dig [@server] domain [query-type] [query-class] [+query-option] [-dig-option]
    [%ignored-comment]
```

DESCRIPTION

Use `dig` ("domain information groper") to gather information from the Domain Name System ("DNS") servers. `dig` has two modes, simple interactive mode for a single query, and batch mode, which executes a query for each line in a list of several query lines. All query options are accessible from the command line.

OPTIONS

The `dig` utility supports the following options:

`@server`

Either a domain name or a dot-notation Internet address. If this field is omitted, `dig` attempts to use the default name server for the machine. If a domain name is specified, this will be resolved using the domain name system resolver, for example, BIND. If the system does not support DNS, specify a dot-notation address. Alternatively, `/etc/resolv.conf` should be present. It indicates where the default name servers reside, so that server itself can be resolved. See `resolver`(3RESOLV) for information on `/etc/resolv.conf`. As an option, set the environment variable `LOCALRES` to name a file which is to be used instead of the `/etc/resolv.conf` standard resolver. `LOCALRES` is specific to the `dig` resolver and is not referenced by the system resolver. If the `LOCALRES` variable is not set or the specified file is not readable, then `/etc/resolv.conf` will be used.

`-domain`

The domain name for which you are requesting information. See the `-x` option for a convenient way to specify an inverse address query.

`query-type`

The type of information (DNS query type) that you are requesting. If omitted, the default is `a` (`T_A=address`). The following types are recognized:

`a`	`T_A`	network address
`any`	`T_ANY`	any and all information about specified domain
`mx`	`T_MX`	mail exchanger for the domain
`ns`	`T_NS`	name servers

soa	T_SOA	zone of authority record
hinfo	T_HINFO	host information
axfr	T_AXFR	zone transfer (must ask an authoritative server)
txt	T_TXT	arbitrary number of strings

See *RFC 1035* for a complete list of values for `query-type`.

`query-class` The network class requested in the query. If omitted, the default is `in` (`C_IN=Internet`). The following classes are recognized:

in	C_IN	Internet class domain
any	C_ANY	any and all class information

See *RFC 1035* for a complete list of values for `query-class`.

`any` can be used to specify a class and a type of query. `dig` parses the first occurrence of `any` to mean `query-type=T_ANY`. To specify `query-class=C_ANY`, either specify `any` twice, or set `query-class` using the `-c` option.

`%ignored-comment` "%" is used to include an argument that is not parsed. This is useful when running `dig` in batch mode. For example:

```
example% dig @128.9.0.32 %venera.isi.edu mx isi.edu
```

-dig option "-" is used to specify an option that affects the operation of `dig`. The following options are currently available:

`-x` *dot-notation-address*
Specify inverse address mapping. Instead of:

```
example% dig 32.0.9.128.in-addr.arpa
```

Specify:

```
example% dig -x 128.9.0.32
```

-f *file*
Batch mode. *file* contains a list of query specifications, that is, `dig` command lines, which are to be executed successively. Lines that begin with ';', '#', or '\n' are ignored. Other options may still appear on command line that will be in effect for each batch query.

-T *time*
Specify the time in seconds between the start of successive queries in batch mode. This option can be used synchronize two or more batch `dig` commands. The default is zero.

-p *port*
Specify port number. This option allows you to query a name server that listens to a non-standard port number. The default is 53.

-P [*ping-string*]
After query returns, execute a `ping`(1M) command for response time comparison. This option makes a call to the shell. The last three ,lines of statistics are printed for the command:

```
example % ping -s -server_name -56 -3
```

If the optional *ping_string* is present, it replaces `ping -s` in the shell command.

-t *query-type*
Specify type of query. You may specify either an integer value to be included in the type field, or use the abbreviated mnemonic. for example, `mx = T_MX`.

-c *query-class*
Specify class of query. You may specify either an integer value to be included in the class field, or use the abbreviated mnemonic, for example, `in = C_IN`.

-k *keydir:keyname*
Sign the query with the `TSIG` key named *keyname* that is in the directory *keydir*.

-envsav
Specifies that after all of the arguments are parsed, the `dig` environment should be saved to a file to become the default environment. This is useful to

bypass the standard set of defaults and use a custom set of options each time `dig` is used. The environment consists of resolver state variable flags, timeout, and retries as well as the flags detailing `dig` output. If the shell environment variable `LOCALDEF` is set to the name of a file, this is where the default `dig` environment is saved. If not, the file `DiG.env` is created in the current working directory.

`LOCALDEF` is specific to the dig resolver, and will not affect operation of the standard `resolver()` library.

Each time dig `is` executed, it looks for `./DiG.env` or the file specified by the shell environment variable `LOCALDEF`. If such file exists and is readable, then the environment is restored from this file before any arguments are parsed. The `DiG.env` file contains binary data and should not be modified directly.

`-envset`
Specifies that after the arguments are parsed, the `dig` environment becomes the default environment for the duration of the batch file, or until the next line that specifies `-envset`. This flag is set by including it in a line in a `dig` batch file. It only affects batch query runs.

`- [no] stick`
Specifies that the `dig` environment, either as read initially or set by the `-envset` option, is to be restored before each query line in a `dig` batch file. The default `-nostick` means that the `dig` environment does not stick. Hence, options specified on a single line in a `dig` batch file will remain in effect for subsequent lines, that is, they are not restored to the "sticky" default. This option only affects batch query runs.

+query-option

"+" is used to specify an option to be changed in the query packet or to change `dig` output specifics. Many of these are the same parameters accepted by `nslookup`(1M). If an option requires a parameter, the form is as follows:

```
+ keyword [=value]
```

Most keywords can be abbreviated. The parsing of the "+" options is very simplistic. A value must not be separated from its keyword by white space. The following keywords are currently available:

Keyword	Abbreviation	Meaning [default]
`[no] debug`	`[deb]`	Turn on or off debugging mode `[deb]`
`[no] d2`		Turn on or off extra debugging mode `[nod2]`
`[no] recurse`	[rec]	Use or do not use recursive lookup `[rec]`
`retry=#`	[ret]	Set number of retries to # [4]
`time=#`	[ti]	Set timeout length to # seconds [4]
`[no] ko`		Keep open option. Implies `vc`. `[noko]`
`[no] vc`		Use or do not use virtual circuit `[novc]`
`[no] defname`	`[def]`	Use or do not use default domain name `[def]`
`[no] search`	`[sea]`	Use or do not use domain search list `[sea]`
`domain=NAME`	`[do]`	Set default domain name to `NAME`
`[no] ignore`	`[i]`	Ignore or do not ignore truncated errors `[noi]`
`[no] primary`	`[pr]`	Use or do not use primary server `[nopr]`

Keyword	Abbreviation	Meaning [default]
[no] aaonly	[aa]	Authoritative query only flag [noaa]
[no] cmd		Echo parsed arguments [cmd]
[no] stats	[st]	Print query statistics [st]
[no] Header	[H]	Print basic header [H]
[no] header	[he]	Print header flags [he]
[no] ttlid	[tt]	Print TTLs [tt]
[no] cl		Print class info [nocl]
[no] qr		Print outgoing query [noqr]
[no] reply	[rep]	Print reply [rep]
[no] ques	[qu]	Print question section [qu]
[no] answer	[an]	Print answer section [an]
[no] author	[au]	Print authoritative section [au]
[no] addit	[ad]	Print additional section [ad]
pfdef		Set to default print flags
pfmin		Set to minimal default print flags
pfset=#		Set print flags to #. The value of # can be hex, octal, or decimal.
pfand=#		Bitwise and print flags with #

Keyword	Abbreviation	Meaning [default]
pfor=#		Bitwise or print flags with #

The retry and time options affect the retransmission strategy used by the resolver library() when sending datagram queries. The algorithm is as follows:

```
for i = 0 to retry - 1
    for j = 1 to num_servers
        send_query
        wait((time * (2**i)) / num_servers)
    end
end
```

dig always uses a value of 1 for num_servers.

ENVIRONMENT VARIABLES

LOCALRES File to use in place of /etc/resolv.conf

LOCALDEF default environment file

FILES

/etc/resolv.conf Initial domain name and name server addresses

./DiG.env Default save file for default options

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu
Interface Stability	External

SEE ALSO

in.named(1M), nslookup(1M), resolver(3RESOLV), attributes(5)

Mockapetris, Paul. *RFC 1035, Domain Names - Implementation and Specification*. Network Working Group. November 1987.

BUGS

dig does not consistently exit with appropriate status messages when a problem occurs somewhere in the resolver(), although most of the common exit cases are handled. This can be problematic when running in batch mode. If dig exits abnormally and is not caught, the entire batch aborts. When such an event is trapped, dig simply continues with the next query.

NAME

directoryserver – front end for the Directory Server (DS)

SYNOPSIS

```
/usr/sbin/directoryserver { setup [-f configuration_file] | uninstall}

/usr/sbin/directoryserver {start-admin | stop-admin |
    restart-admin | startconsole}

/usr/sbin/directoryserver [{-s | -server} server-instance ]{start
    | stop | restart}

/usr/sbin/directoryserver { -s | -server } server-instance {
    monitor | saveconfig | restoreconfig | db2index-task |
    ldif2db-task | ldif2db | ldif2ldap | vlvindex | db2ldif |
    db2ldif-task | db2bak | db2bak-task | bak2db | bak2db-task |
    suffix2instance | account-status | account-activate |
    account-inactivate } {...}

/usr/sbin/directoryserver nativetoascii | admin_ip | ldif |
    pwdhash | idsktune | mmldif | keyupg {...}

/usr/sbin/directoryserver { magt | sagt } {...}

/usr/sbin/directoryserver help [subcommand]
```

DESCRIPTION

The `directoryserver` command is a comprehensive, front end to the utility programs provided by the Solaris Directory Server (DS).

Options for the `directoryserver` command itself must appear before the subcommand. Arguments for a subcommand must appear after the subcommand. Subcommands have specific arguments. See `SUBCOMMANDS`.

SUBCOMMANDS

The following subcommands are supported:

account-inactivate *args*

Inactivates and locks an entry or group of entries.

The `account-inactivate` subcommand supports the following arguments:

[`-D` *rootdn*]
Directory Server *userDN* with root permissions, such as Directory Manager.

[`-h` *host*]
Host name of Directory Server. The default value is the full hostname of the machine where Directory Server is installed.

`-I` *DN*
Entry DN or role DN to activate.

`-j` *file*
Password associated with the user DN. This option allows the password to be stored in clear text in the named file for scripting.

This is considered insecure. Use with extreme caution.

[`-p` *port*]
Directory Server port. The default value is the LDAP port of Directory Server specified at installation time.

`-w` *password*
Password associated with the user DN. Supplying the password on the command line is visible using the `/bin/ps` command. This is considered insecure. Use with extreme caution.

The value `-` can be used in place the password. The program prompts the user for a password to be entered from the terminal.

account-activate *args*
Activates an entry or group of entries.

The `account-activate` subcommand supports the following arguments

`-D` *rootdn*
Directory Server userDN with root permissions, such as Directory Manager.

`-h` *host*
Host name of Directory Server. The default value is the full hostname of the machine where Directory Server is installed.

`-I` DN
Entry DN or role DN to activate.

`-j` *file*
Password associated with the user DN. This option allows the password to be stored in clear text in the named file for scripting.

This is considered insecure. Use with extreme caution.

`-p` *port*
Directory Server port. The default value is the LDAP port of Directory Server specified at installation time.

`-w` *password*
Password associated with the user DN. Supplying the password on the command line is visible using the `/bin/ps` command. This is considered insecure. Use with extreme caution.

The value `-`can be used in place the password. The program prompts the user for a password to be entered from the terminal.

account-status *args*

Provides account status information to establish whether an entry or group of entries is inactivated or not.

The `account-status` subcommand supports the following arguments:

`-D` *rootdn*

`-h` *host*	Host name of Directory Server. The default value is the full hostname of the machine where Directory Server is installed.
`-I` *DN*	Entry DN or role DN whose status is required.
`-j` *file*	Password associated with the user DN. This option allows the password to be stored in clear text in the named file for scripting. This is considered insecure. Use with extreme caution.
`-p` *port*	Directory Server port. The default value is the LDAP port of Directory Server specified at installation time.
`-w` *password*	Password associated with the rootDN. Supplying the password on the command line is visible using the `/bin/ps`

command. This is considered insecure. Use with extreme caution.

The value `-` can be used in place of the password. The program prompts the user for a password to be entered from the terminal.

admin_ip *args*

Change the IP address of the the administrative server in the configuration.

The `admin_ip` subcommand supports the following arguments:

dir_mgr_DN
Directory Manager's DN.

dir_mgr_password
Directory Manager's password.

old_ip
Old IP.

new_ip
New IP.

port_#
Port number.

bak2db *backup_directory*

Restore the database from the most recent archived backup.

Specify *backup_directory* as the backup directory.

bak2db-task *args*

Restore the data to the database.

The bak2db-task subcommand supports the following arguments:

[`-a` *directory*]
Directory where the backup files are stored. By default it is under `/var/ds5/slapd-serverID/bak`

`-D` *rootDN*
User DN with root permissions, such as Directory Manager. The default is the DN of the directory manager which is read from the `nsslapd-root` attribute under `cn=config`.

`-j` *file*
Password associated with the user DN. This option allows the password to be stored in clear text in the named file for scripting.

This is considered insecure. Use with extreme caution.

[`-t` *database_type*]
Database type. The only possible database type is `ldbm`.

[`-v`]
Verbose mode.

`-w` *password*
Password associated with the user DN. Supplying the password on the command line is visible using the `/bin/ps` command. This is considered insecure. Use with extreme caution.

The value `-` can be used in place the password. The program prompts the user for a password to be entered from the terminal.

db2bak-task *args*

Back up the contents of the database. It creates an entry in the directory that launches this dynamic task. An entry is generated based upon the values provided for each option.

The `db2bak-task` subcommand supports the following arguments:

[`-a` *directory*]
Directory where the backup files are stored. By default it is under `/var/ds5/slapd-serverID/bak`. The backup file is named according to the year-month-day-hour format (*YYYY_MM_DD_hhmmss*).

`-D` *rootDN*
User DN with root permissions, such as Directory Manager. The default is the DN of the directory manager which is read from the `nsslapd-root` attribute under `cn=config`.

`-j` *file*
Password associated with the user DN. This option allows the password to be stored in clear text in the named file for scripting.

This is considered insecure. Use with extreme caution.

`-t` *database_type*
Database type. The only possible database type is `ldbm`.

[`-v`]
Verbose mode.

`-w` *password*
Password associated with the user DN. Supplying the password on the command line is visible using the `/bin/ps` command. This is considered insecure. Use with extreme caution.

The value `-` can be used in place the password. The program prompts the user for a password to be entered from the terminal.

db2bak [*backup_directory*]
Create a backup of the current database contents. The server must be stopped to run this subcommand.

The default is `/var/ds5/slapd-serverID/bak`. The backup file is named according to the year-month-day-hour format (*YYYY_MM_DD_hhmmss*).

db2bak *backup_directory*
Restore the database from the most recent archived backup.

Specify *backup_directory* as the backup directory.

db2index-text *args*
Create and generate the new set of indexes to be maintained following the modification of indexing entries in the `cn=config` configuration file.

The `db2index-text` subcommand supports the following arguments:

`-D` *rootdn*
User DN with root permissions, such as Directory Manager.

`-j` *file*
Password associated with the user DN. This option allows the password to be stored in

clear text in the named file for scripting. This is considered insecure. Use with extreme caution.

`-n` *backend_instance*
Instance to be indexed.

[`-t` *attributeName*]
Name of the attribute to be indexed. If omitted, all indexes defined for that instance are generated.

[`-v`]
Verbose mode.

`-w` *password*
Password associated with the user DN. Supplying the password on the command line is visible using the `/bin/ps` command. This is considered insecure. Use with extreme caution.

The value `-` can be used in place the password. The program prompts the user for a password to be entered from the terminal.

db2ldif-task *args*

Exports the contents of the database to LDIF. It creates an entry in the directory that launches this dynamic task. The entry is generated based upon the values you provide for each option. To run this subcommand the server must be running and either `-n` *backend_instance* or `-s` *include* suffix is required.

The `db2ldif-task` subcommand supports the following arguments:

[`-a` *outputfile*]
File name of the output LDIF file.

`-C`
Only the main db file is used.

`-D` *rootDN*
User DN with root permissions, such as Directory Manager.

`-j` *file*
Password associated with the user DN. This option allows the password to be stored in

clear text in the named file for scripting. This is considered insecure.Use with extreme caution.

[`-M`]
Output LDIF is stored in multiple files.

[`-m`]
Minimal base 64 encoding.

{`-n` *backend_instance*}*
Instance to be exported.

[`-N`]
Minimal base 64 encoding.

[`-o`]
Output LDIF to be stored in one file by default with each instance stored in *instance_file* name.

[`-r`]
Export replica.

[`-s`]*includesuffix*}*
Suffix(es) to be included or to specify the subtrees to be included if `-n` has been used.

[`-u`]
Request that the unique ID is not exported.

[`-U`]
Request that the output LDIF is not folded.

`-w` *password*
Password associated with the user DN. Supplying the password on the command line is visible using the `/bin/ps` command. This is considered insecure. Use with extreme caution.

The value `-` can be used in place the password. The program prompts the user for a password to be entered from the terminal.

{`-x` *excludesuffix*}*
Suffixes to be excluded.

[`-1`]
Delete, for reasons of backward compatibility the first line of the LDIF file that gives the version of the LDIF standard.

db2ldif *args*

Export the contents of the database to LDIF. You must specify either the `-n` or the `-s` option or both.

The `db2ldif` subcommand supports the following options:

[`-a` *outputfile*]
File name of the output LDIF file.

[`-C`]
Only use the main db file.

[`-m`]
Minimal base64 encoding.

[`-M`]
Use of several files for storing the output LDIF with each instance stored in *instance_file name* (where file name is the file name specified for `-a` option).

{`-n` *baclemd_instance*}*
Instance to be exported.

[`-N`]
Specify that the entry IDs are not to be included in the LDIF output. The entry IDs are necessary only if the `db2ldif` output is to be used as input to `db2index-text`.

[`-r`]
Export replica.

{`-s` *includesuffix*}*
Suffixes to be included or to specify the subtrees to be included if `-n` has been used.

[{`-x` *excludesuffix*}]*
Suffixes to be excluded.

[`-u`]
Request that the unique id is not exported.

[`-U`]
Request that the output LDIF is not folded.

[`-1`]
Delete, for reasons of backward compatibility, the first line of the LDIF file which gives the version of the LDIF standard.

help [*subcommand*]
: Display `directoryserver` usage message or subcommand specific usage message.

idsktune *args*
: Provide an easy and reliable way of checking the patch levels and kernel parameter settings for your system. You must install the Directory Server before you can run `idsktune`. It gathers information about the operating system, kernel, and TCP stack to make tuning recommendations.

  The `idsktune` subcommand supports the following arguments:

  [`-c`]
  : Client-specific tuning: the output only includes tuning recommendations for running a directory client application.

  [`-D`]
  : Debug mode: the output includes the commands it runs internally, preceded by `DEBUG` heading.

  [`-i` *installdir*]
  : The install directory.

  [`-q`]
  : Quiet mode. Output only includes tuning recommendations. OS version statements are omitted.

  [`-v`]
  : Version. Gives the build date identifying the version of the toll.

keyupg *args*
: Upgrade the key from Lite to normal (only one way).

  The `keyupg` subcommand supports the following arguments:

  `-k`*key*
  : The key to be upgraded.

  `-f` *key_file_path*
  : The key file path.

ldif2db-task args
: Import data to the directory. It create an entry in the directory that launches this dynamic task. The entry is generated based upon the values you provide for each option. The server must be running when you run this subcommand.

The `ldif2sb-task` subcommand supports the following arguments:

[`-c`]
Request that only the core db is created without attribute indexes.

`-D` rootDN
User DN with root permissions, such as Directory Manager.

[`-g` *string*]
Generation of a unique ID. Enter `none` for no unique ID to be generated and `deterministic` for the generated unique ID to be name-based. Generates a time based unique ID by default.

If you use the `deterministic` generation to have a name-based unique ID, you can also specify the namespace you want the server to use as follows:

`-g deterministic` *namespace_id*
where namespace_id is a string of characters in the following format

`00-xxxxxxxx-xxxxxxxx-xxxxxxxx-xxxxxxxx`
Use this option if you want to import the same LDIF file into two different directory servers, and you want the contents of both directories to have the same set of unique IDs. If unique IDs already exist in the LDIF file you are importing, then the existing IDs are imported to the server regardless of the options you have specified.

[`-G` *namespace_id*]
Generate a namespace ID as a name-based unique ID. This is the same as specifying `-g deterministic`.

{`-i` *filename*}*
File name of the input LDIF files. When you import multiple files, they are imported in the order in which you specify them on the command line.

`-j` *file*
Password associated with the user DN. This option allows the password to be stored in

clear text in the named file for scripting. This is considered insecure. Use with extreme caution.

`-n` backend_instance
Instance to be imported.

[`-O`]
Request that only the core db is created without attribute indexes.

{`-s` *includesuffix* }*
Suffixes to be included. This argument can also be used to specify the subtrees to be included with `-n`.

`-w` *password*
Password associated with the user DN. Supplying the password on the command line is visible using the `/bin/ps` command. This is considered insecure. Use with extreme caution.

The value `-` can be used in place the password. The program prompts the user for a password to be entered from the terminal.

[{`-x` *excludesuffix* }*]

[`-v`]
Verbose mode.

ldif *args*

Format LDIF files, and create base 64 encoded attribute values. With Base 64 Encoding you can represent binary data, such as a JPEG image, in LDIF by using base 64 encoding. You identify base 64 encoded data by using the `::` symbol. The `ldif`subcommand takes any input and formats it with the correct line continuation and appropriate attribute information. The subcommand also senses whether the input requires base 64 encoding.

The `ldif` subcommand supports the following arguments

[`-b`]
Interpret the entire input as a single binary value. If `-b` is not present, each line is considered to be a separate input value.

[*attrtype*]
If `-b` is specified, the output is `attrtype:: <base 64 encoded value`.

ldif2db *args*

Import the data to the directory. To run this subcommand the server must be stopped. Note that ldif2db supports LDIF version 1 specifications. You can load an attribute using the URL specifier notation, for example: `jpegphoto:file:///tmp/myphoto.jpg`

[`-c`]
Merge chunk size.

[`-g` *string*]
Generation of a unique ID. Type none for no unique ID to be generated and deterministic for the generated unique ID to be name-based. By default a time based unique ID is generated.

If you use the deterministic generation to have a name-based unique ID, you can also specify the namespace you want the server to use as follows:

`-g deterministic` *namespace_id*

where *namespace_id* is a string of characters in the following format:

`00-xxxxxxxx-xxxxxxxx-xxxxxxxx-xxxxxxxx`

Use this option if you want to import the same LDIF file into two different directory servers, and you want the contents of both directories to have the same set of unique IDs. If unique IDs already exist in the LDIF file you are importing, then the existing IDs are imported to the server regardless of the options you have specified.

[`-G` *naemspace_id*]
Generate a namespace ID as a name-based unique ID. This is the same as specifying the `-g deterministic` option.

{`-` *filename*}*
File name of the input LDIF file(s). When you import multiple files, they are imported in the order in which you specify them on the command line.

`-n` *backend_instance*
Instance to be imported.

[`-O`]
Request that only the core db is created without attribute indexes.

{`-s` *includesuffix*}*
Suffixes to be included or to specify the subtrees to be included if `-n` has been used.

[{`-x` *excludesuffix*}*]
Suffixes to be excluded

ldif2ldap *rootDN password filename*

Perform an import operation over LDAP to the Directory Server. To run this subcommand the server must be running.

The `ldif2ldap` subcommand supports the following arguments:

rootdn
User DN with root permissions, such as Directory Manager.

password
Password associated with the user DN.

filename
File name of the file to be imported. When you import multiple files, they are imported in the order in which you specify them on the command line.

magt CONFIG INIT

Start SNMP master agent. The Config and INIT files are in `/usr/iplanet/ds5/plugins/snmp/magt`. For more information, see the *iPlanet Directory Server 5.1 Administrator's Guide*.

The `magt` subcommand supports the following options:

CONFIG
The CONFIG file defines the community and the manager that master agent works with. Specify the manager value as a valid system name or an IP address.

INIT
The INIT file is a nonvolatile file that contains information from the MIB-II system group, including system location and contact

information. If INIT doesn't already exist, starting the master agent for the first time creates it. An invalid manager name in the CONFIG file causes the master agent start-up to fail.

monitor

Retrieves performance monitoring information using the `ldapsearch` command-line utility.

mmldif *args*

Combine multiple LDIF files into a single authoritative set of entries. Typically each LDIF file is from a master server cooperating in a multi master replication agreement.[e.g. masters that refuse to sync up for whatever reason]. Optionally, it can generate LDIF change files that could be applied to original to bring it up to date with authoritative. At least two input files must be specified.

The `mmldif` subcommand supports the following arguments:

[`-c` *inputfile ...*]
Write a change file (.delta) for each input file. Specify *inputfile* as the input LDIF files.

[`-D`]
Print debugging information.

[`-o` *out.ldif*]
Write authoritative data to this file.

nativetoascii *args*

Convert one language encoding to another. For example, convert a native language to UTF-8 format.

The `nativetoascii` subcommand supports the following options:

`-d` *Encodings Directory*
Path to the directory which contains the `conv` directory

[`-i` *input_filename* `-o` *output_filename*]
The input file name and output file name.

`-l`
List supported encodings

`-r`
Replace existing files.

`-s` *suffix*
Suffix to be mapped to the backend.

`-s` *SourceEncoding*
Source Encoding of input stream.

`-t` *TargetEncoding*
Target Encoding of output stream.

`-v`
Verbose output.

pwdhash *args*

Print the encrypted form of a password using one of the server's encryption algorithms. If a user cannot log in, you can use this script to compare the user's password to the password stored in the directory.

The `pwdhash` subcommand supports the following arguments:

`-c` *comparepwd* | `-s` *scheme*
The available schemes are `SSHA`, `SHA`, `CRYPT` and `CLEARE`. It generates the encrypted passwords according to scheme's algorithm. The `-c` specifies the encrypted password to be compared with. The result of comparison is either `OK` or `doesn't match`.

`-D` *instance-dir*
The instance directory.

[`-H`]
The passwords are hex-encoded.

password ...
The clear passwords to generate encrypted form from or to be compared with.

restart

Restarts the directory server.

When the `-s` option is not specified, restarts all instances of servers. When the `-s` option is specified, restarts the server specified by `-s`.

restart-admin

Restarts the administration server.

restoreconfig

Restores the most recently saved Administration Server configuration information to the NetscapeRoot partition under `/var/ds5/slapd-serverID/confbak`.

sagt `-c` CONFIG	Start proxy SNMP agent. For more information, see the *iPlanet Directory Server 5.1 Administrator's Guide*. The `sagt` subcommand supports the following options: `-c` *configfile* The CONFIG file includes the port that the SNMP daemon listens to. It also needs to include the MIB trees and traps that the proxy SNMP agent forwards. Edit the CONFIG file located in `/usr/iplanet/ds5/plugins/snmp/sagt`.
saveconfig	Saves the administration server configuration information to the `/var/ds5/slapd-serverID/confbak` directory.
setup [`-f` *configuration_file*]	Configures an instance of the directory server or administration server. Creates a basic configuration for the directory server and the administrative server that is used to manage the directory. The `setup` subcommand has two modes of operation. You can invoke it with a curses-based interaction to gather input. Alternatively, you can provide input in a configuration file using the `-f` option. The setup subcommand supports the following option: `-f` *configuration_file* Specifies the configuration file for silent installation.
start	Starts the directory server. When the `-s` option is not specified, starts servers of all instances. When the `-s` option is specified, starts the server instance specified by `-s`.
start-admin	Starts the directory server. When the `-s` option is not specified, restarts all instances of servers. When the `-s` option is specified, restarts the server specified by `-s`.
startconsole	Starts the directory console..

stop
: Stops the directory server.

  When the `-s` option is not specified, restarts all instances of servers. When the `-s` option is specified, restarts the server specified by `-s`.

stop-admin
: Stop the administration server.

suffix2instance {`-s` *suffix*}
: Map a suffix to a backend name.

  Specify `-s` *suffix* as the suffix to be mapped to the backend.

uninstall
: Uninstalls the directory server and the administration server.

  This subcommand stops servers of all instances and removes all the changes created by setup.

vlvindex *args*
: Create virtual list view (VLV) indexes, known in the Directory Server Console as Browsing Indexes. The server must be stopped beforehand.

  The `vlvindex` subcommand supports the following arguments:

  `-d` *debug_level*
  : Specify the debug level to use during index creation. Debug levels are defined in `nsslapd-errorlog-level (error Log Level)`. See the *iPlanet Directory Server 5.1 Configuration, Command, and File Reference*.

  `-n` *backend_instance*
  : Name of the database containing the entries to index.

  `-s` *suffix*
  : Name of the suffix containing the entries to index.

  `-T` *VLVTag*
  : Name of the database containing the entries to index.

OPTIONS

Options for the `directoryserver` command itself must appear before the subcommand argument.

The following options are supported:

`-s` *server-instance*
`-server`

server-instance The server instance name. Specify the directory server instance to process the command against. For some of the listed subcommands the server instance is optional and for other sub commands it is a required option.

EXAMPLES

EXAMPLE 1 Starting All Instances of the Directory Servers

The following command starts all the instances of the directory servers:

```
example% directoryserver start
```

EXAMPLE 2 Starting the Instances of `myhost` of the Directory Server

The following command starts the instances `myhost` of the directory server.

```
example% directoryserver -s myhost start
```

EXAMPLE 3 Running the Monitor Tool and Outputting the Current Status

The following command runs the monitor tool and output the current status of the `ephesus` directory instance.

```
example% directoryserver -s ephesus monitor
```

EXAMPLE 4 Running the idsktune Tool and Outputting Performance Tuning Information

The following command runs the idsktune tool and outputs performance tuning information:

```
example% directoryserver idsktune
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`non-zero` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	IPLTdsr, IPLTdsu

SEE ALSO

iPlanet Directory Server 5.1 Administrator's Guide

iPlanet Directory Server 5.1 Configuration, Command, and File Reference

NAME

disks – creates /dev entries for hard disks attached to the system

SYNOPSIS

/usr/sbin/disks [-C] [-r *rootdir*]

DESCRIPTION

devfsadm(1M) is now the preferred command for /dev and /devices and should be used instead of disks.

disks creates symbolic links in the /dev/dsk and /dev/rdsk directories pointing to the actual disk device special files under the /devices directory tree. It performs the following steps:

1. disks searches the kernel device tree to see what hard disks are attached to the system. It notes the /devices pathnames for the slices on the drive and determines the physical component of the corresponding /dev/dsk or /dev/rdsk name.
2. The /dev/dsk and /dev/rdsk directories are checked for disk entries – that is, symbolic links with names of the form c*N*[t*N*]d*N*s*N*, or c*N*[t*N*]dNp*N*, where *N* represents a decimal number. c*N* is the logical controller number, an arbitrary number assigned by this program to designate a particular disk controller. The first controller found on the first occasion this program is run on a system, is assigned number 0. t*N* is the bus-address number of a subsidiary controller attached to a peripheral bus such as SCSI or IPI (the target number for SCSI, and the facility number for IPI controllers). d*N* is the number of the disk attached to the controller. s*N* is the *slice* number on the disk. p*N* is the FDISK partition number used by fdisk(1M). (IA Only)
3. If only some of the disk entries are found in /dev/dsk for a disk that has been found under the /devices directory tree, disks creates the missing symbolic links. If none of the entries for a particular disk are found in /dev/dsk, disks checks to see if any entries exist for other disks attached to the same controller, and if so, creates new entries using the same controller number as used for other disks on the same controller. If no other /dev/dsk entries are found for slices of disks belonging to the same physical controller as the current disk, disks assigns the lowest-unused controller number and creates entries for the disk slices using this newly-assigned controller number.

disks is run automatically each time a reconfiguration-boot is performed or when add_drv(1M) is executed. When invoking disks(1M) manually, first run drvconfig(1M) to ensure /devices is consistent with the current device configuration.

Notice to Driver Writers

disks considers all devices with a node type of DDI_NT_BLOCK, DDI_NT_BLOCK_CHAN, DDI_NT_CD, DDI_NT_BLOCK_WWN or DDI_NT_CD_CHAN to be disk devices. disks(1M) requires the minor name of disk devices obey the following format conventions.

The minor name for block interfaces consists of a single lowercase ASCII character, a through u. The minor name for character (raw) interfaces consists of a single lowercase ASCII character, a through u, followed by ,raw.

`disks` translates `a` through `p` to `s0` through `s15`, while it translates `q` through `u` to `p0` through `p4`. SPARC drivers should only use the first 8 slices: `a` through `h`, while IA drivers can use `a` through `u`, with `q` through `u` corresponding to `fdisk`(1M) partitions. `q` represents the entire disk, while `r`, `s`, `t`, and `u` represent up to 4 additional partitions.

To prevent `disks` from attempting to automatically generate links for a device, drivers must specify a private node type and refrain from using a node type: `DDI_NT_BLOCK`, `DDI_NT_BLOCK_CHAN`, `DDI_NT_CD`, or `DDI_NT_CD_CHAN` when calling `ddi_create_minor_node`(9F).

OPTIONS

`-C` Causes disks to remove any invalid links after adding any new entries to `/dev/dsk` and `/dev/rdsk`. Invalid links are links which refer to non-existent disk nodes that have been removed, powered off, or are otherwise inaccessible.

`-r` *rootdir* Causes `disks` to presume that the `/dev/dsk`, `/dev/rdsk` and `/devices` directory trees are found under *rootdir*, not directly under `/`.

ERRORS

If disks finds entries of a particular logical controller linked to different physical controllers, it prints an error message and exits without making any changes to the `/dev` directory, since it cannot determine which of the two alternative logical-to-physical mappings is correct. The links should be manually corrected or removed before another reconfiguration-boot is performed.

EXAMPLES

EXAMPLE 1 Creating The Block And Character Minor Devices From Within The `xkdisk` Driver's `attach`(9E) Function.

The following example demonstrates creating the block and character minor devices from within the `xkdisk` driver's `attach`(9E) function.

```
#include    <sys/dkio.h>
/*
 * Create the minor number by combining the instance number
 * with the slice number.
 */
#define MINOR_NUM(i, s)        ((i) << 4 | (s))

int
xkdiskattach(dev_info_t *dip, ddi_attach_cmd_t cmd)
{
    int instance, slice;
    char name[8];

    /* other stuff in attach... */

    instance = ddi_get_instance(dip);
    for (slice = 0; slice < V_NUMPAR; slice++) {
        /*
         * create block device interface
         */
        sprintf(name, "%c", slice + 'a');
```

EXAMPLE 1 Creating The Block And Character Minor Devices From Within The `xkdisk` Driver's `attach`(9E) Function. *(Continued)*

```
        ddi_create_minor_node(dip, name, S_IFBLK,
            MINOR_NUM(instance, slice), DDI_NT_BLOCK_CHAN, 0);

        /*
         * create the raw (character) device interface
         */
        sprintf(name,"%c,raw", slice + 'a');
        ddi_create_minor_node(dip, name, S_IFCHR,
            MINOR_NUM(instance, slice), DDI_NT_BLOCK_CHAN, 0);
    }
}
```

Installing the `xkdisk` disk driver on a SPARCstation 20, with the driver controlling a SCSI disk (target 3 attached to an `esp`(7D) SCSI HBA) and performing a reconfiguration-boot (causing disks to be run) creates the following special files in `/devices`.

```
# ls -l /devices/iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/
brw-r-----   1 root sys   32, 16 Aug 29 00:02 xkdisk@3,0:a
crw-r-----   1 root sys   32, 16 Aug 29 00:02 xkdisk@3,0:a,raw
brw-r-----   1 root sys   32, 17 Aug 29 00:02 xkdisk@3,0:b
crw-r-----   1 root sys   32, 17 Aug 29 00:02 xkdisk@3,0:b,raw
brw-r-----   1 root sys   32, 18 Aug 29 00:02 xkdisk@3,0:c
crw-r-----   1 root sys   32, 18 Aug 29 00:02 xkdisk@3,0:c,raw
brw-r-----   1 root sys   32, 19 Aug 29 00:02 xkdisk@3,0:d
crw-r-----   1 root sys   32, 19 Aug 29 00:02 xkdisk@3,0:d,raw
brw-r-----   1 root sys   32, 20 Aug 29 00:02 xkdisk@3,0:e
crw-r-----   1 root sys   32, 20 Aug 29 00:02 xkdisk@3,0:e,raw
brw-r-----   1 root sys   32, 21 Aug 29 00:02 xkdisk@3,0:f
crw-r-----   1 root sys   32, 21 Aug 29 00:02 xkdisk@3,0:f,raw
brw-r-----   1 root sys   32, 22 Aug 29 00:02 xkdisk@3,0:g
crw-r-----   1 root sys   32, 22 Aug 29 00:02 xkdisk@3,0:g,raw
brw-r-----   1 root sys   32, 23 Aug 29 00:02 xkdisk@3,0:h
crw-r-----   1 root sys   32, 23 Aug 29 00:02 xkdisk@3,0:h,raw
```

/dev/dsk will contain the disk entries to the block device nodes in `/devices`

```
# ls -l /dev/dsk
/dev/dsk/c0t3d0s0 -> ../../devices/[...]/xkdisk@3,0:a
/dev/dsk/c0t3d0s1 -> ../../devices/[...]/xkdisk@3,0:b
/dev/dsk/c0t3d0s2 -> ../../devices/[...]/xkdisk@3,0:c
/dev/dsk/c0t3d0s3 -> ../../devices/[...]/xkdisk@3,0:d
/dev/dsk/c0t3d0s4 -> ../../devices/[...]/xkdisk@3,0:e
/dev/dsk/c0t3d0s5 -> ../../devices/[...]/xkdisk@3,0:f
/dev/dsk/c0t3d0s6 -> ../../devices/[...]/xkdisk@3,0:g
/dev/dsk/c0t3d0s7 -> ../../devices/[...]/xkdisk@3,0:h
```

and /dev/rdsk will contain the disk entries for the character device nodes in `/devices`

```
# ls -l /dev/rdsk
/dev/rdsk/c0t3d0s0 -> ../../devices/[...]/xkdisk@3,0:a,raw
/dev/rdsk/c0t3d0s1 -> ../../devices/[...]/xkdisk@3,0:b,raw
```

EXAMPLE 1 Creating The Block And Character Minor Devices From Within The `xkdisk` Driver's `attach`(9E) Function. *(Continued)*

```
/dev/rdsk/c0t3d0s2 -> ../../devices/[...]/xkdisk@3,0:c,raw
/dev/rdsk/c0t3d0s3 -> ../../devices/[...]/xkdisk@3,0:d,raw
/dev/rdsk/c0t3d0s4 -> ../../devices/[...]/xkdisk@3,0:e,raw
/dev/rdsk/c0t3d0s5 -> ../../devices/[...]/xkdisk@3,0:f,raw
/dev/rdsk/c0t3d0s6 -> ../../devices/[...]/xkdisk@3,0:g,raw
/dev/rdsk/c0t3d0s7 -> ../../devices/[...]/xkdisk@3,0:h,raw
```

FILES

`/dev/dsk/*` disk entries (block device interface)

`/dev/rdsk/*` disk entries (character device interface)

`/devices/*` device special files (minor device nodes)

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`add_drv`(1M), `devfsadm`(1M), `devlinks`(1M), `drvconfig`(1M), `fdisk`(1M), `ports`(1M), `tapes`(1M), `attributes`(5), `dkio`(7I), `esp`(7D), `attach`(9E), `ddi_create_minor_node`(9F)

Writing Device Drivers

BUGS

`disks` silently ignores malformed minor device names.

NAME

diskscan – perform surface analysis

SYNOPSIS

diskscan [-W] [-n] [-y] *raw_device*

DESCRIPTION

diskscan is used by the system administrator to perform surface analysis on a portion of a hard disk. The disk portion may be a raw partition or slice; it is identified using its raw device name. By default, the specified portion of the disk is read (non-destructive) and errors reported on standard error. In addition, a progress report is printed on standard out. The list of bad blocks should be saved in a file and later fed into addbadsec(1M), which will remap them.

OPTIONS

The following options are supported:

-n Causes diskscan to suppress linefeeds when printing progress information on standard out.

-W Causes diskscan to perform write and read surface analysis. This type of surface analysis is destructive and should be invoked with caution.

-y Causes diskscan to suppress the warning regarding destruction of existing data that is issued when -W is used.

OPERANDS

The following operands are supported:

raw_device The address of the disk drive (see FILES).

FILES

The raw device should be /dev/rdsk/c?[t?]d?[ps]?. See disks(1M) for an explanation of SCSI and IDE device naming conventions.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Architecture	IA
Availability	SUNWcsu

SEE ALSO

addbadsec(1M), disks(1M), fdisk(1M), fmthard(1M), format(1M), attributes(5)

NOTES

The format(1M) utility is available to format, label, analyze, and repair SCSI disks. This utility is included with the diskscan, addbadsec(1M), fdisk(1M), and fmthard(1M) commands available for IA. To format an IDE disk, use the DOS format utility; however, to label, analyze, or repair IDE disks on IA systems, use the Solaris format(1M) utility.

NAME

dispadmin – process scheduler administration

SYNOPSIS

dispadmin -l

dispadmin -c *class* -g [-r *res*]

dispadmin -d [*class*]

DESCRIPTION

The dispadmin command displays or changes process scheduler parameters while the system is running.

dispadmin does limited checking on the values supplied in *file* to verify that they are within their required bounds. The checking, however, does not attempt to analyze the effect that the new values have on the performance of the system. Inappropriate values can have a negative effect on system performance. (See *System Administration Guide: Basic Administration*

OPTIONS

The following options are supported:

-c *class* — Specifies the class whose parameters are to be displayed or changed. Valid *class* values are: RT for the real-time class, TS for the time-sharing class, IA for the inter-active class, FSS for the fair-share class, and FX for the fixed-priority class. The time-sharing and inter-active classes share the same scheduler, so changes to the scheduling parameters of one will change those of the other.

-d [*class*] — Sets or displays the name of the default scheduling class to be used on reboot by the startup script /etc/init.d/sysetup. If class name is not specified, the name and description of the current default scheduling class is displayed. If class name is specified and is a valid scheduling class name, then it is saved in dispadmin's private configuration file /etc/dispadmin.conf. Only super-users can set the default scheduling class.

-g — Gets the parameters for the specified class and writes them to the standard output. Parameters for the real-time class are described in rt_dptbl(4). Parameters for the time-sharing and inter-active classes are described in ts_dptbl(4).Parameters for the fair-share class are described in FSS(7). Parameters for the fixed-priority class are described in fx_dptbl(4).

The -g and -s options are mutually exclusive: you may not retrieve the table at the same time you are overwriting it.

-l — Lists the scheduler classes currently configured in the system.

-r *res* — When using the -g option you may also use the -r option to specify a resolution to be used for outputting the time quantum values. If no resolution is specified, time quantum values are in milliseconds. If *res* is specified it must be a positive integer between 1 and 1000000000 inclusive, and the resolution used is the

reciprocal of *res* in seconds. For example, a *res* value of 10 yields time quantum values expressed in tenths of a second; a *res* value of 1000000 yields time quantum values expressed in microseconds. If the time quantum cannot be expressed as an integer in the specified resolution, it is rounded up to the next integral multiple of the specified resolution.

`-s` *file* — Sets scheduler parameters for the specified class using the values in *file*. These values overwrite the current values in memory—they become the parameters that control scheduling of processes in the specified class. The values in *file* must be in the format output by the `-g` option. Moreover, the values must describe a table that is the same size (has same number of priority levels) as the table being overwritten. Super-user privileges are required in order to use the `-s` option.

The `-g` and `-s` options are mutually exclusive: you may not retrieve the table at the same time you are overwriting it.

EXAMPLES

EXAMPLE 1 Retrieving the Current Scheduler Parameters for the real-time class

The following command retrieves the current scheduler parameters for the real-time class from kernel memory and writes them to the standard output. Time quantum values are in microseconds.

```
dispadmin -c RT -g -r 1000000
```

EXAMPLE 2 Overwriting the Current Scheduler Parameters for the Real-time Class

The following command overwrites the current scheduler parameters for the real-time class with the values specified in `rt.config`.

```
dispadmin -c RT -s rt.config
```

EXAMPLE 3 Retrieving the Current Scheduler Parameters for the Time-sharing Class

The following command retrieves the current scheduler parameters for the time-sharing class from kernel memory and writes them to the standard output. Time quantum values are in nanoseconds.

```
dispadmin -c TS -g -r 1000000000
```

EXAMPLE 4 Overwriting the Current Scheduler Parameters for the Time-sharing Class

The following command overwrites the current scheduler parameters for the time-sharing class with the values specified in `ts.config`.

```
dispadmin -c TS -s ts.config
```

FILES

/etc/dispadmin.conf

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`priocntl`(1), `priocntl`(2), `fx_dptbl`(4), `rt_dptbl`(4), `ts_dptbl`(4), `attributes`(5), `FSS`(7)

System Administration Guide: Basic Administration Programming Interfaces Guide

DIAGNOSTICS

`dispadmin` prints an appropriate diagnostic message if it fails to overwrite the current scheduler parameters due to lack of required permissions or a problem with the specified input file.

NAME

dmesg – collect system diagnostic messages to form error log

SYNOPSIS

/usr/bin/dmesg

/usr/sbin/dmesg

DESCRIPTION

`dmesg` is made obsolete by `syslogd`(1M) for maintenance of the system error log.

`dmesg` looks in a system buffer for recently printed diagnostic messages and prints them on the standard output.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWesu (32-bit) SUNWesxu (64-bit)

SEE ALSO

`syslogd`(1M), `attributes`(5)

NAME

dmi_cmd – DMI command line interface utility

SYNOPSIS

dmi_cmd -AL -c *compId* -g *groupId* [-dp] [-a *attrId*] [-m *max-count*] [-r *req-mode*] [-s *hostname*]

dmi_cmd -CD -c *compId* [-s *hostname*]

dmi_cmd -CI *mif-file* [-s *hostname*]

dmi_cmd -CL [-dp] [-c *compId*] [-m *max-count*] [-r *req-mode*] [-s *hostname*]

dmi_cmd -GD -c *compId* -g *groupId* [-s *hostname*]

dmi_cmd -GI *schema-file* -c *compId* [-s *hostname*]

dmi_cmd -GL -c *compId* -g *groupId* [-dp] [-m *max-count*] [-r *req-mode*] [-s *hostname*]

dmi_cmd -GM -c *compId* [-m *max-count*] [-s *hostname*]

dmi_cmd -h

dmi_cmd -ND -c *compId* -l *language-string* [-s *hostname*]

dmi_cmd -NI *schema-file* -c *compId* [-s *hostname*]

dmi_cmd -NL -c *compId* [-s *hostname*]

dmi_cmd -V [-s *hostname*]

dmi_cmd -W *config-file* [-s *hostname*]

dmi_cmd -X [-s *hostname*]

DESCRIPTION

The dmi_cmd utility provides the ability to:

- Obtain version information about the DMI Service Provider
- Set the configuration to describe the language required by the management application
- Obtain configuration information describing the current language in use for the session
- Install components into the database
- List components in a system to determine what is installed
- Delete an existing component from the database
- Install group schemas to an existing component in the database
- List class names for all groups in a component
- List the groups within a component
- Delete a group from a component
- Install a language schema for an existing component in the database
- List the set of language mappings installed for a specified component
- Delete a specific language mapping for a component

- List the properties for one or more attributes in a group

OPTIONS

The following options are supported:

`-a` *attrId*	Specify an attribute by its ID (positive integer). The default value is `0`.
`-AL`	List the attributes for the specified component.
`-c` *compId*	Specify a component by its ID (positive integer). The default value is `0`.
`-CD`	Delete the specified component.
`-CI` *mif-file*	Install the component described in the *mif-file*.
`-CL`	List component information.
`-d`	Display descriptions.
`-g` *groupId*	Specify a group by its ID (positive integer). The default value is `0`.
`-GD`	Delete a group for the specified component.
`-GI` *schema-file*	Install the group schema specified in *schema-file*.
`-GL`	List the groups for the specified component.
`-GM`	List the class names for the specified component.
`-h`	Help. Print the command line usage.
`-l` *language-string*	Specify a language mapping.
`-m` *max-count*	Specify the maximum number of components to display.
`-ND`	Delete a language mapping for the specified component.
`-NI` *schema-file*	Install the language schema specified in *schema-file*.
`-NL`	List the language mappings for a specified component.
`-p`	Display the pragma string.
`-r` *req-mode*	Specify the request mode. The valid values are: `1` DMI_UNIQUE - access the specified item (or table row). `2` DMI_FIRST - access the first item. `3` DMI_NEXT - access the next item. The default request mode is `1` DMI_UNIQUE.
`-s` *hostname*	Specify the host machine on which `dmispd` is running. The default host is the local host.

`-V`	Version. Prints version information about the DMI Service Provider.
`-W` *config-file*	Set the configuration specified in *config-file* to `dmispd`.
`-X`	Retrieve configuration information describing the current language in use.

EXIT STATUS

The following error values are returned:

`0`	Successful completion.
`−1`	An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsadmi

SEE ALSO

`dmiget`(1M), `dmispd`(1M), `attributes`(5)

NAME

dmiget – DMI command line retrieval utility

SYNOPSIS

dmiget `-c` *compId* `[-a` *attrId*`]` `[-g` *groupId*`]` `[-s` *hostname*`]`

dmiget `-h`

DESCRIPTION

The `dmiget` utility retrieves the table information of a specific component in the DMI Service Provider.

OPTIONS

The following options are supported:

`-a` *attrId* — Display the attribute information for the component specified with the `-c` argument.

`-c` *compId* — Display all the table information for the specified component.

`-g` *groupId* — Display all the attribute information in the group specified with *groupId* for the component specified with the `-c` argument

`-h` — Help. Print the command line usage.

`-s` *hostname* — Specify the host machine on which `dmispd` is running. The default host is the local host.

EXIT STATUS

The following error values are returned:

`0` — Successful completion.

`–1` — An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsadmi

SEE ALSO

`dmi_cmd`(1M), `dmispd`(1M), `attributes`(5)

NAME

dminfo – report information about a device entry in a device maps file

SYNOPSIS

dminfo [-v] [-a] [-f *pathname*]

dminfo [-v] [-a] [-f *pathname*] -n *dev -name...*

dminfo [-v] [-a] [-f *pathname*] -d *dev -path...*

dminfo [-v] [-a] [-f *pathname*] -t *dev -type...*

dminfo [-v] [-f *pathname*] -u *dm -entry*

DESCRIPTION

dminfo reports and updates information about the device_maps(4) file.

OPTIONS

The following options are supported

-a	Succeed if any of the requested entries are found. If used with -v, all entries that match the requested case(s) are printed.
-d *dev–path*	Search by *dev–path*. Search device_maps(4) for a device special pathname in the *device_list* field matching the *dev–path* argument. This option cannot be used with -n, -t or -u.
-f *pathname*	Use a device_maps file with *pathname* instead of /etc/security/device_maps.
-n *dev–name*	Search by *dev–name*. Search device_maps(4) for a *device_name* field matching *dev–name*. This option cannot be used with -d, -t or -u.
-t *dev–type*	Search by *dev–type*. Search device_maps(4) for a *device_type* field matching the given *dev–type*. This option cannot be used with -d, -n or -u.
-u *dm–entry*	Update the device_maps(4) file. This option is provided to add entries to the device_maps(4) file. The *dm–entry* must be a complete device_maps(4) file entry. The *dm–entry* has fields, as in the device_maps file. It uses the colon (:) as a field separator, and white space as the *device_list* subfield separators. The *dm–entry* is not made if any fields are missing, or if the *dm–entry* would be a duplicate. The default device maps file can be updated only by the super user.
-v	Verbose. Print the requested entry or entries, one line per entry, on the standard output. If no entries are specified, all are printed.

EXIT STATUS

0	Successful completion.
1	Request failed.
2	Incorrect syntax.

FILES

`/etc/security/device_maps`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`bsmconv`(1M), `device_maps`(4), `attributes`(5)

NOTES

The functionality described in this man page is available only if the Basic Security Module (BSM) has been enabled. See `bsmconv`(1M) for more information.

NAME

dmispd – Sun Solstice Enterprise DMI Service Provider

SYNOPSIS

/usr/lib/dmi/dmispd [-h] [-c *config-dir*] [-d *debug-level*]

DESCRIPTION

The DMI Service Provider, `dmispd`, is the core of the DMI solution. Management applications and Component instrumentations communicate with each other through the Service Provider. The Service Provider coordinates and arbitrates requests from the management application to the specified component instrumentations. The Service Provider handles runtime management of the Component Interface (CI) and the Management Interface (MI), including component installation, registration at the MI and CI level, request serialization and synchronization, event handling for CI, and general flow control and housekeeping.

The Service Provider is invoked from a start-up script at boot time only if contents of the DMI Service Provider configuration file `/etc/dmi/conf/dmispd.conf` are non-trivial.

OPTIONS

The following options are supported:

`-c` *config-dir* — Specify the full path of the directory containing the `dmispd.conf` configuration file. The default directory is `/etc/dmi/conf`.

`-d` *debug-level* — Debug. Levels from 0 to 5 are supported, giving various levels of debug information. The default is 0, meaning no debug information is given.

If this option is omitted, then `dmispd` is run as a daemon process.

`-h` — Help. Print the command line usage.

EXIT STATUS

The following error values are returned:

`0` — Successful completion.

`1` — An error occurred.

FILES

`/etc/dmi/conf/dmispd.conf` — DMI Service Provider configuration file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsadmi

SEE ALSO

`snmpXdmid`(1M), `attributes`(5)

NAME

dnskeygen – generate public, private, and shared secret keys for DNS

SYNOPSIS

dnskeygen [[-DHR] *size*] [-F] [-zhu] [-a] [-c] [-p *num*] [-s *num*] -n *name*

DESCRIPTION

The dnskeygen utility is a tool to generate and maintain keys for DNS security with the Domain Name System ("DNS"). Use dnskeygen to generate public and private keys to authenticate zone data or shared secret keys for request and transaction signatures.

dnskeygen stores each key in two files:

```
K<name>+<algorithm>+<footprint>.private
```

and

```
K<name>+<algorithm>+<footprint>.key
```

The key is stored in a portable format within K<name>+<alg>+<footprint>.private. The public key is stored in K<name>+<alg>+<footprint>.private in the DNS zone file format:

```
<name> IN KEY <flags><algorithm><protocol><exponent|modulus>
```

The underlying cryptographic math is done by the DNSSAFE and Foundation Toolkit libraries.

OPTIONS

The dnskeygen utility supports the following options:

-D — Generate a DSA/DSS key. The value of *size* must be one of the following: 512, 576, 640, 704, 768, 832, 896, 960 or 1024.

-F — Use a large exponent for key generation. Use for RSA only.

-H — Generate a HMAC-MD5 key. The value of *size* must be between 128 and 504.

-R — Generate an RSA key. The value of *size* must be between 512 and 4096.

-a — Cannot use key for authentication.

-c — Cannot use key for encryption.

-h — Generate host or service key.

-n *name* — Set the key's name to *name*.

-p *num* — Set the key's protocol field to *num*. The values for *num* are as follows:

- 3 — If -z or -h is specified (DNSSEC), this is the default value.
- 2 — Unless specified, the default value for all other options.

	`1`	Use this value for TLS.
	`4`	Use this value for IPSEC.
	`255`	Use this value for ANY.
`-s` *num*	Set the key's strength field to *num*. The default value of *num* is `0`.	
`-u`	Generate User key, for example, for email.	
`-z`	Generate Zone key for DNS validation.	

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu
Interface Stability	Standard Bind 8.2.4

SEE ALSO

`attributes`(5)

Eastlake III, D. and Kaufman, C. *RFC 2065, Domain Name System Security Extension*. Network Working Group. January 1997.

Vixie, P., Gudmundsson, O., Eastlake III, D., and Wellington, B. *RFC 2845, Secret Key Transaction Authentication for DNS (TSIG)*. Network Working Group. May 2000.

NAME

domainname – Set or display name of the current domain

SYNOPSIS

domainname [*name-of-domain*]

DESCRIPTION

Without an argument, domainname displays the name of the current domain, which typically encompasses a group of hosts or passwd entries under the same administration.The domainname command is used by various components of Solaris to resolve names for types such as passwd, hosts and aliases. By default, various naming services such as NIS, NIS+, the Internet Domain Name Service (DNS) and sendmail(1M) use this domainname to resolve names. The domainname is normally a valid Internet domain name.

The domainname for various naming services can also be set by other means. For example, ypinit can be used to specify a different domain for all NIS calls. The file /etc/resolv.conf can be used to specify a different domain for DNS lookups. For sendmail, the domainname can be specified through the *sendmail_vars* entry in the /etc/nsswitch.conf file, or through the /etc/mail/sendmail.cf file. Only the superuser can set the name of the domain by specifying the new domainname as an argument. The domain name of the machine is usually set during boot-time through the domainname command in the /etc/init.d/inetinit file. If the new domain name is not saved in the /etc/defaultdomain file, the machine will revert back to the old domain after rebooting.

FILES

/etc/defaultdomain

/etc/init.d/inetinit

/etc/mail/sendmail.cf

/etc/nsswitch.conf

/etc/resolv.conf

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

nis+(1), nischown(1), nispasswd(1), hostconfig(1M), named(1M), nisaddcred(1M), sendmail(1M), ypinit(1M), sys-unconfig(1M), aliases(4), defaultdomain(4), hosts(4), nsswitch.conf(4), passwd(4), attributes(5)

NAME

drvconfig – configure the /devices directory

SYNOPSIS

```
drvconfig [-bn] [-a alias_name] [-c class_name] [-i drivername]
     [-m major_num] [-r rootdir]
```

DESCRIPTION

devfsadm(1M) is now the preferred command for /dev and /devices and should be used instead of drvconfig.

The default operation of drvconfig is to create the /devices directory tree that describes, in the filesystem namespace, the hardware layout of a particular machine. Hardware devices present on the machine and powered on as well as pseudo-drivers are represented under /devices. Normally this command is run automatically after a new driver has been installed (with add_drv(1M)) and the system has been rebooted.

/etc/minor_perm File

drvconfig reads the /etc/minor_perm file to obtain permission information and applies the permissions only to nodes that it has just created. It does not change permissions on already existing nodes. The format of the /etc/minor_perm file is as follows:

name : *minor_name permissions owner group*

minor_name may be the actual name of the minor node, or contain shell metacharacters to represent several minor nodes (see sh(1)).

For example:

```
sd:* 0640 root sys
zs:[a-z],cu 0600 uucp uucp
mm:kmem 0640 root bin
```

The first line sets all devices exported by the sd node to 0640 permissions, owned by root, with group sys. In the second line, devices such as a,cu and z,cu exported by the zs driver are set to 0600 permission, owned by uucp, with group uucp. In the third line the kmem device exported by the mm driver is set to 0640 permission, owned by root, with group bin.

OPTIONS

The following options are supported:

-a *alias_name*	Add the name *alias_name* to the list of aliases that this driver is known by. This option, if used, must be used with the -m *major_num*, the -b and the -i *drivername* options.
-b	Add a new major number to name binding into the kernel's internal name_to_major tables. This option is not normally used directly, but is used by other utilities such as add_drv(1M). Use of the -b option requires that -i and -m be used also. No /devices entries are created.
-c *class_name*	The driver being added to the system exports the class *class_name*. This option is not normally used directly, but is used by other utilities. It is only effective when used with the -b option.

`-i` *drivername* — Only configure the devices for the named driver. The following options are used by the implementation of `add_drv`(1M) and `rem_drv`(1M), and may not be supported in future versions of Solaris:

`-m` *major_num* — Specify the major number *major_num* for this driver to add to the kernel's `name_to_major` binding tables.

`-n` — Do not try to load and attach any drivers, or if the `-i` option is given, do not try to attach the driver named *drivername*.

`-r` *rootdir* — Build the device tree under the directory specified by *rootdir* instead of the default `/devices` directory.

EXIT STATUS

`0` — Successful completion.

non-zero — An error occurred.

FILES

`/devices`	Device nodes directory
`/etc/minor_perm`	Minor mode permissions
`/etc/name_to_major`	Major number binding
`/etc/driver_classes`	Driver class binding file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`sh`(1), `add_drv`(1M), `devlinks`(1M), `disks`(1M), `modinfo`(1M), `modload`(1M), `modunload`(1M), `ports`(1M), `rem_drv`(1M), `tapes`(1M), `path_to_inst`(4), `attributes`(5)

NOTES

This document does not constitute an API. `/etc/minor_perm`, `/etc/name_to_major`, `/etc/driver_classes`, and `/devices` may not exist or may have different contents or interpretations in a future release. The existence of this notice does not imply that any other documentation that lacks this notice constitutes an API.

NAME

dsvclockd – DHCP service lock daemon

SYNOPSIS

/usr/lib/inet/dsvclockd [-d 1 | 2] [-f] [-v]

DESCRIPTION

The dsvclockd daemon is a lock manager that works in conjunction with the Dynamic Host Configuration Protocol (DHCP) Data Service Library (libdhcpsvc). It provides shared or exclusive access to the dhcp_network(4) and dhcptab(4) tables. This service is used by the SUNWbinfiles and SUNWfiles DHCP data store modules. See dhcp_modules(5).

dsvclockd is started on demand by libdhcpsvc. The dsvclockd daemon should be started manually only if command line options need to be specified.

OPTIONS

The following options are supported:

-d 1 | 2 — Set debug level. Two levels of debugging are currently available, 1 and 2. Level 2 is more verbose.

-f — Run in the foreground instead of as a daemon process. When this option is used, messages are sent to standard error instead of to syslog(3C).

-v — Provide verbose output.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWdhcsu
Interface Stability	Unstable

SEE ALSO

syslog(3C), dhcp_network(4), dhcptab(4), dhcp_modules(5), attributes(5)

NAME

dumpadm – configure operating system crash dump

SYNOPSIS

/usr/sbin/dumpadm [-nuy] [-c *content-type*] [-d *dump-device*] [-m *min*k | *min*m | *min*%] [-s *savecore-dir*] [-r *root-dir*]

DESCRIPTION

The dumpadm program is an administrative command that manages the configuration of the operating system crash dump facility. A crash dump is a disk copy of the physical memory of the computer at the time of a fatal system error. When a fatal operating system error occurs, a message describing the error is printed to the console. The operating system then generates a crash dump by writing the contents of physical memory to a predetermined dump device, which is typically a local disk partition. The dump device can be configured by way of dumpadm. Once the crash dump has been written to the dump device, the system will reboot.

Fatal operating system errors can be caused by bugs in the operating system, its associated device drivers and loadable modules, or by faulty hardware. Whatever the cause, the crash dump itself provides invaluable information to your support engineer to aid in diagnosing the problem. As such, it is vital that the crash dump be retrieved and given to your support provider. Following an operating system crash, the savecore(1M) utility is executed automatically during boot to retrieve the crash dump from the dump device, and write it to a pair of files in your file system named *unix.X* and *vmcore.X*, where X is an integer identifying the dump. Together, these data files form the *saved crash dump*. The directory in which the crash dump is saved on reboot can also be configured using dumpadm.

By default, the dump device is configured to be an appropriate swap partition. Swap partitions are disk partitions reserved as virtual memory backing store for the operating system, and thus no permanent information resides there to be overwritten by the dump. See swap(1M). To view the current dump configuration, execute dumpadm with no arguments:

```
example# dumpadm

      Dump content: kernel pages
       Dump device: /dev/dsk/c0t0d0s1 (swap)
Savecore directory: /var/crash/saturn
  Savecore enabled: yes
```

When no options are specified, dumpadm prints the current crash dump configuration. The example shows the set of default values: the dump content is set to kernel memory pages only, the dump device is a swap disk partition, the directory for savecore files is set to /var/crash/hostname, and savecore is set to run automatically on reboot.

When one or more options are specified, dumpadm verifies that your changes are valid, and if so, reconfigures the crash dump parameters and displays the resulting configuration. You must be root to view or change dump parameters.

OPTIONS

The following options are supported:

`-c` *content-type*

Modify the dump configuration so that the crash dump consists of the specified dump content. The content should be one of the following:

`kernel`	Kernel memory pages only.
`all`	All memory pages.
`curproc`	Kernel memory pages, and the memory pages of the process whose thread was currently executing on the CPU on which the crash dump was initiated. If the thread executing on that CPU is a kernel thread not associated with any user process, only kernel pages will be dumped.

`-d` *dump-device*

Modify the dump configuration to use the specified dump device. The dump device may one of the following:

dump-device	A specific dump device specified as an absolute pathname, such as `/dev/dsk/` *cNtNdNsN*.
`swap`	If the special token `swap` is specified as the dump device, `dumpadm` examines the active swap entries and selects the most appropriate entry to configure as the dump device. See `swap`(1M). Refer to the `NOTES` below for details of the algorithm used to select an appropriate swap entry. When the system is first installed, dumpadm uses `swap` to determine the initial dump device setting.

`-m` *min*`k` | *min*`m` | *min*`%`

Create a `minfree` file in the current savecore directory indicating that `savecore` should maintain at least the specified amount of free space in the file system where the savecore directory is located. The `min` argument can be one of the following:

`k`	A positive integer suffixed with the unit `k` specifying kilobytes.
`m`	A positive integer suffixed with the unit `m` specifying megabytes.

`%` A % symbol, indicating that the `minfree` value should be computed as the specified percentage of the total current size of the file system containing the savecore directory. The `savecore` command will consult the `minfree` file, if present, prior to writing the dump files. If the size of these files would decrease the amount of free disk space below the `minfree` threshold, no dump files are written and an error message is logged. The administrator should immediately clean up the savecore directory to provide adequate free space, and re-execute the `savecore` command manually. The administrator can also specify an alternate directory on the `savecore` command-line.

`-n` Modify the dump configuration to not run `savecore` automatically on reboot. This is not the recommended system configuration; if the dump device is a swap partition, the dump data will be overwritten as the system begins to swap. If `savecore` is not executed shortly after boot, crash dump retrieval may not be possible.

`-r` *root-dir* Specify an alternate root directory relative to which `dumpadm` should create files. If no `-r` argument is specified, the default root directory "/" is used.

`-s` *savecore-dir* Modify the dump configuration to use the specified directory to save files written by `savecore`. The directory should be an absolute path and exist on the system. If upon reboot the directory does not exist, it will be created prior to the execution of `savecore`. See the `NOTES` section below for a discussion of security issues relating to access to the savecore directory. The default savecore directory is `/var/crash/hostname` where is the output of the `-n` option to the `uname`(1) command.

`-u` Forcibly update the kernel dump configuration based on the contents of `/etc/dumpadm.conf`. Normally this option is used only on reboot by the startup script `/etc/init.d/savecore`, when the `dumpadm` settings from the previous boot must be restored. Your dump configuration is saved in the configuration file for this purpose. If the configuration file is missing or contains invalid values for any dump properties, the

default values are substituted. Following the update, the configuration file is resynchronized with the kernel dump configuration.

`-y` Modify the dump configuration to automatically run `savecore` on reboot. This is the default for this dump setting.

EXAMPLES

EXAMPLE 1 Reconfiguring The Dump Device To A Dedicated Dump Device:

The following command reconfigures the dump device to a dedicated dump device:

```
example# dumpadm -d /dev/dsk/c0t2d0s2

          Dump content: kernel pages
           Dump device: /dev/dsk/c0t2d0s2 (dedicated)
    Savecore directory: /var/crash/saturn
      Savecore enabled: yes
```

EXIT STATUS

The following exit values are returned:

`0` Dump configuration is valid and the specified modifications, if any, were made successfully.

`1` A fatal error occurred in either obtaining or modifying the dump configuration.

`2` Invalid command line options were specified.

FILES

`/dev/dump`
`/etc/init.d/savecore`
`/etc/dumpadm.conf`
savecore-directory`/minfree`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsr

SEE ALSO

`uname`(1), `savecore`(1M), `swap`(1M), `attributes`(5)

Dump Device Selection

When the special `swap` token is specified as the argument to `dumpadm -d` the utility will attempt to configure the most appropriate swap device as the dump device. `dumpadm` configures the largest swap block device as the dump device; if no block devices are available for swap, the largest swap entry is configured as the dump device. If no swap entries are present, or none can be configured as the dump device, a warning message will be displayed. While local and remote swap files can be configured as the dump device, this is not recommended.

Dump Device/Swap Device Interaction

In the event that the dump device is also a swap device, and the swap device is deleted by the administrator using the `swap -d` command, the `swap` command will automatically invoke `dumpadm -d swap` in order to attempt to configure another appropriate swap device as the dump device. If no swap devices remain or none can be configured as the dump device, the crash dump will be disabled and a warning message will be displayed. Similarly, if the crash dump is disabled and the administrator adds a new swap device using the `swap -a` command, `dumpadm -d swap` will be invoked to re-enable the crash dump using the new swap device.

Once `dumpadm -d swap` has been issued, the new dump device is stored in the configuration file for subsequent reboots. If a larger or more appropriate swap device is added by the administrator, the dump device is not changed; the administrator must re-execute `dumpadm -d swap` to reselect the most appropriate device fom the new list of swap devices.

Minimum Free Space

If the `dumpadm -m` option is used to create a `minfree` file based on a percentage of the total size of the file system containing the savecore directory, this value is not automatically recomputed if the file system subsequently changes size. In this case, the administrator must re-execute `dumpadm -m` to recompute the `minfree` value. If no such file exists in the savecore directory, `savecore` will default to a free space threshold of one megabyte. If no free space threshold is desired, a minfree file containing size 0 can be created.

Security Issues

If, upon reboot, the specified savecore directory is not present, it will be created prior to the execution of `savecore` with permissions 0700 (read, write, execute by owner only) and owner `root`. It is recommended that alternate savecore directories also be created with similar permissions, as the operating system crash dump files themselves may contain secure information.

NAME

editmap – query and edit single records in database maps for sendmail

SYNOPSIS

editmap -C *file* [-N] [-f] [-q | -u | -x] *maptype mapname key* ["*value*"...]

DESCRIPTION

The editmap command queries or edits one record in a database maps used by the keyed map lookups in sendmail(1M). Arguments are passed on the command line and output (for queries) is directed to standard output.

Depending on how it is compiled, editmap handles up to three different database formats, selected using the *maptype* parameter. See OPERANDS.

If the TrustedUser option is set in the sendmail configuration file and editmap is invoked as root, the generated files are owned by the specified TrustedUser.

OPTIONS

The following options are supported:

-C *file* — Use the specified sendmail configuration file (*file*) to look up the TrustedUser option.

-f — Disable the folding of all upper case letters in the key to lower case. Normally, all upper case letters in the key are folded to upper case. This is intended to mesh with the -f flag in the K line in sendmail.cf. The value is never case folded.

-N — Include the null byte that terminates strings in the map (for alias maps).

-q — Query the map for the specified key. If found, print value to standard output and exit with 0. If not found then print an error message to stdout and exit with EX_UNAVAILABLE.

-u — Update the record for key with value or inserts a new record if one doesn't exist. Exits with 0 on success or EX_IOERR on failure.

-x — Delete the specific key from the map. Exit with 0 on success or EX_IOERR on failure.

OPERANDS

The following operands are supported:

key — The left hand side of a record.

Each record is of the form:

key value

key and *value* are separated by white space.

mapname — File name of the database map being created.

maptype — Specifies the database format. The following *maptype* parameters are available:

dbm — Specifies DBM format maps.

btree — Specifies B-Tree format maps.

`hash` Specifies hash format maps.

value The right hand side of a record.

Each record is of the form:

key value

key and *value* are separated by white space.

ATTRIBUTES See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsndmu

SEE ALSO `makemap`(1M), `sendmail`(1M), `attributes`(5)

NAME

edquota – edit user quotas for ufs file system

SYNOPSIS

edquota [-p *proto_user*] *username*...

edquota -t

DESCRIPTION

`edquota` is a quota editor. One or more users may be specified on the command line. For each user a temporary file is created with an ASCII representation of the current disk quotas for that user for each mounted ufs file system that has a `quotas` file, and an editor is then invoked on the file. The quotas may then be modified, new quotas added, etc. Upon leaving the editor, `edquota` reads the temporary file and modifies the binary quota files to reflect the changes made.

The editor invoked is `vi`(1) unless the `EDITOR` environment variable specifies otherwise.

Only the super-user may edit quotas. In order for quotas to be established on a file system, the root directory of the file system must contain a file, owned by root, called `quotas`. (See `quotaon`(1M).)

proto_user and *username* can be numeric, corresponding to the UID of a user. Unassigned UIDs may be specified; unassigned names may not. In this way, default quotas can be established for users who are later assigned a UID.

If no options are specified, the temporary file created will have one or more lines of the form

fs *mount_point* blocks (soft =*number*, hard =*number*) inodes (soft =*number*, hard =*number*)

Where a block is considered to be a 1024 byte (1K) block.

The *number* fields may be modified to reflect desired values.

OPTIONS

`-p` Duplicate the quotas of the *proto_user* specified for each *username* specified. This is the normal mechanism used to initialize quotas for groups of users.

`-t` Edit the soft time limits for each file system. If the time limits are zero, the default time limits in `/usr/include/sys/fs/ufs_quota.h` are used. The temporary file created will have one or more lines of the form

fs *mount_point* blocks time limit = *number tmunit*, files time limit = *number tmunit*

tmunit may be one of ''month'', ''week'', ''day'', ''hour'', ''min'' or ''sec''; characters appended to these keywords are ignored, so you may write ''months'' or ''minutes'' if you prefer. The *number* and *tmunit* fields may be modified to set desired values. Time limits are printed in the greatest possible time unit such that the value is greater than or equal to one. If ''default'' is printed after the *tmunit*, this indicates that the value shown is zero (the default).

USAGE

See `largefile`(5) for the description of the behavior of `edquota` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

FILES

`quotas`	quota file at the file system root
`/etc/mnttab`	table of mounted file systems

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`vi`(1), `quota`(1M), `quotacheck`(1M), `quotaon`(1M), `repquota`(1M), `attributes`(5), `largefile`(5), `quotactl`(7I)

NOTES

`quotacheck`(1M) *must* be invoked when setting initial quota limits for users; if not, the quota limit remains 0 and no changes made with `edquota` will take effect.

Users with a UID greater than 67108864 cannot be given quotas.

NAME

eeprom – EEPROM display and load utility

SPARC

/usr/platform/ *platform-name* /sbin/eeprom [-] [-f *device*] [*parameter* [=*value*]]

IA

/usr/platform/ *platform-name* /sbin/eeprom [-] [-f *device*] [-I] [*mmu-modlist*] [*parameter* [=*value*]]

DESCRIPTION

`eeprom` displays or changes the values of parameters in the EEPROM. It processes parameters in the order given. When processing a *parameter* accompanied by a *value*, `eeprom` makes the indicated alteration to the EEPROM; otherwise, it displays the *parameter*'s value. When given no parameter specifiers, `eeprom` displays the values of all EEPROM parameters. A '`-`' (hyphen) flag specifies that parameters and values are to be read from the standard input (one *parameter* or *parameter*=*value* per line).

Only the super-user may alter the EEPROM contents.

`eeprom` verifies the EEPROM checksums and complains if they are incorrect.

platform-name is the name of the platform implementation and can be found using the `-i` option of `uname`(1).

SPARC

SPARC based systems implement firmware password protection with `eeprom`, using the `security-mode`, `security-password` and `security-#badlogins` properties.

IA

EEPROM storage is simulated using a file residing in the platform-specific boot area. The `/platform/`*platform-name*`/boot/solaris/bootenv.rc` file simulates EEPROM storage.

Because IA based systems typically implement password protection in the system BIOS, there is no support for password protection in the `eeprom` program. While it is possible to set the `security-mode`, `security-password` and `security-#badlogins` properties on IA based systems, these properties have no special meaning or behavior on IA based systems.

OPTIONS

`-f` *device* — Use *device* as the EEPROM device.

IA Only

`-I` — Initialize boot properties on an IA based system. Only `init`(1M) run-level initialization scripts should use this option.

IA Only

acpi-user-options — A configuration variable that controls the use of ACPI. A value of `0x0` attempts to use ACPI if it is available on the system. A value of `0x2` disables the use of ACPI. Defaults to `0x0`.

mmu-modlist — A colon-separated list of candidate modules that implement memory management. If *mmu-modlist* is defined, it overrides the default list derived from the memory configuration on IA based systems. Instead, the first module in the list that is found in `/platform/`*platform-name*`/kernel/mmu` is used.

NVRAM CONFIGURATION PARAMETERS

Not all OpenBoot systems support all parameters. Defaults vary depending on the system and the PROM revision. See the output in the "Default Value" column of the `printenv` command, as entered at the `ok` (OpenBoot) prompt, to determine the default for your system.

auto-boot?	If `true`, boots automatically after power-on or reset. Defaults to `true`.
ansi-terminal?	Configuration variable used to control the behavior of the terminal emulator. The value `false` makes the terminal emulator stop interpreting ANSI escape sequences; instead, echoes them to the output device. Defaults to `true`.
boot-command	Command executed if `auto-boot?` is `true`. Defaults to `boot`.
boot-device	Device from which to boot. *boot-device* may contain 0 or more device specifiers separated by spaces. Each device specifier may be either a prom device alias or a prom device path. The boot prom will attempt to open each successive device specifier in the list beginning with the first device specifier. The first device specifier that opens successfully will be used as the device to boot from. Defaults to `disk net`.
boot-file	File to boot (an empty string lets the secondary booter choose default). Defaults to empty string.
boot-from	Boot device and file (OpenBoot PROM version 1.*x* only). Defaults to `vmunix`.
boot-from-diag	Diagnostic boot device and file (OpenBoot PROM version 1.*x* only). Defaults to `le( )unix`.
com*X*-noprobe	Where *X* is the number of the serial port, prevents device probe on serial port *X*.
diag-device	Diagnostic boot source device. Defaults to `net`.
diag-file	File from which to boot in diagnostic mode. Defaults to empty string.
diag-level	Diagnostics level. Values include `off`, `min`, `max` and `menus`. There may be additional platform-specific values. When set to `off`, POST is not called. If POST is called, the value is made available as an argument to, and is interpreted by POST. Defaults to `platform-dependent`.
diag-switch?	If `true`, run in diagnostic mode. Defaults to `false` on most desktop systems, `true` on most servers.

error-reset-recovery	Recover after an error reset trap. Defaults to platform-specific setting. On platforms supporting this variable, it replaces the `watchdog-reboot?`, `watchdog-sync?`, `redmode-reboot?`, `redmode-sync?`, `sir-sync?`, and `xir-sync?` parameters. The options are: none — Print a message describing the reset trap and go to OpenBoot PROM's user interface, *aka* `OK` prompt. sync — Invoke OpenBoot PROM's `sync` word after the reset trap. Some platforms may treat this as `none` after an externally initiated reset (XIR) trap. boot — Reboot after the reset trap. Some platforms may treat this as `none` after an XIR trap.
fcode-debug?	If `true`, include name parameter for plug-in device FCodes. Defaults to `false`.
hardware-revision	System version information.
input-device	Input device used at power-on (usually `keyboard`, `ttya`, or `ttyb`). Defaults to `keyboard`.
keyboard-click?	If `true`, enable keyboard click. Defaults to `false`.
keymap	Keymap for custom keyboard.
last-hardware-update	System update information.
load-base	Default load address for client programs. Default value is `16384`.
local-mac-address?	If true, network drivers use their own MAC address, not the system's. Defaults to `false`.
mfg-mode	Manufacturing mode argument for POST. Possible values include `off` or `chamber`. The value is passed as an argument to POST. Defaults to `off`.
mfg-switch?	If true, repeat system self-tests until interrupted with STOP-A. Defaults to `false`.
nvramrc	Contents of NVRAMRC. Defaults to empty.
oem-banner	Custom OEM banner (enabled by setting `oem-banner?` to `true`). Defaults to empty string.
oem-banner?	If `true`, use custom OEM banner. Defaults to `false`.

oem-logo	Byte array custom OEM logo (enabled by setting `oem-logo?` to `true`). Displayed in hexadecimal.
oem-logo?	If `true`, use custom OEM logo (else, use Sun logo). Defaults to `false`.
output-device	Output device used at power-on (usually `screen`, `ttya`, or `ttyb`). Defaults to `screen`.
redmode-reboot?	Specify `true` to reboot after a redmode reset trap. Defaults to `true`. (Sun Enterprise 10000 only.)
redmode-sync?	Specify `true` to invoke OpenBoot PROM's `sync` word after a redmode reset trap. Defaults to `false`. (Sun Enterprise 10000 only.)
sbus-probe-list	Designate which SBus slots are probed and in what order. Defaults to `0123`.
screen-#columns	Number of on-screen columns (characters/line). Defaults to `80`.
screen-#rows	Number of on-screen rows (lines). Defaults to `34`.
scsi-initiator-id	SCSI bus address of host adapter, range 0-7. Defaults to `7`.
sd-targets	Map SCSI disk units (OpenBoot PROM version 1.*x* only). Defaults to `31204567`, which means that unit 0 maps to target `3`, unit 1 maps to target `1`, and so on.
security-#badlogins	Number of incorrect security password attempts.This property has no special meaning or behavior on IA based systems.
security-mode	Firmware security level (options: `none`, `command`, or `full`). If set to `command` or `full`, system will prompt for PROM security password. Defaults to `none`.This property has no special meaning or behavior on IA based systems.

security-password	Firmware security password (never displayed). Can be set only when `security-mode` is set to `command` or `full`.This property has no special meaning or behavior on IA based systems.
	`example# eeprom security-password=`
	`Changing PROM password:`
	`New password:`
	`Retype new password:`

selftest-#megs	Megabytes of RAM to test. Ignored if `diag-switch?` is `true`. Defaults to `1`.

sir-sync?	Specify `true` to invoke OpenBoot PROM's `sync` word after a software-initiated reset (SIR) trap. Defaults to `false`. (Sun Enterprise 10000 only.)
skip-vme-loopback?	If `true`, POST does not do VMEbus loopback tests. Defaults to `false`.
st-targets	Map SCSI tape units (OpenBoot PROM version 1.*x* only). Defaults to `45670123`, which means that unit 0 maps to target `4`, unit 1 maps to target `5`, and so on.
sunmon-compat?	If `true`, display Restricted Monitor prompt (>). Defaults to `false`.
testarea	One-byte scratch field, available for read/write test. Defaults to `0`.
tpe-link-test?	Enable 10baseT link test for built-in twisted pair Ethernet. Defaults to `true`.
ttya-mode	TTYA (baud rate, #bits, parity, #stop, handshake). Defaults to `9600,8,n,1,-`. Fields, in left-to-right order, are: Baud rate: 110, 300, 1200, 4800, 9600 . . . Data bits: 5, 6, 7, 8 Parity: n(none), e(even), o(odd), m(mark), s(space) Stop bits: 1, 1.5, 2 Handshake: –(none), h(hardware:rts/cts), s(software:xon/xoff)
ttyb-mode	TTYB (baud rate, #bits, parity, #stop, handshake). Defaults to `9600,8,n,1,-`. Fields, in left-to-right order, are: Baud rate: 110, 300, 1200, 4800, 9600 . . . Data bits: 5, 6, 7, 8 Stop bits: 1, 1.5, 2 Parity: n(none), e(even), o(odd), m(mark), s(space) Handshake: –(none), h(hardware:rts/cts), s(software:xon/xoff)
ttya-ignore-cd	If `true`, operating system ignores carrier-detect on TTYA. Defaults to `true`.

ttyb-ignore-cd	If `true`, operating system ignores carrier-detect on TTYB. Defaults to `true`.
ttya-rts-dtr-off	If `true`, operating system does not assert DTR and RTS on TTYA. Defaults to `false`.
ttyb-rts-dtr-off	If `true`, operating system does not assert DTR and RTS on TTYB. Defaults to `false`.
use-nvramrc?	If `true`, execute commands in NVRAMRC during system start-up. Defaults to `false`.
version2?	If `true`, hybrid (1.*x*/2.*x*) PROM comes up in version 2.*x*. Defaults to `true`.
watchdog-reboot?	If `true`, reboot after watchdog reset. Defaults to `false`.
watchdog-sync?	Specify `true` to invoke OpenBoot PROM's `sync` word after a watchdog reset trap. Defaults to `false`. (Sun Enterprise 10000 only.)
xir-sync?	Specify `true` to invoke OpenBoot PROM's `sync` word after an XIR trap. Defaults to `false`. (Sun Enterprise 10000 only.)

EXAMPLES

EXAMPLE 1 Changing the Number of Megabytes of RAM.

The following example demonstrates the method for changing from one to two the number of megabytes of RAM that the system will test.

```
example# eeprom selftest-#megs
selftest-#megs=1

example# eeprom selftest-#megs=2

example# eeprom selftest-#megs
selftest-#megs=2
```

EXAMPLE 2 Setting the `auto-boot?` Parameter to `true`.

The following example demonstrates the method for setting the `auto-boot?` parameter to `true`.

```
example# eeprom auto-boot?=true
```

When the `eeprom` command is executed in user mode, the parameters with a trailing question mark (?) need to be enclosed in double quotation marks (" ") to prevent the shell from interpreting the question mark. Preceding the question mark with an escape character (\) will also prevent the shell from interpreting the question mark.

```
example% eeprom "auto-boot?"=true
```

EXAMPLE 2 Setting the `auto-boot?` Parameter to `true`. *(Continued)*

EXAMPLE 3 Enabling and Disabling PAE Mode

Certain IA machines support Physical Address Extension (PAE) mode. To enable and disable PAE mode on these machines, use commands such as those below.

To enable PAE mode:

```
example# eeprom mmu-modlist=mmu36
```

To disable PAE mode:

```
example# eeprom mmu-modlist=mmu32
```

These commands take effect following your next reboot.

FILES

`/dev/openprom`
Device file

`/usr/platform/`*platform-name*`/sbin/eeprom`
Platform-specific version of `eeprom`. Use `uname -i` to obtain *platform-name*.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`passwd`(1), `init`(1M), `sh`(1), `uname`(1), `attributes`(5)

OpenBoot 3.x Command Reference Manual

ONC+ Developer's Guide

NAME

efdaemon – embedded FCode interpreter daemon

SYNOPSIS

/usr/lib/efcode/sparcv9/efdaemon [-d]

DESCRIPTION

`efdaemon`, the embedded `FCode` interpreter daemon, invokes the embedded `FCode` interpreter when the daemon receives an interpretation request. A new session of the interpreter is started for each unique request by invoking the script `/usr/lib/efcode/efcode`.

`efdaemon` is used on selected platforms as part of the processing of some dynamic reconfiguration events.

OPTIONS

The following option is supported:

`-d` Set debug output. Log debug messages as `LOG_DEBUG` level messages by using `syslog()`. See `syslog`(3C).

FILES

`/dev/fcode`
`FCode` interpreter pseudo device, which is a portal for receipt of `FCode` interpretation requests

`/usr/lib/efcode/efcode`
Shell script that invokes the embedded `FCode` interpreter

`/usr/lib/efcode/interpreter`
Embedded `FCode` interpreter

`/usr/lib/efcode/sparcv9/interpreter`
Embedded `FCode` interpreter

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWefcx, SUNWefcux, SUNWefcr, SUNWefclx

SEE ALSO

`prtconf`(1M), `syslog`(3C), `attributes`(5)

NAME

etrn – start mail queue run

SYNOPSIS

etrn [-v] *server-host* [*client-hosts*]

DESCRIPTION

SMTP's ETRN command allows an SMTP client and server to interact, giving the server an opportunity to start the processing of its queues for messages to go to a given host. This is meant to be used in start-up conditions, as well as for mail nodes that have transient connections to their service providers.

The etrn utility initiates an SMTP session with the host *server-host* and sends one or more ETRN commands as follows: If no *client-hosts* are specified, etrn looks up every host name for which sendmail(1M) accepts email and, for each name, sends an ETRN command with that name as the argument. If any *client-hosts* are specified, etrn uses each of these as arguments for successive ETRN commands.

OPTIONS

The following option is supported:

-v The normal mode of operation for etrn is to do all of its work silently. The -v option makes it verbose, which causes etrn to display its conversations with the remote SMTP server.

ENVIRONMENT VARIABLES

No environment variables are used. However, at system start-up, /etc/init.d/sendmail reads /etc/default/sendmail. In this file, if the variable ETRN_HOSTS is set, /etc/init.d/sendmail parses this variable and invokes etrn appropriately. ETRN_HOSTS should be of the form:

```
"s1:c1.1,c1.2        s2:c2.1 s3:c3.1,c3.2,c3.3"
```

That is, white-space separated groups of *server:client* where *client* can be one or more comma-separated names. Note that the *:client* part is optional. *server* is the name of the server to prod; a mail queue run is requested for each *client* name. This is comparable to running:

```
/usr/lib/sendmail -qR client
```

on the host *server*.

EXAMPLES

EXAMPLE 1 Using etrn

Inserting the line:

```
ETRN_HOSTS="s1.domain.com:clnt.domain.com s2.domain.com:clnt.domain.com"
```

in /etc/default/sendmail results in /etc/init.d/sendmail invoking etrn such that ETRN commands are sent to both s1.domain.com and s2.domain.com, with both having clnt.domain.com as the ETRN argument.

The line:

```
ETRN_HOSTS="server.domain.com:client1.domain.com,client2.domain.com"
```

EXAMPLE 1 Using etrn *(Continued)*

results in two `ETRN` commands being sent to `server.domain.com`, one with the argument `client1.domain.com`, the other with the argument `client2.domain.com`.

The line:

```
ETRN_HOSTS="server1.domain.com server2.domain.com"
```

results in set of a `ETRN` commands being sent to both `server1.domain.com` and `server2.domain.com`; each set contains one `ETRN` command for each host name for which `sendmail`(1M) accepts email, with that host name as the argument.

FILES

`/etc/mail/sendmail.cf`
: `sendmail` configuration file

`/etc/default/sendmail`
: Variables used by `/etc/init.d/sendmail`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsndmu

SEE ALSO

`sendmail`(1M), RFC 1985.

CAVEATS

Not all SMTP servers support `ETRN`.

NAME

fbconfig – Frame Buffer configuration utility

SYNOPSIS

fbconfig [-list | -help]

fbconfig [-dev *device_filename*] [-prconf] [-propt] [-res]

fbconfig [-dev *device_filename*] [-res *resolution-specification*] *device_specific_options*

DESCRIPTION

`fbconfig` is the generic command line interface to query and configure frame buffer attributes.

The following form of `fbconfig` is the interface for the device independent operations performed by `fbconfig`:

fbconfig [-list | -help]

The following form of `fbconfig` is the interface for configuring a frame buffer:

fbconfig [-dev *device_filename*] [-prconf] [-propt] [-res]

If the `-dev` option is omitted, the default frame buffer (`/dev/fb` or /dev/fb0) is assumed. In the absence of specific options, the response will depend upon the device specific configuration program and how it responds to no options

OPTIONS

The following options are supported:

`-dev` *device_filename*
Specify the FFB special file. The default is `/dev/fbs/ffb0`.

`-help`
Print the `fbconfig` command usage summary. This is the default option.

`-list`
Print the list of installed frame buffers and associated device specific configuration routines.

Device Filename	Specific Config Program
/dev/fbs/ffb0	SUNWffb_config
/dev/fbs/ffb1	SUNWffb_config
/dev/fbs/m640	SUNWm64_config
/dev/fbs/cgsix0	not configurable

`-prconf`
Print the current hardware configuration.

`-propt`
Print the current software configuration.

OPERANDS

The following operands are supported:

device_specific_options — *device_specific_options* are specified in the format shown by the `-help` output, or the corresponding

device-specific man page.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWfbc

SEE ALSO

`afbconfig`(1M), `ffbconfig`(1M), `attributes`(5)

LIMITATIONS

Because of limitations in the `m64` kernel driver and related software, `fbconfig` (with the `-prconf` option) is unable to distinguish between a current depth of 24 or 8+24. The `-propt` option returns the depth specified in the `OWconfig` file, which will be in effect following the next restart of the window system. The `xwininfo` utility, usually shipped in the package containing frame buffer software (such as `SUNWxwplt`), reports current depth of a specified window.

NAME

fdetach – detach a name from a STREAMS-based file descriptor

SYNOPSIS

`fdetach` *path*

DESCRIPTION

The `fdetach` command detaches a STREAMS-based file descriptor from a name in the file system. Use the *path* operand to specify the path name of the object in the file system name space, which was previously attached. See `fattach`(3C).

The user must be the owner of the file or a user with the appropriate privileges. All subsequent operations on *path* will operate on the underlying file system entry and not on the STREAMS file. The permissions and status of the entry are restored to the state they were in before the STREAMS file was attached to the entry.

OPERANDS

The following operands are supported:

path Specifies the the path name of the object in the file system name space, which was previously attached.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`fattach`(3C), `fdetach`(3C), `attributes`(5), `streamio`(7I)

STREAMS Programming Guide

NAME

fdisk – create or modify fixed disk partition table

SYNOPSIS

```
fdisk [-o offset] [-s size] [-P fill_patt] [-S geom_file] [-w | r | d | n | I
    | B | t | T | g | G | R] [-F fdisk_file] [ [-v] -W {fdisk_file |
    −}] [-h] [-b masterboot] [-A id : act : bhead : bsect : bcyl : ehead :
    esect : ecyl : rsect : numsect] [-D id : act : bhead: bsect : bcyl : ehead:
    esect : ecyl : rsect : numsect] rdevice
```

DESCRIPTION

This command is used to create and modify the partition table, and to install the master boot (IA only) record that is put in the first sector of the fixed disk. This table is used by the first-stage bootstrap (or firmware) to identify parts of the disk reserved for different operating systems, and to identify the partition containing the second-stage bootstrap (the *active* Solaris partition). The *rdevice* argument must be used to specify the raw device associated with the fixed disk, for example, `/dev/rdsk/c0t0d0p0`.

The program can operate in three different modes. The first is interactive mode. In interactive mode, the program displays the partition table as it exists on the disk, and then presents a menu allowing the user to modify the table. The menu, questions, warnings, and error messages are intended to be self-explanatory.

In interactive mode, if there is no partition table on the disk, the user is given the options of creating a default partitioning or specifying the initial table values. The default partitioning allocates the entire disk for the Solaris system and makes the Solaris system partition active. In either case, when the initial table is created, `fdisk` also writes out the first-stage bootstrap (IA only) code along with the partition table.

The second mode of operation is used for automated entry addition, entry deletion, or replacement of the entire `fdisk` table. This mode can add or delete an entry described on the command line. In this mode the entire `fdisk` table can be read in from a file replacing the original table. `fdisk` can also be used to create this file. There is a command line option that will cause `fdisk` to replace any `fdisk` table with the default of the whole disk for the Solaris system.

The third mode of operation is used for disk diagnostics. In this mode, a section of the disk can be filled with a user specified pattern, and mode sections of the disk can also be read or written.

Menu Options

The menu options for interactive mode given by the `fdisk` program are:

`Create a partition`

This option allows the user to create a new partition. The maximum number of partitions is 4. The program will ask for the type of the partition (SOLARIS, MS-DOS, UNIX, or other). It will then ask for the size of the partition as a percentage of the disk. The user may also enter the letter `c` at this point, in which case the program will ask for the starting cylinder number and size of the partition in cylinders. If a `c` is not entered, the program will determine the starting cylinder number where the partition will fit. In either case, if the partition would overlap an existing partition or will not fit, a message is displayed and the program returns to the original menu.

`Change Active (Boot from) partition`
This option allows the user to specify the partition where the first-stage bootstrap will look for the second-stage bootstrap, otherwise known as the *active* partition.

`Delete a partition`
This option allows the user to delete a previously created partition. Note that this will destroy all data in that partition.

Use the following options to include your modifications to the partition table at this time or to cancel the session without modifying the table:

`Exit`
This option writes the new version of the table created during this session with `fdisk` out to the fixed disk, and exits the program.

`Cancel`
This option exits without modifying the partition table.

OPTIONS

The following options apply to `fdisk`:

`-A` *id:act:bhead:bsect:bcyl:ehead:esect:ecyl:rsect:numsect*
Add a partition as described by the argument (see the `-F` option below for the format). Use of this option will zero out the VTOC on the Solaris partition if the `fdisk` table changes.

`-b` *master_boot*
Specify the file *master_boot* as the master boot program. The default master boot program is `/usr/lib/fs/ufs/mboot`.

`-B`
Default to one Solaris partition that uses the whole disk.

`-d`
Turn on verbose *debug* mode. This will cause `fdisk` to print its state on stderr as it is used. The output from this option should not be used with `-F`.

`-D` *id:act:bhead:bsect:bcyl:ehead:esect:ecyl:rsect:numsect*
Delete a partition as described by the argument (see the `-F` option below for the format). Note that the argument must be an exact match or the entry will not be deleted! Use of this option will zero out the VTOC on the Solaris partition if the `fdisk` table changes.

`-F` *fdisk_file*
Use fdisk file *fdisk_file* to initialize table. Use of this option will zero out the VTOC on the Solaris partition if the `fdisk` table changes.

The *fdisk_file* contains up to four specification lines. Each line is delimited by a new-line character (`\n`). If the first character of a line is an asterisk (*), the line is treated as a comment. Each line is composed of entries that are position-dependent, are separated by ''white space'' or colons, and have the following format:

id act bhead bsect bcyl ehead esect ecyl rsect numsect

where the entries have the following values:

id This is the type of partition and the correct numeric values may be found in `fdisk.h`.

act This is the active partition flag; `0` means not active and `128` means active.

bhead This is the head where the partition starts. If this is set to `0`, `fdisk` will correctly fill this in from other information.

bsect This is the sector where the partition starts. If this is set to `0`, `fdisk` will correctly fill this in from other information.

bcyl This is the cylinder where the partition starts. If this is set to `0`, `fdisk` will correctly fill this in from other information.

ehead This is the head where the partition ends. If this is set to `0`, `fdisk` will correctly fill this in from other information.

esect This is the sector where the partition ends. If this is set to `0`, `fdisk` will correctly fill this in from other information.

ecyl This is the cylinder where the partition ends. If this is set to `0`, `fdisk` will correctly fill this in from other information.

rsect The relative sector from the beginning of the disk where the partition starts. This must be specified and can be used by `fdisk` to fill in other fields.

numsect The size in sectors of this disk partition. This must be specified and can be used by `fdisk` to fill in other fields.

`-g`
Get the label geometry for disk and display on stdout (see the `-S` option for the format).

`-G`
Get the physical geometry for disk and display on stdout (see the `-S` option for the format).

`-h`
Issue verbose message; message will list all options and supply an explanation for each.

`-I`
Forgo device checks. This is used to generate a file image of what would go on a disk without using the device. Note that you must use `-S` with this option (see above).

`-n`
Don't update `fdisk` table unless explicitly specified by another option. If no other options are used, `-n` will only write the master boot record to the disk. In addition, note that `fdisk` will not come up in interactive mode if the `-n` option is specified.

`-o` *offset*
Block offset from start of disk. This option is used for `-P`, `-r`, and `-w`. Zero is assumed when this option is not used.

`-P` *fill_patt*
Fill disk with pattern *fill_patt*. *fill_patt* can be decimal or hex and is used as number for constant long word pattern. If *fill_patt* is `#`, then pattern is block # for each block. Pattern is put in each block as long words and fills each block (see `-o` and `-s`).

`-r`
Read from disk and write to stdout. See `-o` and `-s`, which specify the starting point and size of the operation.

`-R`
Treat disk as read-only. This is for testing purposes.

`-s` *size*
Number of blocks to perform operation on (see `-o`).

`-S` *geom_file*
Set the label geometry to the content of the *geom_file*. The *geom_file* contains one specification line. Each line is delimited by a new-line character (`\n`). If the first character of a line is an asterisk (*), the line is treated as a comment. Each line is composed of entries that are position-dependent, are separated by white space, and have the following format:

pcyl ncyl acyl bcyl nheads nsectors sectsiz

where the entries have the following values:

pcyl	This is the number of physical cylinders for the drive.
ncyl	This is the number of usable cylinders for the drive.
acyl	This is the number of alt cylinders for the drive.
bcyl	This is the number of offset cylinders for the drive (should be zero).
nheads	The number of heads for this drive.
nsectors	The number of sectors per track.
sectsiz	The size in bytes of a sector.

`-t`
Adjust incorrect slice table entries so that they will not cross partition table boundaries.

`-T`
Remove incorrect slice table entries that span partition table boundaries.

`-v`
Output the HBA (virtual) geometry dimensions. This option must be used in conjunction with the `-W` flag. This option will work for platforms which support virtual geometry. (IA only)

`-w`
Write to disk and read from stdin. See `-o` and `-s`, which specify the starting point and size of the operation.

`-W` –
Output the disk table to `stdout`.

`-W` *fdisk_file*
Create an `fdisk` file *fdisk_file* from disk table. This can be used with the `-F` option below.

FILES

`/dev/rdsk/c0t0d0p0` Raw device associated with the fixed disk.

`/usr/lib/fs/ufs/mboot` Default master boot program.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Architecture	IA
Availability	SUNWcsu

SEE ALSO

`uname`(1), `fmthard`(1M), `prtvtoc`(1M) `attributes`(5)

DIAGNOSTICS

Most messages will be self-explanatory. The following may appear immediately after starting the program:

`Fdisk: cannot open <device>`
This indicates that the device name argument is not valid.

`Fdisk: unable to get device parameters for device <device>`
This indicates a problem with the configuration of the fixed disk, or an error in the fixed disk driver.

`Fdisk: error reading partition table`
This indicates that some error occurred when trying initially to read the fixed disk. This could be a problem with the fixed disk controller or driver, or with the configuration of the fixed disk.

`Fdisk: error writing boot record`
This indicates that some error occurred when trying to write the new partition table out to the fixed disk. This could be a problem with the fixed disk controller, the disk itself, the driver, or the configuration of the fixed disk.

NAME

ff – list file names and statistics for a file system

SYNOPSIS

ff [-F *FSType*] [-V] [*generic_options*] [-o *specific_options*] *special*...

DESCRIPTION

ff prints the pathnames and inode numbers of files in the file system which resides on the special device *special*. Other information about the files may be printed using options described below. Selection criteria may be used to instruct ff to only print information for certain files. If no selection criteria are specified, information for all files considered will be printed (the default); the -i option may be used to limit files to those whose inodes are specified.

Output is sorted in ascending inode number order. The default line produced by ff is:

path-name i-number

The maximum information the command will provide is:

path-name i-number size uid

OPTIONS

-F — Specify the *FSType* on which to operate. The *FSType* should either be specified here or be determinable from /etc/vfstab by matching the *special* with an entry in the table, or by consulting /etc/default/fs.

-V — Echo the complete command line, but do not execute the command. The command line is generated by using the options and arguments provided by the user and adding to them information derived from /etc/vfstab. This option may be used to verify and validate the command line.

generic_options — Options that are supported by most FSType-specific modules of the command. The following options are available:

- -I — Do not print the i-node number after each path name.
- -l — Generate a supplementary list of all path names for multiply-linked files.
- -p *prefix* — The specified *prefix* will be added to each generated path name. The default is '.' (dot).
- -s — Print the file size, in bytes, after each path name.
- -u — Print the owner's login name after each path name.

`-a` *-n*	Select if the file has been accessed in *n* days.
`-m` *-n*	Select if the file has been written or created in *n* days.
`-c` *-n*	Select if file's status has been changed in *n* days.
`-n` *file*	Select if the file has been modified more recently than the argument `file`.
`-i` *i-node-list*	Generate names for only those i-nodes specified in *i-node-list*. *i-node-list* is a list of numbers separated by commas (with no intervening spaces).

`-o` Specify *FSType*-specific options in a comma separated (without spaces) list of suboptions and keyword-attribute pairs for interpretation by the *FSType*-specific module of the command.

OPERANDS

special A special device.

USAGE

See `largefile`(5) for the description of the behavior of `ff` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

FILES

`/etc/default/fs` default local file system type. Default values can be set for the following flags in `/etc/default/fs`. For example: `LOCAL=ufs`

`LOCAL` The default partition for a command if no FSType is specified.

`/etc/vfstab` list of default parameters for each file system

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`find`(1), `ncheck`(1M), `stat`(2), `vfstab`(4), `attributes`(5), `largefile`(5) Manual pages for the *FSType*-specific modules of `ff`.

NOTES

This command may not be supported for all *FSTypes*.

The `-a`, `-m`, and `-c` flags examine the `st_atime`, `st_mtime`, and `st_ctime` fields of the `stat` structure respectively. (See `stat`(2).)

NAME

ffbconfig, SUNWffb_config – configure the FFB Graphics Accelerator

SYNOPSIS

```
/usr/sbin/ffbconfig [-dev device-filename] [-res video-mode [now | try]
     [noconfirm | nocheck]] [-file | machine | system] [-deflinear
     | true | false] [-defoverlay | true | false] [-linearorder |
     first | last] [-overlayorder | first | last] [-expvis | enable
     | disable] [-sov | enable | disable] [-maxwids n] [-extovl |
     enable | disable] [-g gamma-correction-value]
     [-gfile gamma-correction-file] [-propt] [-prconf] [-defaults]

/usr/sbin/ffbconfig [-propt ] [-prconf]

/usr/sbin/ffbconfig [-help] [-res ?]
```

DESCRIPTION

`ffbconfig` configures the FFB Graphics Accelerator and some of the X11 window system defaults for FFB.

The first form of `ffbconfig` stores the specified options in the OWconfig file. These options will be used to initialize the FFB device the next time the window system is run on that device. Updating options in the OWconfig file provides persistence of these options across window system sessions and system reboots.

The second and third forms of `ffbconfig`, which invoke only the `-prconf`, `-propt`, `-help`, and `-res` ? options do not update the OWconfig file. Additionally, for the third form all other options are ignored.

Options may be specified for only one FFB device at a time. Specifying options for multiple FFB devices requires multiple invocations of `ffbconfig`.

Only FFB-specific options can be specified through `ffbconfig`. The normal window system options for specifying default depth, default visual class and so forth are still specified as device modifiers on the `openwin` command line. See the *OpenWindows Desktop Reference Manual* for details.

The user can also specify the OWconfig file that is to be updated. By default, the machine-specific file in the `/etc/openwin` directory tree is updated. The `-file` option can be used to specify an alternate file to use. For example, the system-global OWconfig file in the `/usr/openwin` directory tree can be updated instead.

Both of these standard OWconfig files can only be written by root. Consequently, the `ffbconfig` program, which is owned by the root user, always runs with setuid root permission.

OPTIONS

`-dev` *device-filename*
Specifies the FFB special file. The default is `/dev/fbs/ffb0`.

`-file machine |system`
Specifies which OWconfig file to update. If `machine`, the machine-specific OWconfig file in the `/etc/openwin` directory tree is used. If `system`, the global OWconfig file in the `/usr/openwin` directory tree is used. If the file does not exist, it is created.

`-res` *video-mode* [`now` | `try` [`noconfirm` | `nocheck`]]

Specifies the video mode used to drive the monitor connected to the specified FFB device.

video-mode has the format of *width*x*height*x*rate* where *width* is the screen width in pixels, *height* is the screen height in pixels, and *rate* is the vertical frequency of the screen refresh.

The `s` suffix, as in `960x680x112s` and `960x680x108s`, indicates stereo video modes. The `i` suffix, as in `640x480x60i` and `768x575x50i`, indicates interlaced video timing. If absent, non-interlaced timing will be used.

`-res` (the third form in the `SYNOPSIS`) also accepts formats with `@` (at sign) in front of the refresh rate instead of `x`. `1280x1024@76` is an example of this format.

Some video-modes are supported only on certain revisions of FFB. Also, some video-modes, supported by FFB, may not be supported by the monitor. The list of video-modes supported by the FFB device and the monitor can be obtained by running `ffbconfig` with the `-res` ? option.

The following table lists all possible video modes supported on FFB:

Name	Description
`1024x768x60`	
`1024x768x70`	
`1024x768x75`	
`1024x768x77`	
`1024x800x84`	
`1152x900x66`	
`1152x900x76`	
`1280x800x76`	
`1280x1024x60`	
`1280x1024x67`	
`1280x1024x76`	
`960x680x112s`	(stereo)
`960x680x108s`	(stereo)
`640x480x60`	
`640x480x60i`	(interlaced)

Name	Description
768x575x50i	(interlaced)
1440x900x76	(hi-res)
1600x1000x66	(hi-res)
1600x1000x76i	(hi-res)
1600x1280x76	(hi-res)
1920x1080x72	(hi-res)
1920x1200x70	(hi-res)

Symbolic names

For convenience, some video modes have symbolic names defined for them. Instead of the form *width*x*height*x*rate*, one of these names may be supplied as the argument to -res. The meaning of the symbolic name none is that when the window system is run the screen resolution will be the video mode that is currently programmed in the device.

Name	Corresponding Video Mode
svga	1024x768x60
1152	1152x900x76
1280	1280x1024x76
stereo	960x680x112s
ntsc	640x480x60i
pal	768x575x50i
none	(video mode currently programmed in device)

The -res option also accepts additional, optional arguments immediately following the video mode specification. Any or all of these may be present.

now
Specifies that the FFB device will be immediately programmed to display this video mode, in addition to updating the video mode in the OWconfig file. This option is useful for changing the video mode before starting the window system.

It is inadvisable to use this suboption with ffbconfig while the configured device is being used (for example, while running the window system); unpredictable results may occur. To run ffbconfig with the now suboption, first bring the

window system down. If the `now` suboption is used within a window system session, the video mode will be changed immediately, but the width and height of the affected screen won't change until the window system is exited and re-entered. In addition, the system may not recognize changes in stereo mode. Consequently, this usage is strongly discouraged.

`noconfirm`
Instructs `ffbconfig` to bypass confirmation and and warning messages and to program the requested video mode anyway.

Using the `-res` option, the user could potentially put the system into an usable state, a state where there is no video output. This can happen if there is ambiguity in the monitor sense codes for the particular code read. To reduce the chance of this, the default behavior of `ffbconfig` is to print a warning message to this effect and to prompt the user to find out if it is okay to continue. This option is useful when `ffbconfig` is being run from a shell script.

`nocheck`
Suspends normal error checking based on the monitor sense code. The video mode specified by the user will be accepted regardless of whether it is appropriate for the currently attached monitor. This option is useful if a different monitor is to be connected to the FFB device. Note: Use of this option implies `noconfirm` as well.

`try`
Programs the specified video mode on a trial basis. The user will be asked to confirm the video mode by typing `y` within 10 seconds. The user may also terminate the trial before 10 seconds are up by typing any character. Any character other than `y` or RETURN is considered a `no` and the previous video mode will be restored and `ffbconfig` will not change the video mode in the OWconfig file and other options specified will still take effect. If a RETURN is pressed, the user is prompted for a yes or no answer on whether to keep the new video mode. This option implies the now suboption (see the warning note on the now suboption).

`-deflinear true | false`
FFB possesses two types of visuals: linear and nonlinear. Linear visuals are gamma corrected and nonlinear visuals are not. There are two visuals that have both linear and nonlinear versions: 24-bit TrueColor and 8-bit StaticGray.

`-deflinear true` sets the default visual to the linear visual that satisfies other specified default visual selection options. Specifically, the default visual selection options are those set by the `Xsun` (1) `defdepth` and `defclass` options. See *OpenWindows Desktop Reference Manual* for details.

`-deflinear false` (or if there is no linear visual that satisfies the other default visual selection options) sets the default visual to t the non-linear visual as the default.

This option cannot be used when the `-defoverlay` option is present, because FFB does not possess a linear overlay visual.

`-defoverlay true | false`
FFB provides an 8-bit PseudoColor visual whose pixels are disjoint from the rest of the FFB visuals. This is called the overlay visual. Windows created in this visual will not damage windows created in other visuals. The converse, however, is not true. Windows created in other visuals will damage overlay windows. This visual has `256 maxwids` of opaque color values. See `-maxwids` in `OPTIONS`.

If `-defoverlay` is `true`, the overlay visual will be made the default visual. If `-defoverlay` is `false`, the nonoverlay visual that satisfies the other default visual selection options, such as `defdepth` and `defclass`, will be chosen as the default visual. See the *OpenWindows Desktop Reference Manual* for details.

Whenever `-defoverlay true` is used, the default depth and class chosen on the openwin command line must be 8-bit PseudoColor. If not, a warning message will be printed and the `-defoverlay` option will be treated as false. This option cannot be used when the `-deflinear` option is present, because FFB doesn't possess a linear overlay visual.

`-linearorder first | last`
If `first`, linear visuals will come before their non-linear counterparts on the X11 screen visual list for the FFB screen. If `last`, the nonlinear visuals will come before the linear ones.

`-overlayorder first | last`
If `true`, the depth 8 PseudoColor Overlay visual will come before the non-overlay visual on the X11 screen visual list for the FFB screen. If `false`, the non-overlay visual will come before the overlay one.

`-expvis enable | disable`
If enabled, OpenGL Visual Expansion will be activated. Multiple instances of selected visual groups (8-bit PseudoColor, 24-bit TrueColor and so forth) can be found in the screen visual list.

`-sov enable | disable`
Advertises the root window's `SERVER_OVERLAY_VISUALS` property. SOV visuals will be exported and their transparent types, values and layers can be retrieved through this property. If `-sov disable` is specified, the `SERVER_OVERLAY_VISUALS` property will not be defined. SOV visuals will not be exported.

`-maxwids` *n*
Specifies the maximum number of FFB X channel pixel values that are reserved for use as window sIDs (WIDs). The remainder of the pixel values in overlay colormaps are used for normal X11 opaque color pixels. The reserved WIDs are allocated on a first-come first-serve basis by 3D graphics windows (such as XGL), MBX windows, and windows that have a non-default visual. The X channel codes `0` to (`255`-*n*) will be opaque color pixels. The X channel codes (`255`-*n*+`1`) to `255` will be reserved for use as WIDs. Legal values on FFB, FFB2 are: `1`, `2`, `4`, `8`, `16`, and `32`. Legal values on FFB2+ are: `1`, `2`, `4`, `8`, `16`, `32`, and `64`.

`-extovl enable | disable`
This option is available only on FFB2+. If enabled, extended overlay is available. The overlay visuals will have 256 opaque colors. The SOV visuals will have 255 opaque colors and 1 transparent color. This option enables hardware supported transparency which provides better performance for windows using the SOV visuals.

`-g` *gamma-correction value*
This option is available only on FFB2+. This option allows changing the gamma correction value. All linear visuals provide gamma correction. By default the gamma correction value is 2.22. Any value less than zero is illegal. The gamma correction value is applied to the linear visual, which then has an effective gamma value of 1.0, which is the value returned by `XSolarisGetVisualGamma`(3). See `XSolarisGetVisualGamma`(3) for a description of that function.

This option can be used while the window system is running. Changing the gamma correction value will affect all the windows being displayed using the linear visuals.

`-gfile` *gamma-correction file*
This option is available only on FFB2+. This option loads gamma correction table from the specified file. This file should be formatted to provide the gamma correction values for R, G and B channels on each line. This file should provide 256 triplet values, each in hexadecimal format and separated by at least 1 space. Following is an example of this file:

```
0x00 0x00 0x00
0x01 0x01 0x01
0x02 0x02 0x02
...
...
0xff 0xff 0xff
```

Using this option, the gamma correction table can be loaded while the window system is running. The new gamma correction will affect all the windows being displayed using the linear visuals. Note, when gamma correction is being done using user specified table, the gamma correction value is undefined. By default, the window system assumes a gamma correction value of 2.22 and loads the gamma table it creates corresponding to this value.

`-defaults`
Resets all option values to their default values.

`-propt`
Prints the current values of all FFB options in the OWconfig file specified by the `-file` option for the device specified by the `-dev` option. Prints the values of options as they will be in the OWconfig file after the call to `ffbconfig` completes. The following is a typical display using the `-propt` option:

```
--- OpenWindows Configuration for /dev/fbs/ffb0 ---
OWconfig: machine
Video Mode: NONE
Default Visual: Non-Linear Normal Visual
```

```
Visual Ordering: Linear Visuals are last
                 Overlay Visuals are last
OpenGL Visuals: disabled
SOV: disabled
Allocated WIDs: 32
```

`-prconf`
Prints the FFB hardware configuration. The following is a typical display using the `-prconf` option:

```
--- Hardware Configuration for /dev/fbs/ffb0 ---
Type: double-buffered FFB2 with Z-buffer
Board: rev x
PROM Information: @(#)ffb2.fth x.x xx/xx/xx
FBC: version x
DAC: Brooktree 9068, version x
3DRAM: Mitsubishi 1309, version x
EDID Data: Available - EDID version 1 revision x
Monitor Sense ID: 4  (Sun 37x29cm RGB color monitor)
Monitor possible resolutions: 1024x768x60, 1024x768x70,
      1024x768x75, 1152x900x66, 1152x900x76,
      1280x1024x67, 1280x1024x76, 960x680x112s,
      640x480x60
Current resolution setting: 1280x1024x76
```

`-help`
Prints a list of the `ffbconfig` command line options, along with a brief explanation of each.

DEFAULTS

For a given invocation of `ffbconfig` command line if an option does not appear on the command line, the corresponding OWconfig option is not updated; it retains its previous value.

When the window system is run, if an FFB option has never been specified via `ffbconfig`, a default value is used. The option defaults are listed in the following table:

Option	Default
`-dev`	`/dev/fbs/ffb0`
`-file`	`machine`
`-res`	`none`
`-deflinear`	`false`
`-defoverlay`	`false`
`-linearorder`	`last`

Option	Default
-overlayorder	last
-expvis	enabled
-sov	enabled
-maxwids	32

The default for the -res option of none means that when the window system is run the screen resolution will be the video mode that is currently programmed in the device.

This provides compatibility for users who are used to specifying the device resolution through the PROM. On some devices (for example, GX) this is the only way of specifying the video mode. This means that the PROM ultimately determines the default FFB video mode.

EXAMPLES

EXAMPLE 1 Changing The Monitor Type

The following example switches the monitor type to the resolution of 1280 × 1024 at 76 Hz:

```
example% /usr/sbin/ffbconfig -res 1280x1024x76
```

FILES

/dev/fbs/ffb0 device special file

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWffbcf

SEE ALSO

mmap(2), attributes(5), fbio(7I), ffb(7D)

OpenWindows Desktop Reference Manual

NAME

ff_ufs – list file names and statistics for a ufs file system

SYNOPSIS

ff `-F ufs [`*generic_options*`] [-o a,m,s]` *special*...

DESCRIPTION

`ff` prints the pathnames and inode numbers of files in the file system which resides on the special device *special*.

See `ff`(1M) for information regarding the `ff` command. See `OPTIONS` for information regarding the ufs-specific options.

OPTIONS

The following options are supported:

`-o` Specify `ufs` file system specific options. The following options available are:

- `a` Print the '.' and '. .' directory entries.
- `m` Print mode information. This option must be specified in conjunction with the `-i` *i-node-list* option (see `ff`(1M)).
- `s` Print only special files and files with set-user-ID mode.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`find`(1), `ff`(1M), `ncheck`(1M), `attributes`(5)

NAME

flar – administer flash archives

SYNOPSIS

flar -i [-l] [-k *keyword*] [-t [-p *posn*] [-b *blocksize*]] *archive*

flar -c [-d *dir*] [-u *section*] [-t [-p *posn*] [-b *blocksize*]] *archive*

flar -s [-d *dir*] [-u *section*] [-f] [-S *section*] [-t [-p *posn*] [-b *blocksize*]] *archive*

DESCRIPTION

The `flar` command is used to administer flash archives. A flash archive is an easily transportable version of a reference configuration of the Solaris operating environment, plus optional other software. Such an archive is used for the rapid installation of Solaris on large numbers of machines. You use the `flarcreate`(1M) to create a flash archive.

The `flar` command includes subcommands for extracting information, splitting archives, and combining them. Subcommands are indicated by the first option in a `flar` command line. These options are as follows:

`-c` Combine the individual sections that make up an archive into the archive. If *dir* is specified (see `-d` option below), the sections will be gathered from *dir*; otherwise, they will be gathered from the current directory. Each section is assumed to be in a separate file, the names of which are the section names. At a minimum, the archive cookie (`cookie`), archive identification (`identification`), and archive files (`archive`) sections must be present. If `archive` is a directory, its contents are archived using `cpio` prior to inclusion in the archive. If so specified in the `identification` section, the contents are compressed.

Note that no validation is performed on any of the sections. In particular, no fields in the `identification` section are validated or updated. See `flash_archive`(4) for a description of the archive sections.

`-i` Extract information on an archive. This subcommand is analogous to `pkginfo`.

`-s` Split an archive into one file for each section of the archive. Each section is copied into a separate file in *dir*, if *dir* is specified (see `-d` option below), or the current directory if it is not. The files resulting from the split are named after the sections. The archive cookie is stored in a file named `cookie`. If *section* is specified (see `-u` option below), only the named section is copied.

The options for each subcommand are described below.

OPTIONS

The options for `flar -i` (extract info) subcommand are as follows:

`-k` *keyword* Only the value of the keyword *keyword* is returned.

`-l` List all files in the archive. Does not process content from any sections other than the `archive` section.

The following are `flar -i` options used with tape archives:

`-b` *blocksize* — The block size to be used when creating the archive. If not specified, a default block size of 64K is used.

`-p` *posn* — Specifies the position on the tape device where the archive should be created. If not specified, the current position of the tape device is examined.

`-t` — The archive to be analyzed is located on a tape device. The path to the device is specified by *outfile* (see OPERANDS).

The options for `flar -s` and `-c` (split and combine archives) subcommands are as follows:

`-d` *dir* — Retrieve sections from *dir*, rather than from the current directory.

`-f` — (Used with `-s` only.) Extract the `archive` section into directory called `archive`, rather than placing it in a file of the same name as the section.

`-S` *section* — (Used with `-s` only.) Extract only the section named *section* from the archive.

`-u` *section* — Appends *section* to the list of sections to be included. The default list includes the `cookie`, `identification`, and `archive` sections. *section* can be a single section name or a space-separated list of section names.

The following options are used with tape archives (with both `-c` and `-s`):

`-b` *blocksize* — The block size to be used when creating the archive. If not specified, a default block size of 64K is used.

`-p` *posn* — Used only with `-t`. Specifies the position on the tape device where the archive should be created. If not specified, the current position of the tape device is used.

`-t` — Create an archive on or read an archive from a tape device. The *outfile* operand (see OPERANDS) is assumed to be the name of the tape device.

OPERANDS

The following operand is supported:

outfile — Path to tape device if the `-t` option was used. Otherwise, the path to a flash archive.

EXIT STATUS

The following exit values are returned for the `-c` and `-s` subcommands:

`0` — Successful completion.

`>0` — An error occurred.

The following exit values are returned for the `-i` subcommand:

`0` — Successful completion.

`1` — Command failed. If the `-k` option is used and the requested keyword is not found, `flar` returns `2`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWinst

SEE ALSO

`flarcreate`(1M), `flash_archive`(4), `attributes`(5)

NAME

flarcreate – create a flash archive from a master system

SYNOPSIS

```
flarcreate -n name [-R root] [-S] [-H] [-c] [-x exclude] [-t [-p posn]
     [-b blocksize]] [-i date] [-m master] [-u section ... [-d dir]] [-U
     key=val ...] [-f [filelist | -] [-F]] [-a author] [-e descr | -E
     descr_file] [-T type] outfile
```

DESCRIPTION

The `flarcreate` command creates a flash archive from a master system. A master system is one that contains a reference configuration, which is a particular configuration of the Solaris operating environment, plus optional other software. A flash archive is an easily transportable version of the reference configuration.

You can run `flarcreate` in multi- or single-user mode. You can also use the command when the master system is booted from the first Solaris software CD or from a Solaris net image.

Archive creation should be performed when the master system is in as stable a state as possible. Following archive creation, use the `flar`(1M) command to administer a flash archive.

See `flash_archive`(4) for a description of the flash archive.

OPTIONS

The `flarcreate` command has the following general options:

`-c`	Compress the archive using `compress`(1)
`-f` *filelist*	Use the contents of *filelist* as a list of files to include in the archive. The files are included in addition to the normal file list, unless `-F` is specified (see below). If *filelist* is `-`, the list is taken from standard input.
`-F`	Include only files in the list specified by `-f`. This option makes `-f` *filelist* an absolute list, rather than a list that is appended to the normal file list.
`-H`	Do not generate hash identifier.
`-R` *root*	Create the archive from the filesystem tree rooted at *root*. If you do not specify this option, `flarcreate` creates an archive from a filesystem rooted at `/`.
`-S`	Do not include sizing information in the archive.
`-U` *key=val...*	Include the user-defined keyword(s) and values in the archive identification section.
`-x` *exclude*	Exclude the directory *exclude* from the archive. Note that the *exclude* directory is assumed to be relative to the alternate root specified using `-R`.

Use the following options with user-defined sections.

`-d` *dir*	Retrieve the section file specified with `-u` from *dir*.

`-u` *section* ... Included the user-defined section located in the file *section* in the archive. *section* must be a blank-separated list of section names as described in `flash_archive`(4).

Use the following options with tape archives.

`-b` *blocksize* The block size to be used when creating the archive. If not specified, a default block size of 64K is used.

`-p` *posn* Used only with `-t`. Specifies the position on the tape device where the archive should be created. If not specified, the current position of the tape device is used.

`-t` Create an archive on a tape device. The *outfile* operand (see OPERANDS) is assumed to be the name of the tape device.

The following options are used for archive identification.

`-a` *author* *author* is used to provide an author name for the archive identification section. If you do not specify `-a`, no author name is included in the identification section.

`-e` *descr* The description to be included in the archive as the value of the `content_description` archive identification key. This option is incompatible with `-E`.

`-E` *descr_file* The description to be used as the value of the archive identification `content_description` key is retrieved from the file *descr_file*. This option is incompatible with `-e`.

`-i` *date* By default, the value for the `creation_date` field in the identification section is generated automatically, based on the current system time and date. If you specify the `-i` option, *date* is used instead.

`-m` *master* By default, the value for the `creation_master` field in the identification section is the name of the system on which you run `flarcreate`, as reported by `uname -n`. If you specify `-m`, *master* is used instead.

`-n` *name* This option is required if you are using `flarcreate` for archive identification. The value of *name* is used to identify the archive. It is also used as the value of the `content_name` archive identification flag.

`-T` *type* Content type included in the archive as the value of the `content_type` archive identification key. If you do not specify `-T`, the `content_type` keyword is not included.

OPERANDS The following operands are supported:

archive Path to the flash archive.

outfile Name of the tape device if the `-t` option is used.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWinst

SEE ALSO

`flar`(1M), `flash_archive`(4), `attributes`(5)

NAME

fmthard – populate VTOC on hard disks

SPARC

fmthard -d *data* | -n *volume_name* | -s *datafile* [-i] /dev/rdsk/c? [t?] d?s2

IA

fmthard -d *data* | -n *volume_name* | -s *datafile* [-i] [-p *pboot*] [-b *bootblk*] /dev/rdsk/c? [t?] d?s2

DESCRIPTION

The `fmthard` command updates the VTOC (Volume Table of Contents) on hard disks and, on IA systems, adds boot information to the Solaris `fdisk` partition. One or more of the options `-s` *datafile*, `-d` *data*, or `-n` *volume_name* must be used to request modifications to the disk label. To print disk label contents, see `prtvtoc`(1M). The `/dev/rdsk/c?`[`t?`]`d?s2` file must be the character special file of the device where the new VTOC is to be installed. On IA systems, `fdisk`(1M) must be run on the drive before `fmthard`.

If you are using an IA system, note that the term ''partition'' in this page refers to *slices* within the IA `fdisk` partition on IA machines. Do not confuse the partitions created by `fmthard` with the partitions created by `fdisk`.

OPTIONS

The following options are supported:

`-d` *data*
: The *data* argument of this option is a string representing the information for a particular partition in the current VTOC. The string must be of the format *part:tag:flag:start:size* where *part* is the partition number, *tag* is the ID TAG of the partition, *flag* is the set of permission flags, *start* is the starting sector number of the partition, and *size* is the number of sectors in the partition. See the description of the *datafile* below for more information on these fields.

`-i`
: This option allows the command to create the desired VTOC table, but prints the information to standard output instead of modifying the VTOC on the disk.

`-n` *volume_name*
: This option is used to give the disk a *volume_name* up to 8 characters long.

`-s` *datafile*
: This option is used to populate the VTOC according to a *datafile* created by the user. If the *datafile* is "–", `fmthard` reads from standard input. The *datafile* format is described below. This option causes all of the disk partition timestamp fields to be set to zero.

  Every VTOC generated by `fmthard` will also have partition 2, by convention, that corresponds to the whole disk. If the input in *datafile* does not specify an entry for partition 2, a default partition 2 entry will be created automatically in VTOC with the tag `V_BACKUP` and size equal to the full size of the disk.

The *datafile* contains one specification line for each partition, starting with partition 0. Each line is delimited by a new-line character (`\n`). If the first character of a line is an asterisk (*), the line is treated as a comment. Each line is composed of entries that are position-dependent, separated by "white space" and having the following format:

partition tag flag starting_sector size_in_sectors

where the entries have the following values:

partition	The partition number. Currently, for Solaris SPARC, a disk can have up to 8 partitions, `0–7`. Even though the *partition* field has 4 bits, only 3 bits are currently used. For IA, all 4 bits are used to allow slices `0–15`. Each Solaris `fdisk` partition can have up to 16 slices.
tag	The partition tag: a decimal number. The following are reserved codes: `0` (`V_UNASSIGNED`), `1` (`V_BOOT`), `2` (`V_ROOT`), `3` (`V_SWAP`), `4` (`V_USR`), `5` (`V_BACKUP`), `6` (`V_STAND`), `7` (`V_VAR`), and `8` (`V_HOME`).
flag	The flag allows a partition to be flagged as unmountable or read only, the masks being: `V_UNMNT 0x01`, and `V_RONLY 0x10`. For mountable partitions use `0x00`.
starting_sector	The sector number (decimal) on which the partition starts.
size_in_sectors	The number (decimal) of sectors occupied by the partition.

You can save the output of a `prtvtoc` command to a file, edit the file, and use it as the *datafile* argument to the `-s` option.

IA Options

The functionality provided by the following two IA options is also provided by `installboot`(1M). Because the functionality described here may be removed in future versions of `fmthard`, you should use `installboot` to install boot records. The following options currently apply to `fmthard`:

`-p` *pboot* — This option allows the user to override the default partition boot file, `/usr/platform/`*platform-name*`/lib/fs/ufs/pboot`. The partition boot file is platform dependent, where *platform-name* can be determined using the `-i` option to `uname`(1).

`-b` *bootblk* — This option allows the user to override the default *bootblk* file, `/usr/platform/`*platform-name*`/lib/fs/ufs/bootblk`. The boot block file is platform dependent, where *platform-name* can be determined using the `-i` option to `uname`(1).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`uname`(1), `format`(1M), `prtvtoc`(1M), `attributes`(5)

IA Only

`fdisk`(1M), `installboot`(1M)

NOTES

Special care should be exercised when overwriting an existing VTOC, as incorrect entries could result in current data being inaccessible. As a precaution, save the old VTOC.

`fmthard` cannot write a disk label on an unlabeled disk. Use `format`(1M) for this purpose.

NAME

fncheck – check for consistency between FNS data and NIS+ data

SYNOPSIS

fncheck [-r] [-s] [-u] [-t *type*] [*domain_name*]

DESCRIPTION

fncheck is used for checking for inconsistencies between FNS username or hostname contexts and the contents of the corresponding NIS+ *passwd.org_dir* or *hosts.org_dir* tables, respectively, in the NIS+ domain *domain_name*. If *domain_name* is omitted, the domain name of the current machine is used. By default (in the absense of the -r and -s options), the following inconsistencies are displayed:

- items that appear only in the FNS context but do not appear in the NIS+ table,
- items that appear only in the NIS+ table but do not appear in the FNS context.

OPTIONS

-r — Display only items that appear in the FNS context but do not appear in the corresponding NIS+ table.

-s — Display items that appear in the NIS+ table but do not appear in the corresponding FNS context.

-u — Update the FNS context based on information in the corresponding NIS+ table. If the -r option is used, items that appear only in the FNS context are removed from the FNS context. If the -s option is used, items that appear only in the NIS+ table are added to the FNS context. If neither -r or -s are specified, items are added and removed from the FNS context to make it consistent with the corresponding NIS+ table.

-t *type* — Specify the type of context to check. type can be either hostname or username. If this option is omitted, both hostname and username contexts are checked. If type is hostname, the FNS hostname context is checked against the NIS+ *hosts.org_dir* table. If type is username, the FNS username context is checked against the NIS+ *passwd.org_dir* table.

USAGE

Although fncheck can be used to add users and hosts to the username and hostname contexts as new users and hosts are added to NIS+, that is not its intended purpose. fncheck is an expensive operation because it makes complete comparisons of the NIS+ table and the corresponding FNS context. When a user or host is added or removed from NIS+ using admintool (see admintool(1M)), it automatically updates the appropriate FNS contexts.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWfns

SEE ALSO

admintool(1M), fncreate(1M), fndestroy(1M), nis(1), attributes(5), fns(5), fns_policies(5)

NAME

fncopy – copy FNS contexts, possibly from one naming service to another naming service

SYNOPSIS

```
fncopy [-f filename] [-i old-naming-service] [-o new-naming-service]
     old-fns-context  new-fns-context
```

DESCRIPTION

`fncopy` copies recursively the FNS context, *old-fns-context*, and attributes to a new FNS context, *new-fns-context*. If `-i` and `-o` options are specified with the respective naming service, the *old-fns-context* with be resolved using *old-naming-service* as the underlying naming service, and *new-fns-context* will be created using *new-naming-service* as the underlying naming service. In the absence of `-i` and `-o` options, the default naming service will be used (see `fnselect`(1M)).

When the `-f` option is used, *filename* names a file containing a list of contexts in the *old-fns-context* that should be copied to the *new-fns-context*.

If the FNS context *new-fns-context* already exists in the target naming service, *new-naming-service*, this command will copy only the contexts and bindings that do not exist in the target naming service. This command will not over-write any of the existing FNS contexts in the target naming service.

This command follows links and copies FNS contexts and binding to the *new-fns-context* namespace.

OPTIONS

The following options are supported:

`-f` *filename*	Specifies a file name that contains a list of FNS contexts to be copied.
`-i` *old-naming-service*	Specifies the source naming service; currently only `nis` is supported.
`-o` *new-naming-service*	Specifies the target naming service; currently only `nisplus` is supported.

OPERANDS

The following operands are supported:

new-fns-context	The new FNS context.
old-fns-context	The current FNS context.

EXAMPLES

EXAMPLE 1 Copying the `fncopy` FNS Printer Context

The following command copies the FNS printer context `.../fed-naming.eng.sun.com/service/printer` and its subcontexts and bindings to the FNS printer context `.../sun.com/orgunit/ssi.eng/service/printer`.

```
% fncopy .../fed-naming.eng.sun.com/service/printer \
     .../sun.com/orgunit/ssi.eng/service/printer
```

EXAMPLE 2 Copying the NIS FNS Users' Contexts

The following command copies the NIS FNS users' contexts specified in the file `/etc/ssi-users-list` to NIS+ FNS users' context of the orgunit `ssi.eng`:

```
% fncopy -i nis -o nisplus -f /etc/ssi-users-list \
     thisorgunit/user org/ssi.eng/user
```

EXIT STATUS

`0` Operation was successful.

`1` Operation failed.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWfns

SEE ALSO

`fnbind`(1), `fnunbind`(1), `fncreate`(1M), `fncreate_fs`(1M), `fncreate_printer`(1M), `fndestroy`(1M), `attributes`(5), `fns`(5)

NAME

fncreate – create an FNS context

SYNOPSIS

fncreate -t *context_type* [-Dosv] [-f *input_file*] [-r *reference_type*] *composite_name*

DESCRIPTION

`fncreate` creates an FNS context of type *context_type*, where a *context_type* must be one of `org`, `hostname`, `host`, `username`, `user`, `service`, `fs`, `site`, `nsid`, or `generic`. It takes as the last argument a composite name, *composite_name*, for the context to be created. In addition to creating the context named, `fncreate` also creates subcontexts of the named context using FNS Policies of what types of contexts should be bound in those contexts. See `fns_policies`(5).

`fncreate` discovers which naming service is in use and creates contexts in the appropriate naming service. When FNS is being initially set up, it will by default create contexts for NIS+. This default can be changed by the use of `fnselect`(1M) to explicitly select a naming service.

When using FNS for a NIS+ environment, `fncreate` creates NIS+ tables and directories in the NIS+ hierarchy. See `fns_nis+`(5) for more information on the necessary NIS+ credentials and the use of the environment variable `NIS_GROUP` when using `fncreate` and other FNS commands.

When using FNS for a NIS environment, `fncreate` creates NIS maps and hence must be executed as superuser on the NIS master of the FNS-related maps. See `fns_nis`(5) for more information specific to the use of FNS in a NIS environment.

When using FNS for an environment that uses `/etc` files for its naming information, `fncreate` creates files in the `/var/fn` directory. See `fns_files`(5) for more information specific to the use of FNS for files.

OPTIONS

The following options are supported:

`-D` — Information about the creation of a context, and corresponding NIS+ directories and tables, or NIS maps, or files entry, is displayed as each context is created.

`-f` *input_file* — Create a context for every user or host listed in `input_file`. This option is only applicable when used with the `-t username` or `-t hostname` options. The format of the file is an atomic user name or host name per line. This option is used to create contexts for a subset of the `users/hosts` found in the corresponding `passwd` or `hosts` database of the naming service (that is, for NIS+ these are the `passwd.org_dir` or `hosts.org_dir` tables, respectively). If this option is omitted, `fncreate` creates a context for every `user/host` found in the corresponding `passwd` or `hosts` database.

`-o`

Only the context named by *composite_name* is created; no subcontexts are created. When this option is omitted, subcontexts are created according to the FNS Policies for the type of the new object.

`-t` *context_type*

The following are valid entries for *context_type*:

`org`

Create organization context, and default subcontexts, for an existing NIS+ domain, NIS domain, or `/etc` files environment.

For NIS+, *composite_name* is of the form *org/domain/* where *domain* is a NIS+ domain. An empty domain name indicates the creation of the organization context for the root NIS+ domain; otherwise, the domain name names the corresponding NIS+ domain. *domain* can be either the fully-qualified NIS+ domain name — dot ('.')-terminated — or the NIS+ domain name named relative to the NIS+ root domain.

For example, the following creates the root organization context and its subcontexts for the NIS+ root domain `Wiz.Com.`:

```
eg% fncreate -t org org//
```

The same thing could have been achieved using the following command:

```
eg% fncreate -t org org/Wiz.COM./
```

Typically, this is the first FNS context created.

To create the organization context for a subdomain of `Wiz.COM.`, execute either of the following commands:

```
eg% fncreate -t org org/sales/
```

or

```
eg% fncreate -t org \
org/sales.Wiz.COM./
```

Note that if the corresponding NIS+ domain does not exist, `fncreate` fails. See `nissetup`(1M) for setting up a NIS+ domain.

A `ctx_dir` directory is created under the directory of the organization named.

For NIS or an /etc files environment, *domain* should be NULL (empty) because NIS and /etc files do not support a hierarchy namespace of domains. For example, the following command creates the organization context for the NIS or /etc files environment:

```
eg% fncreate -t org org//
```

For NIS+, NIS, and /etc files, creating the organization context also creates the organization's immediate subcontexts host, user, and service and their subcontexts. This includes a context for every host entry in the corresponding hosts database of the naming service (that is, hosts.org_dir NIS+ table, or hosts NIS map, or /etc/hosts file), and a context for every user entry in the passwd database of the naming service (that is, passwd.org_dir NIS+ table, or passwd NIS map, or /etc/passwd file) unless the option -o is specified. Bindings for these subcontexts are recorded under the organization context.

hostname
: Create a hostname context in which atomic host names can be bound, and bind the reference of the context to *composite_name*. If the suffix of *composite_name* is host/, the hostname context created is also bound to the composite name with this suffix replaced by _host/, and the reverse (that is, if a composite name with a _host/ suffix was supplied, a binding would be created for host/). Also create a host context for every host entry in the corresponding hosts database of the naming service (hosts.org_dir NIS+ table, or hosts NIS map, or /etc/hosts file), unless either option -o or -f is specified. The following example creates host contexts for all hosts in the sales organization:

```
eg% fncreate -t hostname \
org/sales/host/
```

Typically, a hostname context need not be created explicitly since it is created by default, as a subcontext under org.

host
: Create a host context for a specific host, and its service and fs subcontexts, and bind the reference of the context to *composite_name*. For

example, the following creates a host context and `service` and `fs` subcontexts for host `sylvan`:

```
eg% fncreate -t host \
org/sales/host/sylvan/
```

`username`
Create a username context in which atomic user names can be bound, and bind the reference of the context to *composite_name*. If the suffix of *composite_name* is `user/`, the username context created is also bound to the composite name with this suffix replaced by `_user/`, and the reverse. Also create a user context for every user entry in the corresponding `passwd` database of the naming service (that is, `passwd.org_dir` NIS+ table, or `passwd` NIS map, or `/etc/passwd` file), unless either the option `-o` or `-f` is specified. The following example creates username contexts for all users in the `sales` organization:

```
eg% fncreate -t username \
org/sales/user/
```

Typically, a username context need not be created explicitly since it is created by default, as a subcontext under `org`.

`user`
Create a user context for a specific user, and its `service` and `fs` subcontexts, and bind the reference of the context to *composite_name*. For example, the following creates a user context and `service` and `fs` subcontexts for user `jsmith`:

```
eg% fncreate -t user \
org/sales/user/jsmith/
```

`service`
Create a service context in which slash-separated left-to-right service names can be bound, and bind the reference of the context to *composite_name*. If the suffix of *composite_name* is `service/`, the service context created is also bound to the composite name with this suffix replaced by `_service/`, and the reverse. Typically, a service context need not be created explicitly since it is created by default, as a subcontext under `org`, `host`, or `user` contexts.

`fs`
Create a file system context for a user or host, and bind the reference of the context to *composite_name*.

The composite name must be the name of a host or a user, with either `fs/` or `_fs/` appended to it. If the suffix of *composite_name* is `fs/`, the file system context created is also bound to the composite name with this suffix replaced by `_fs/`, and the reverse.

Typically, a file system context need not be created explicitly since it is created by default, as a subcontext of a user or host context.

The file system context of a user is the user's home directory as stored in the `passwd` database of the naming service (that is, in NIS+ table `passwd.org_dir`, or `passwd` NIS map, or `/etc/passwd` file). The file system context of a host is the set of NFS file systems that the host exports.

Use the `fncreate_fs`(1M) command to create file system contexts for organizations and sites, or to create file system contexts other than the defaults for users and hosts.

`site`
Create a site context in which dot-separated right-to-left site names can be bound, and a service subcontext, and bind the reference of the context to *composite_name*. If the suffix of *composite_name* is `site/`, the hostname context created is also bound to the composite name with this suffix replaced by `_site/`, and the reverse. Typically, a `site` context is created at the same level as the `org` context and is used for creating a geographical namespace that complements the organizational namespace of an enterprise.

`nsid`
Create a context in which namespace identifiers can be bound. This context has a flat namespace, in which only atomic names can be bound. An example of such a context is the context to which the name `site/east/` is bound. This context can have the following bindings: `site/east/host`, `site/east/user`, and `site/east/service`.

`generic`
Create a generic context in which slash-separated left-to-right names can be bound, and bind the reference of the context to *composite_name*. The option `-r` can be used to specify the reference type to be associated with the context. If the `-r` option is

omitted, the reference type used is the reference type of the parent context if the parent context is a generic context; otherwise, the reference type is `onc_fn_generic`.

`-r` *reference_type* — Use *reference_type* as the reference type of the generic context being created. This option can be used only with the `-t generic` option.

`-s` — Create the context and bind it in to supercede any existing binding associated with *composite_name*. If this option is omitted, `fncreate` fails if *composite_name* is already bound.

`-v` — Information about the creation of a context is displayed as each context is created.

OPERANDS

The following operand is supported:

composite_name — An FNS named object.

EXAMPLES

EXAMPLE 1 Creating A Host Context

This example illustrates the creation of a host context in the root organization and a user context in a sub-organization.

The following command creates a context, and subcontexts, for the root organization:

```
% fncreate -t org org//
```

It causes the following commands to be invoked automatically:

```
% fncreate -t service org//service/
% fncreate -t hostname org//host/
% fncreate -t username org//user/
```

The following command creates a context, and subcontexts, for host `sylvan`:

```
% fncreate -t host org//host/sylvan/
```

It causes the following commands to be invoked automatically:

```
% fncreate -t service org//host/sylvan/service/
eg% fncreate -t fs org//host/sylvan/fs/
```

The following command creates a context, and subcontexts, associated with a sub-organization `dct`:

```
% fncreate -t org org/dct/
```

It causes the following commands to be invoked automatically:

```
% fncreate -t service org/dct/service/
% fncreate -t hostname org/dct/host/
% fncreate -t username org/dct/user/
```

EXAMPLE 1 Creating A Host Context *(Continued)*

The following command creates a context, and subcontexts, for user `msmith`:

```
% fncreate -t user org/dct/user/msmith/
```

It causes the following commands to be invoked automatically:

```
% fncreate -t service org/dct/user/msmith/service/
% fncreate -t fs org/dct/user/msmith/fs/
```

The following commands create service contexts:

```
% fncreate -t service org/dct/service/fax
% fncreate -t service org/dct/service/fax/classA
```

EXIT STATUS

`0` Operation was successful.

`1` Operation failed.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWfns

SEE ALSO

`nis`(1), `fncheck`(1M), `fncreate_fs`(1M), `fndestroy`(1M), `fnselect`(1M), `nissetup`(1M), `xfn`(3XFN), `attributes`(5), `fns`(5), `fns_files`(5), `fns_nis`(5), `fns_nis+`(5), `fns_policies`(5), `fns_references`(5)

NAME

fncreate_fs – create FNS file system contexts

SYNOPSIS

fncreate_fs [-r] [-v] -f *input_file composite_name*

fncreate_fs [-r] [-v] *composite_name* [*mount_options*] [*mount_location*...]

DESCRIPTION

The fncreate_fs command creates or updates the FNS file system context named by *composite_name*. A description of the context's bindings is provided in *input_file* if the first form of the command is used, or is given on the command line if the second form is used.

OPTIONS

-r — Replace the bindings in the context named by *composite_name* with only those specified in the input. This is equivalent to destroying the context (and, recursively, its subcontexts), and then running fncreate_fs without this option. This option should be used with care.

-v — Verbose. Display information about the contexts being created and modified.

-f *input_file* — Read input from *input_file*. If *input_file* is '-' (hyphen), read from standard input instead.

OPERANDS

The following operand is supported:

composite_name — An FNS named object.

USAGE

The fncreate_fs command populates the file system portions of the FNS namespace. The automounter (see automount(1M)) will then "mount" the FNS namespace under /xfn. The directory with the FNS name org/engineering/fs, for example, can be found on the file system as /xfn/org/engineering/fs.

The format of the input to fncreate_fs is similar, but not identical, to the format of indirect automount maps. Differences are enumerated in the NOTES section below.

Input File Format

The input file supplies the names and values to be bound in the context of *composite_name*. Its format is a sequence of lines of the form:

name [-*options*] [*location* . . .] For each such entry, a reference to the *location*(s) and the corresponding *options* is bound to the name *composite_name/name*. The *name* field may be a simple atomic name, a slash-separated hierarchical name, or '.' (period). If it is '.' then the reference is bound directly to *composite_name*. The *name* field must not begin with a slash.

The *location* field specifies the host or hosts that serve the files for *composite_name/name*. In the case of a simple NFS mount, *location* takes the form:

host : *path* where *host* is the name of the host from which to mount the file system, and *path* is the path name of the directory to mount.

The *options* field is a comma-separated list of the mount options to use when mounting the location bound to *composite_name/name*. These options also apply to any subcontexts of *composite_name/name* that do not specify their own mount options. If *options* is given but *location* is not, the options apply to subcontexts only.

If neither *options* nor a *location* is given, then no reference is bound to *composite_name/name*. Any existing reference is unbound.

A single logical line may be continued across multiple input lines by escaping the newline with a '\' (backslash). Comments begin with a '#' that is either at the beginning of a line or is prefixed by whitespace, and end at the end of the line.

Command-line Input

If no *input_file* is specified on the command line, then the *options* and *location* fields given on the command line are bound directly to *composite_name*. This is equivalent to providing a one-line input file with a '.' in the *name* field.

Multiple Locations

Multiple *location* fields may be specified for NFS file systems that are exported from multiple, functionally-equivalent locations. If several locations in the list share the same path name, they may be combined using a comma-separated list of host names:

host1, *host2*, . . . : *path*

The hosts may be weighted, with the weighting factor appended to the host name as a non-negative integer in parentheses: the lower the number, the more desirable the server. The default weighting factor is `0` (most desirable). In the example:

```
alpha,bravo,charlie(1),delta(2):/usr/man
```

hosts `alpha` and `bravo` are the most desirable; host `delta` is the least desirable.

See the `USAGE` section of `automount`(1M) for additional information on how the automounter interprets the location field.

Variable Substitution

Variable names, prefixed by '$', may be used with the *options* or *location* fields. For example, a *location* may be given as:

`svr1:/export/$CPU`The automounter will substitute client-specific values for these variables when mounting the corresponding file systems. In the above example, `$CPU` is replaced by the output of `uname -p`; for example, "`sparc`". See the `USAGE` section of `automount`(1M) for more information on how the automounter treats variable substitution.

Alternate Input Format

For additional compatibility with automount maps (see `automount`(1M)), the following input format is accepted:

```
name        [options] [location  . . .] \
    /offset1    [options1] location1  . . . \
    /offset2    [options2] location2  . . . \
     . . .
```

where each *offset* field is a slash-separated hierarchy. This is interpreted as being equivalent to:

```
name    [options] [location  . . .^]
name/offset1    [options1] location1  . . .
name/offset2    [options2] location2  . . .
 . . .(the first line being omitted if both options and location are omitted).
```

This format is for compatibility only; it provides no additional functionality. Its use is deprecated.

EXAMPLES

EXAMPLE 1 Using the `fncreate_fs` Command

The following examples illustrate the use of the `fncreate_fs` command. The call:

```
example%  cat input1
src     -ro       svr1:/export/src
dist    -ro       svr2,svr3:/export/dist
example%  fncreate_fs -f input1 org/engineering/fs
```

creates a file system context for the engineering organization. It specifies that `org/engineering/fs/src` is a read-only NFS mount from server `svr1`, and that `org/engineering/fs/dist` is a read-only NFS mount from either `svr2` or `svr3`.

Once this is done, there are several equivalent ways to create the engineering organization's `src/cmd` context. It could be done using the composite name `org/engineering/fs`:

```
example%  cat input2
src/cmd        svr1:/export/cmd
example%  fncreate_fs -f input2 org/engineering/fs
```

Equivalently, it could be done using the composite name `org/engineering/fs/src`:

```
example%  cat input3
cmd        svr1:/export/cmd
example%  fncreate_fs -f input3 org/engineering/fs/src
```

The same results could also be achieved by:

```
example%  fncreate_fs org/engineering/fs/src/cmd svr1:/export/cmd
```

Note that `cmd` will also be mounted read-only, since it is a subcontext of `src` and does not have mount options of its own.

In the first example of this section, the `-ro` mount option was specified for each entry in the input file. It could instead have been specified only once:

```
example%  cat input4
.       -ro
src        svr1:/export/src
dist        svr2,svr3:/export/dist
example%  fncreate_fs -f input4 org/engineering/fs
```

EXAMPLE 1 Using the `fncreate_fs` Command *(Continued)*

The `-ro` option here applies to all bindings in the context `org/engineering/fs` and any of its subcontexts. In particular, it also applies to the `cmd` context from the above examples.

The following will change the NFS server for the `src` context:

```
example%  fncreate_fs org/engineering/fs/src svr4:/export/src
```

Had the `-r` option been used, the `cmd` subcontext would have been destroyed as well:

```
example%  fncreate_fs -r org/engineering/fs/src svr4:/export/src
```

Only the FNS context is destroyed. The `/export/cmd` directory on `svr1` is not affected.

The file system contexts of users and hosts are not usually created by `fncreate_fs` (see the `NOTES` section below). The defaults set by `fncreate`, however, may be overridden. For example, the call:

```
example%  fncreate_fs user/jane/fs svr1:/export/home/jane
```

sets Jane's file system to be an NFS mount from `svr1`.

EXIT STATUS

`0` Operation was successful.

`1` Operation failed.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWfns

SEE ALSO

`fnbind`(1), `fnlist`(1), `fnlookup`(1), `fnunbind`(1), `nis+`(1), `automount`(1M), `fncreate`(1M), `fndestroy`(1M), `attributes`(5), `fns`(5), `fns_files`(5), `fns_nis`(5), `fns_nis+`(5), `fns_policies`(5)

NOTES

The `fncreate_fs` command affects the FNS file system namespace only. It does not have any effect on the servers that export the files and directories from which the namespace is constructed. Destroying an FNS context does not remove any files on any server.

FNS policies specify that file system contexts are bound after the namespace identifier `fs` in composite names (see `fns_policies`(5)). Therefore, *composite_name* must contain an `fs`. The alias `_fs` may be used in place of `fs`.

The context named by the components of *composite_name* preceding `fs` must exist prior to the call to `fncreate_fs`, since `fncreate_fs` creates *only* file system contexts.

Default file system contexts for hosts and users are generally created by the command `fncreate`(1M). These defaults may be overridden using `fncreate_fs`. Overriding a host's default file system context is unlikely to make sense.

The input file format is similar to the format of indirect automount maps (see `automount`(1M)). The differences are:

- the *name* field may be hierarchical, and may be '`.`'
- there are no included maps or special maps
- there may be entries with neither options nor locations
- the characters '`*`' and '`&`' have no special meaning

The process executing the `fncreate_fs` command may need certain credentials to update information in the underlying naming service. See `fns_nis`(5), `fns_nis+`(5), and `fns_files`(5) for more information.

NAME

fncreate_printer – create new printers in the FNS namespace

SYNOPSIS

fncreate_printer [-sv] *compositename printername printeraddr* [*printeraddr*...]

fncreate_printer [-sv] [-f *filename*] *compositename*

DESCRIPTION

fncreate_printer creates a new printer context for an *organization*, *user*, *host*, or *site* object. *compositename* is the FNS name of the object. fncreate_printer uses *printername* to name the new printer and binds it to an FNS reference constructed from the set of *printeraddr*s. fncreate_printer may also be used to add new *printeraddr*s for an existing *printername*.

The command also supports creating a set of printers as listed in the file *filename*.

The new printer context is created with the FNS name <*compositename*>/service/printer/<*printername*>. If the intermediate service or printer names do not exist, their FNS contexts are also created by this command. Normally, these intermediate contexts would be created by an administrative script that uses fncreate(1M), and is run at the time a new FNS organization is set up. The reference bound to the FNS printer name is of type onc_printers and is constructed from the set of *printeraddr*s. A *printeraddr* is of the form <*addresstype*> = <*address*>. See printers.conf(4) for the format of *printeraddr* and also the examples below for currently supported address types and address strings.

An FNS printer name is accepted as a valid printer name by lp(1), lpstat(1), cancel(1), lpmove(1M), lpr(1B), lpq(1B), and lprm(1B).

The *printername* argument may be a slash-separated name. In this case, prior to creating the printer context denoted by the ''leaf'' name, this command will create printer context(s) for the intermediate node(s) if they do not already exist. See EXAMPLES.

fncreate_printer creates entries in the naming service determined by fnselect(1M). See fnselect(1M) for more information on the default naming service and on selecting a naming service. Furthermore, the process executing the fncreate_printer command may require certain credentials to update information in the underlying namespace. See fns_nis+(5), fns_nis(5), and fns_files(5) for more information.

OPTIONS

-s	The new *address* supersedes an existing address with the same *addresstype*, if any, for <*compositename*>/service/printer/<*printername*>. If this option is omitted, it appends the *printeraddr* to an existing reference, or creates a new reference using *printeraddr* for the printer.
-v	Displays information about individual printer contexts as they are created.
-f *filename*	Use *filename* to obtain a list of printers for which to create contexts. If this option is omitted, /etc/printers.conf is used as the

input file, in which case the `-s` option should be used to supersede the entries already present in this file.

OPERANDS

filename	The file that contains a list of printers to be created. This file uses the same format as `/etc/printers.conf`. See `printers.conf`(4) for more information.
printername	The name of the new printer context created.
printeraddr	An address to be associated with the printer context name.
compositename	The FNS name for the `org`, `host`, `user`, or `site` object for which the new printer contexts are created.

EXAMPLES

EXAMPLE 1 Creating Printer Contexts

The following examples illustrate creating a set of printer contexts under an organization, a printer context for a user, and a printer context associated with a hierarchical printer name for a site, respectively. To create printers for an organization:

```
example% fncreate_printer -s org/marketing
```

This causes the creation of a printer context for every entry listed in the `/etc/printers.conf` file on the system where the command is executed. The printer contexts thus created are bound under the organization's printer context, `org/marketing/service/printer`. The `-s` flag is required to force the creation of the printer contexts in the underlying namespace, since the default `/etc/printers.conf` file is being used.

To create a printer named `ps` for user `jsmith` and associate it with the `killtree` printer served by the print server `paperwaster`:

```
example% fncreate_printer -s usr/jsmith ps bsdaddr=paperwaster,killtree
```

This causes `jsmith`'s `ps` printername to be associated with the `killtree` printer on the server `paperwaster`, overwriting any existing address of type `bsdaddr`. The user can print to this printer using the command:

```
example% lp -d thisuser/service/printer/ps <filename>
```

To create a printer with the hierarchical name `color/fast` under a site:

```
example% fncreate_printer site/bldg14/northwing color/fast \
        bsdaddr=paperwaster,laser
```

This causes the printer named `site/bldg14/northwing/service/printer/color/fast` to be associated with the `laser` printer on server `paperwaster`. If the intermediate printer context `site/bldg14/northwing/service/printer/color` does not exist, it will also be created and associated with the same printer. If the printer name `site/bldg14/northwing/service/printer/color/fast` already exists and has an address of type `bsdaddr` associated with it, this command will fail.

EXIT STATUS

`0`	Successful operation.

1 Operation failed.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWfns

SEE ALSO

`cancel`(1), `lp`(1), `lpq`(1B), `lpr`(1B), `lprm`(1B), `lpstat`(1), `fncreate`(1M), `fnselect`(1M), `lpmove`(1M), `printers`(4), `printers.conf`(4), `attributes`(5), `fns`(5), `fns_files`(5), `fns_nis`(5), `fns_nis+`(5)

NAME

fndestroy – destroy an FNS context

SYNOPSIS

`fndestroy` *composite_name*

DESCRIPTION

`fndestroy` removes the context bound to *composite_name*. The context is not removed if there are subcontexts associated with *composite_name*.

EXAMPLES

EXAMPLE 1 Using The `fndestroy` Command

The command

```
example% fndestroy user/jsmith/
```

destroys the context named by `user/jsmith/` and removes the binding of `jsmith` from the context `user/`.

This command fails if the context `user/jsmith/` contains subcontexts, or if the invoker does not have the NIS+ credentials required to delete the NIS+ tables that store the user's bindings. See `fns`(5).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWfns

SEE ALSO

`fnlist`(1), `fnlookup`(1), `fnunbind`(1), `fncreate`(1M), `attributes`(5), `fns`(5), `fns_policies`(5)

NAME

fnselect – select a specific naming service to use for the FNS Initial Context

SYNOPSIS

fnselect [-D]

fnselect *naming-service*

DESCRIPTION

`fnselect` is used to set the specified naming service to be used to construct the bindings in the FNS Initial Context. This setting affects the entire machine and affects applications that make subsequent calls to `fn_ctx_handle_from_initial`(3XFN). This setting can be changed only by an administrator who has root privilege on the machine.

OPTIONS

`-D` Displays the actual naming service used to generate the FNS Initial Context.

OPERANDS

naming-service The following are possible values for *naming-service*:

`default`	Use the FNS default algorithm for determining the target naming service.
`nisplus`	Use NIS+ as the target naming service.
`nis`	Use NIS as the target naming service.
`files`	Use `/etc` files as the target naming service.

USAGE

When the `default` option is selected, FNS determines the underlying naming service using the following algorithm:

- First, it checks for NIS+ with FNS installed.
- If the result is `TRUE`, then FNS assumes `nisplus` as the underlying naming service.
- Otherwise, it checks if the system is a NIS client.
- If `TRUE`, FNS assumes `nis` as the underlying naming service.
- Otherwise, FNS assumes `/etc files`.

`fnselect` without any arguments displays the service currently selected for the Initial Context (one of `default`, `nisplus`, `nis`, or `files`).

When the `-D` option is specified and the current setting is `default`, `fnselect` will use the algorithm that is used by FNS and display the actual naming service used for the FNS Initial Context.

EXAMPLES

EXAMPLE 1 Using The `fnselect` Command

The command

```
example% fnselect nisplus
```

will select NIS+ as the underlying naming service for the FNS Initial Context.

The command

EXAMPLE 1 Using The `fnselect` Command *(Continued)*

```
example% fnselect
```

will print the naming service currently being used to generate the FNS Initial Context.

EXIT STATUS

`0` Operation was successful.

`1` Operation failed.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWfns

SEE ALSO

`fnbind`(1), `fnlist`(1), `fnlookup`(1), `fnunbind`(1), `fncreate`(1M), `fncreate_fs`(1M), `fncreate_printer`(1M), `fndestroy`(1M), `fn_ctx_handle_from_initial`(3XFN), `attributes`(5), `fns`(5), `fns_initial_context`(5)

NAME

fnsypd – update FNS context on an NIS master server

SYNOPSIS

/usr/sbin/fnsypd

DESCRIPTION

The `fnsypd` daemon is a Remote Procedure Call (RPC) service that accepts requests from NIS clients to update and modify Federated Naming Service (FNS) contexts. This daemon runs on an NIS master server with FNS on top of it. The `fnsypd` daemon requires the Secure Key Infrastructure (SKI) mechanism for authentication. The SKI mechanism is part of the SUNWski package. If SUNWski is not installed, authentication cannot be performed and users will receive "permission denied" error messages. The SUNWski man pages are located at `/opt/SUNWski/man`.

`fnsypd` enables users and hosts to modify only their respective FNS contexts. Organization, site, hostname and username contexts cannot be modified using `fnsypd`.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWfns

SEE ALSO

`nis`(1), `attributes`(5), `fns`(5), `fns_policies`(5)

NAME

format – disk partitioning and maintenance utility

SYNOPSIS

```
format [-f command-file] [-l log-file] [-x data-file] [-d disk-name]
     [-t disk-type] [-p partition-name] [-s] [-m] [-M] [-e] [disk-list]
```

DESCRIPTION

`format` enables you to format, label, repair and analyze disks on your system. Unlike previous disk maintenance programs, `format` runs under SunOS. Because there are limitations to what can be done to the system disk while the system is running, `format` is also supported within the memory-resident system environment. For most applications, however, running `format` under SunOS is the more convenient approach.

`format` first uses the disk list defined in *data-file* if the `-x` option is used. `format` then checks for the `FORMAT_PATH` environment variable, a colon-separated list of filenames and/or directories. In the case of a directory, `format` searches for a file named `format.dat` in that directory; a filename should be an absolute pathname, and is used without change. `format` adds all disk and partition definitions in each specified file to the working set. Multiple identical definitions are silently ignored. If `FORMAT_PATH` is not set, the path defaults to `/etc/format.dat`.

disk-list is a list of disks in the form `c?t?d?` or `/dev/rdsk/c?t?d?s?`. With the latter form, shell wildcard specifications are supported. For example, specifying `/dev/rdsk/c2*` causes `format` to work on all drives connected to controller `c2` only. If no *disk-list* is specified, `format` lists all the disks present in the system that can be administered by `format`.

Removable media devices are listed only when users execute `format` in expert mode (option `-e`). This feature is provided for backward compatibility. Use `rmformat`(1) for rewritable removable media devices.

OPTIONS

The following options are supported:

`-d` *disk-name*
: Specify which disk should be made current upon entry into the program. The disk is specified by its logical name (for instance, `-d c0t1d0`). This can also be accomplished by specifying a single disk in the disk list.

`-e`
: Enable SCSI expert menu. Note this option is not recommended for casual use.

`-f` *command-file*
: Take command input from *command-file* rather than the standard input. The file must contain commands that appear just as they would if they had been entered from the keyboard. With this option, `format` does not issue `continue?` prompts; there is no need to specify y(es) or n(o) answers in the *command-file*. In non-interactive mode, `format` does not initially expect the input of a disk selection number. The user must specify the current working disk with the `-d` *disk-name*

	option when format is invoked, or specify `disk` and the disk selection number in the *command-file*.
`-l` *log-file*	Log a transcript of the `format` session to the indicated *log-file*, including the standard input, the standard output and the standard error.
`-m`	Enable extended messages. Provides more detailed information in the event of an error.
`-M`	Enable extended and diagnostic messages. Provides extensive information on the state of a SCSI device's mode pages, during formatting.
`-p` *partition-name*	Specify the partition table for the disk which is current upon entry into the program. The table is specified by its name as defined in the data file. This option can be used only if a disk is being made current, and its type is either specified or available from the disk label.
`-s`	Silent. Suppress all of the standard output. Error messages are still displayed. This is generally used in conjunction with the `-f` option.
`-t` *disk-type*	Specify the type of disk which is current upon entry into the program. A disk's type is specified by name in the data file. This option can only be used if a disk is being made current as described above.
`-x` *data-file*	Use the list of disks contained in *data-file*.

USAGE

When you invoke format with no options or with the `-e`, `-l`, `-m`, `-M`, or `-s` options, the program displays a numbered list of available disks and prompts you to specify a disk by list number. If the machine has more than 10 disks, press Space to see the next screenful of disks.

You can specify a disk by list number even if the disk is not displayed in the current screenful. For example, if the current screen shows disks 11-20, you can enter `25` to specify the twenty-fifth disk on the list. If you enter a number for a disk that is not currently displayed, `format` prompts you to verify your selection. If you enter a number from the displayed list, `format` silently accepts your selection.

After you specify a disk, `format` displays its main menu. This menu enables you to perform the following tasks:

analyze	Run read, write, and compare tests.
backup	Search for backup labels.
cache	Enable, disable, and query the state of the write cache and read cache. This menu item only appears when `format` is invoked with the `-e` option, and is only supported on SCSI devices..

current	Display the device name, the disk geometry, and the pathname to the disk device.
defect	Retrieve and print defect lists. This option is supported only on SCSI devices. IDE disks perform automatic defect management. Upon using the `defect` option on an IDE disk, you receive the message: `Controller does not support defect management or disk supports automatic defect management.`
disk	Choose the disk that will be used in subsequent operations (known as the current disk.)
fdisk	Run the `fdisk`(1M) program to create a `fdisk` partition for Solaris software (IA based systems only).
format	Format and verify the current disk. This option is supported only on SCSI devices. IDE disks are pre-formatted by the manufacturer. Upon using the `format` option on an IDE disk, you receive the message: `Cannot format this drive. Please use your manufacturer-supplied formatting utility.`
inquiry	Display the vendor, product name, and revision level of the current drive.
label	Write a new label to the current disk.
partition	Create and modify slices.
quit	Exit the format menu.
repair	Repair a specific block on the disk.
save	Save new disk and slice information.
type	Select (define) a disk type.
verify	Read and display labels. Print information such as the number of cylinders, alternate cylinders, heads, sectors, and the partition table.
volname	Label the disk with a new eight character volume name.

ENVIRONMENT VARIABLES

`FORMAT_PATH`	a colon-separated list of filenames and/or directories of disk and partition definitions. If a directory is specified, `format` searches for the file `format.dat` in that directory.

FILES

`/etc/format.dat`	default data file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`fmthard`(1M), `prtvtoc`(1M), `rmformat`(1), `format.dat`(4), `attributes`(5), `sd`(7D)

See *Disk Management* in *System Administration Guide: Basic Administration*

IA Only

`fdisk`(1M)

WARNINGS

When the `format` function is selected to format the Maxtor 207MB disk, the following message displays:

```
Mode sense page(4) reports rpm value as 0, adjusting it to 3600
```

This is a drive bug that may also occur with older third party drives. The above message is not an error; the drive will still function correctly.

Cylinder 0 contains the partition table (disk label), which can be overwritten if used in a raw disk partition by third party software.

NOTES

`format` provides a help facility you can use whenever format is expecting input. You can request help about what information is expected by simply entering a question mark (`?`) and `format` prints a brief description of what type of input is needed. If you enter a `?` at the menu prompt, a list of available commands is displayed.

For SCSI disks, formatting is done with both Primary and Grown defects list by default. However, if only Primary list is extracted in defect menu before formatting, formatting will be done with Primary list only.

Changing the state of the caches is only supported on SCSI devices, and not all SCSI devices support changing or saving the state of the caches.

NAME

fsck – check and repair file systems

SYNOPSIS

fsck [-F *FSType*] [-m] [-V] [*special*...]

fsck [-F *FSType*] [-n | N | y | Y] [-V] [-o *FSType-specific-options*] [*special*...]

DESCRIPTION

`fsck` audits and interactively repairs inconsistent file system conditions. If the file system is inconsistent the default action for each correction is to wait for the user to respond `yes` or `no`. If the user does not have write permission `fsck` defaults to a `no` action. Some corrective actions will result in loss of data. The amount and severity of data loss can be determined from the diagnostic output.

FSType-specific-options are options specified in a comma-separated (with no intervening spaces) list of options or keyword-attribute pairs for interpretation by the *FSType*-specific module of the command.

special represents the character special device on which the file system resides, for example, `/dev/rdsk/c1t0d0s7`. Note: the character special device, not the block special device, should be used. `fsck` will not work on a block device if it is mounted.

If no *special* device is specified `fsck` checks the file systems listed in `/etc/vfstab`. Those entries in `/etc/vfstab` which have a character special device entry in the `fsckdev` field and have a non-zero numeric entry in the `fsckpass` field will be checked. Specifying `-F` *FSType* limits the file systems to be checked to those of the type indicated.

If *special* is specified, but `-F` is not, the file system type will be determined by looking for a matching entry in `/etc/vfstab`. If no entry is found, the default local file system type specified in `/etc/default/fs` will be used.

If a file system type supports parallel checking, for example, `ufs`, some file systems eligible for checking may be checked in parallel. Consult the file system-specific man page (for example, `fsck_ufs`(1M)) for more information.

OPTIONS

The following generic options are supported:

`-F` *FSType* — Specify the file system type on which to operate.

`-m` — Check but do not repair. This option checks that the file system is suitable for mounting, returning the appropriate exit status. If the file system is ready for mounting, `fsck` displays a message such as:

```
ufs fsck: sanity check: /dev/rdsk/c0t3d0s1 okay
```

`-n | -N` — Assume a `no` response to all questions asked by `fsck`; do not open the file system for writing.

`-V` Echo the expanded command line but do not execute the command. This option may be used to verify and to validate the command line.

`-y | Y` Assume a `yes` response to all questions asked by `fsck`.

`-o` *specific-options* These *specific-options* can be any combination of the following separated by commas (with no intervening spaces).

`b=`*n*
Use block *n* as the super block for the file system. Block 32 is always one of the alternate super blocks. Determine the location of other super blocks by running `newfs`(1M) with the `-Nv` options specified.

`c`
If the file system is in the old (static table) format, convert it to the new (dynamic table) format. If the file system is in the new format, convert it to the old format provided the old format can support the file system configuration. In interactive mode, `fsck` will list the direction the conversion is to be made and ask whether the conversion should be done. If a negative answer is given, no further operations are done on the file system. In preen mode, the direction of the conversion is listed and done if possible without user interaction. Conversion in preen mode is best used when all the file systems are being converted at once. The format of a file system can be determined from the first line of output from `fstyp`(1M). Note: the `c` option is seldom used and is included only for compatibility with pre-4.1 releases. There is no guarantee that this option will be included in future releases.

`f`
Force checking of file systems regardless of the state of their super block clean flag.

`p`
Check and fix the file system non-interactively (“preen”). Exit immediately if there is a problem requiring intervention. This option is required to enable parallel file system checking.

`w`
Check writable file systems only.

EXIT STATUS

`0` file system is okay and does not need checking

`1` erroneous parameters are specified

32	file system is unmounted and needs checking (`fsck -m` only)
33	file system is already mounted
34	cannot stat device
36	uncorrectable errors detected - terminate normally
37	a signal was caught during processing
39	uncorrectable errors detected - terminate immediately
40	for root, same as `0`.

USAGE

See `largefile`(5) for the description of the behavior of `fsck` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

FILES

`/etc/default/fs` default local file system type. Default values can be set for the following flags in `/etc/default/fs`. For example: `LOCAL=ufs`.

`LOCAL` The default partition for a command if no FSType is specified.

`/etc/vfstab` list of default parameters for each file system

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`clri`(1M), `fsck_cachefs`(1M), `fsck_ufs`(1M), `fsdb_ufs`(1M), `fsirand`(1M), `fstyp`(1M), `mkfs`(1M), `mkfs_ufs`(1M), `mountall`(1M), `newfs`(1M), `reboot`(1M), `fs_ufs`(4), `vfstab`(4), `attributes`(5), `largefile`(5)

WARNINGS

The operating system buffers file system data. Running `fsck` on a mounted file system can cause the operating system's buffers to become out of date with respect to the disk. For this reason, the file system should be *unmounted* when `fsck` is used. If this is not possible, care should be taken that the system is quiescent and that it is rebooted immediately after `fsck` is run. Quite often, however, this will not be sufficient. A panic will probably occur if running `fsck` on a file system modifies the file system.

NOTES

This command may not be supported for all *FSTypes*.

Running `fsck` on file systems larger than 2 Gb fails if the user chooses to use the block interface to the device:

```
fsck /dev/dsk/c?t?d?s?
```

rather than the raw (character special) device:

```
fsck /dev/rdsk/c?t?d?s?
```

Starting with Solaris 9, `fsck` manages extended attribute data on the disk. (See `fsattr`(5) for a description of extended file attributes.) A file system with extended attributes can be mounted on versions of Solaris that are not attribute-aware (versions prior to Solaris 9), but the attributes will not be accessible and `fsck` will strip them from the files and place them in `lost+found`. Once the attributes have been stripped, the file system is completely stable on versions of Solaris that are attribute-aware, but would be considered corrupted on attribute-aware versions. In the latter circumstance, run the attribute-aware `fsck` to stabilize the file system before using it in an attribute-aware environment.

NAME

fsck_cachefs – check integrity of data cached with CacheFS

SYNOPSIS

fsck -F cachefs [-m | -o | noclean] *cache_directory*

DESCRIPTION

The CacheFS version of the `fsck` command checks the integrity of a cache directory. By default it corrects any CacheFS problems it finds. There is no interactive mode. The most likely invocation of `fsck` for CacheFS file systems is at boot time from an entry in the `/etc/vfstab` file (see `vfstab`(4)).

OPTIONS

Two command line options are available:

`-m`	Check, but do not repair.
`-o noclean`	Force a check on the cache even if there is no reason to suspect there is a problem.

EXAMPLES

EXAMPLE 1 An example of the `fsck` command.

The following example forces a check on the cache directory `/cache3`:

```
example% fsck -F cachefs -o noclean /cache3
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`cfsadmin`(1M), `fsck`(1M), `mount_cachefs`(1M), `vfstab`(4), `attributes`(5)

NAME

fsck_pcfs – file system consistency check and interactive repair

SYNOPSIS

fsck -F pcfs [*generic_options*] *special*

fsck -F pcfs [*generic_options*] [-o *specific_options*] *special*

DESCRIPTION

The `fsck` utility audits and interactively repairs inconsistent conditions on file systems. *special* represents the character special device on which the file system resides, for example `/dev/rdiskette`. The character special device, not the block special device, should be used.

In the case of correcting serious inconsistencies, by default, `fsck` asks for confirmation before making a repair and waits for the operator to respond either `yes` or `no`. If the operator does not have write permission on the file system, `fsck` defaults to a `-n` (no corrections) action. See `fsck`(1M).

Repairing some file system inconsistencies may result in loss of data. The amount and severity of data loss may be determined from the diagnostic output.

When executed with the verify option (`-o v`), `fsck_pcfs` automatically scans the entire file system to verify that all of its allocation units are accessible. If it finds any units inaccessible, it updates the file allocation table (FAT) appropriately. It also updates any effected directory entries to reflect the problem. This directory update includes truncating the file at the point in its allocation chain where the file data is no longer accessible. Any remaining accessible allocation units become orphaned.

Orphaned chains of accessible allocation units are, with the operator's concurrence, linked back into the file system as files in the root directory. These files are assigned names of the form file*NNNN*.chk, where the *N*s are digits in the integral range from `0` through `9`.

After successfully scanning and correcting any errors in the file system, `fsck` displays a summary of information about the file system. This summary includes the size of the file system in bytes, the number of bytes used in directories and individual files, and the number of available allocation units remaining in the file system.

OPTIONS

generic_options

The following generic options are supported:

`-m`

Check but do not repair. This option checks that the file system is suitable for mounting, returning the appropriate exit status. If the file system is ready for mounting, `fsck` displays a message such as:

```
pcfs fsck: sanity check:
/dev/rdiskette okay
```

`-n | -N`

Assume a no response to all questions asked by `fsck`; do not open the file system for writing.

`-V` Echo the expanded command line, but do not execute the command. This option may be used to verify and to validate the command line.

`-y | -Y` Assume a `yes` response to all questions asked by `fsck`.

`-o` *specific_options* Specify `pcfs` file system specific options in a comma-separated list, in any combination, with no intervening spaces.

- v — Verify all allocation units are accessible prior to correcting inconsistencies in the metadata.
- p — Check and fix the file system non-interactively (`preen`). Exit immediately if there is a problem requiring intervention.
- w — Check writable file systems only.

FILES

special The device which contains the pcfs. The device name for a diskette is specified as `/dev/rdiskette0` for the first diskette drive, or `/dev/rdiskette1` for a second diskette drive. A hard disk device or high-capacity removable device name much be qualified with a suffix to indicate the proper `FDISK` partition.

For example, in the names: `/dev/rdsk/c0t0d0p0:c` and `/dev/rdsk/c0t4d0s2:c`, the `:c` suffix indicates the first partition on the disk contains the `pcfs`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWesu
Interface Stability	Stable

SEE ALSO

`fsck`(1M), `fstyp`(1M), `fdisk`(1M), `mkfs`(1M), `mkfs_pcfs`(1M), `mountall`(1M), `attributes`(5), `pcfs`(7FS),

WARNINGS

The operating system buffers file system data. Running `fsck` on a mounted file system can cause the operating system's buffers to become out of date with respect to the disk. For this reason, the file system should be unmounted when `fsck` is used. If this is not possible, care should be taken that the system is quiescent and that it is rebooted immediately after `fsck` is run. Quite often, however, this is not sufficient. A panic will probably occur if running `fsck` on a file system modifies the file system.

NAME

fsck_udfs – file system consistency check and interactive repair

SYNOPSIS

fsck -F udfs [*generic_options*] [*special* ...]

fsck -F udfs [*generic_options*] [-o *specific_options*] [*special* ...]

DESCRIPTION

`fsck` audits and interactively repairs inconsistent conditions on file systems. A file system to be checked can be specified by giving the name of the block or character special device or by giving the name of its mount point if a matching entry exists in `/etc/vfstab`.

special represents the character special device, for example, `/dev/rdsk/c0t2d0s0`, on which the file system resides. The character special device, not the block special device should be used. `fsck` does not work on a mounted block device.

If no special device is specified, all `udfs` file systems specified in the `vfstab` file with a `fsckdev` entry are checked. If the `-p` (`preen`) option is specified, `udfs` file systems with an `fsckpass` number greater than `1` are checked in parallel. See `fsck`(1M).

In the case of correcting serious inconsistencies, by default, `fsck` asks for confirmation before making a repair and waits for the operator to respond with either `yes` or `no`. If the operator does not have write permission on the file system, `fsck` defaults to the `-n` (no corrections) option. See `fsck`(1M).

Repairing some file system inconsistencies can result in loss of data. The amount and severity of data loss can be determined from the diagnostic output.

`fsck` automatically corrects innocuous inconsistencies. It displays a message for each corrected inconsistency that identifies the nature of the correction which took place on the file system. After successfully correcting a file system, `fsck` prints the number of files on that file system and the number of used and free blocks.

Inconsistencies checked are as follows:

- Blocks claimed by more than one file or the free list
- Blocks claimed by a file or the free list outside the range of the file system
- Incorrect link counts in file entries
- Incorrect directory sizes
- Bad file entry format
- Blocks not accounted for anywhere
- Directory checks, file pointing to unallocated file entry and absence of a parent directory entry
- Descriptor checks, more blocks for files than there are in the file system
- Bad free block list format
- Total free block count incorrect

OPTIONS

The following options are supported:

generic_options
The following *generic_options* are supported:

`-m`
Check but do not repair. This option checks to be sure that the file system is suitable for mounting, and returns the appropriate exit status. If the file system is ready for mounting, `fsck` displays a message such as:

```
udfs fsck: sanity check: /dev/rdsk/c0t2d0s0 okay
```

`-n | -N`
Assume a `no` response to all questions asked by `fsck`; do not open the file system for writing.

`-V`
Echo the expanded command line, but do not execute the command. This option can be used to verify and to validate the command line.

`-y | -Y`
Assume a `yes` response to all questions asked by `fsck`.

`-o` *specific_options*
Specify `udfs` file system specific options in a comma-separated list with no intervening spaces. The following *specific_options* are available:

`f`
Force checking of file systems regardless of the state of their logical volume integrity state.

`p`
Check and fix the file system non-interactively (`preen`). Exit immediately if there is a problem that requires intervention. This option is required to enable parallel file system checking.

`w`
Check writable file systems only.

FILES

`/etc/vtstab` List of default parameters for each file system.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWudf

SEE ALSO

`fsck`(1M), `fsdb_udfs`(1M), `fstyp`(1M), `mkfs`(1M), `mkfs_udfs`(1M), `mountall`(1M), `reboot`(1M), `vfstab`(4), `attributes`(5)

WARNINGS

The operating system buffers file system data. Running `fsck` on a mounted file system can cause the operating system's buffers to become out of date with respect to the disk. For this reason, use `fsck` only when the file system is unmounted. If this is

not possible, take care that the system is quiescent and that it is rebooted immediately after running `fsck`. A panic will probably occur if running `fsck` on a file system that modifies the file system while it is mounted.

If an unmount of the file system is not done before the system is shut down, the file system might become corrupted. In this case, a file system check needs to be completed before the next mount operation.

DIAGNOSTICS

not writable
: You cannot write to the device.

Currently Mounted on
: The device is already mounted and cannot run `fsck`.

FILE SYSTEM WAS MODIFIED
: File system has been modified to bring it to a consistent state.

Can't read allocation extent
: Cannot read the block containing allocation extent.

Bad tag on alloc extent
: Invalid tag detected when expecting an allocation extent.

Volume sequence tag error
: Invalid tag detected in the volume sequence.

Space bitmap tag error
: Invalid tag detected in the space bitmap.

UNEXPECTED INCONSISTENCY; RUN fsck MANUALLY
: Use `fsck` in interactive mode.

NAME

fsck_ufs – file system consistency check and interactive repair

SYNOPSIS

fsck -F ufs [*generic-options*] [*special*...]

fsck -F ufs [*generic-options*] [-o *specific-options*] [*special*...]

DESCRIPTION

The `fsck` utility audits and interactively repairs inconsistent conditions on file systems. A file system to be checked may be specified by giving the name of the block or character *special* device or by giving the name of its mount point if a matching entry exists in `/etc/vfstab`.

The *special* parameter represents the character special device, for example, `/dev/rdsk/c1t0d0s7`, on which the file system resides. The character special device, not the block special device should be used. The `fsck` utility will not work on a block device if the block device is mounted, unless the file system is error-locked.

If no *special* device is specified, all `ufs` file systems specified in the `vfstab` with a `fsckdev` entry will be checked. If the `-p` (''preen'') option is specified, `ufs` file systems with an `fsckpass` number greater than `1` are checked in parallel. See `fsck`(1M).

In the case of correcting serious inconsistencies, by default, `fsck` asks for confirmation before making a repair and waits for the operator to respond either `yes` or `no`. If the operator does not have write permission on the file system, `fsck` will default to a `-n` (no corrections) action. See `fsck`(1M).

Repairing some file system inconsistencies can result in loss of data. The amount and severity of data loss can be determined from the diagnostic output.

The `fsck` utility automatically corrects innocuous inconsistencies such as unreferenced inodes, too-large link counts in inodes, missing blocks in the free list, blocks appearing in the free list and also in files, or incorrect counts in the super block. It displays a message for each inconsistency corrected that identifies the nature of the correction on the file system which took place. After successfully correcting a file system, `fsck` prints the number of files on that file system, the number of used and free blocks, and the percentage of fragmentation.

Inconsistencies checked are as follows:

- Blocks claimed by more than one inode or the free list.
- Blocks claimed by an inode or the free list outside the range of the file system.
- Incorrect link counts.
- Incorrect directory sizes.
- Bad inode format.
- Blocks not accounted for anywhere.
- Directory checks, file pointing to unallocated inode, inode number out of range, and absence of '`.`' and '`. .`' as the first two entries in each directory.
- Super Block checks: more blocks for inodes than there are in the file system.

- Bad free block list format.
- Total free block and/or free inode count incorrect.

Orphaned files and directories (allocated but unreferenced) are, with the operator's concurrence, reconnected by placing them in the `lost+found` directory. The name assigned is the inode number. If the `lost+found` directory does not exist, it is created. If there is insufficient space in the `lost+found` directory, its size is increased.

An attempt to mount a `ufs` file system with the `-o nolargefiles` option will fail if the file system has ever contained a large file (a file whose size is greater than or equal to 2 Gbyte). Invoking `fsck` resets the file system state if no large files are present in the file system. A successful mount of the file system after invoking `fsck` indicates the absence of large files in the file system. An unsuccessful mount attempt indicates the presence of at least one large file. See `mount_ufs`(1M).

OPTIONS

The *generic-options* consist of the following options:

`-m` Check but do not repair. This option checks that the file system is suitable for mounting, returning the appropriate exit status. If the file system is ready for mounting, `fsck` displays a message such as:

```
ufs fsck: sanity check:
/dev/rdsk/c0t3d0s1 okay
```

`-n | N` Assume a `no` response to all questions asked by `fsck`; do not open the file system for writing.

`-V` Echo the expanded command line, but do not execute the command. This option may be used to verify and to validate the command line.

`-y | Y` Assume a `yes` response to all questions asked by `fsck`.

See generic `fsck`(1M) for the details for specifying *special*.

`-o` *specific-options* Specify `ufs` file system specific options. These options can be any combination of the following separated by commas (with no intervening spaces).

`b=`*n*
Use block *n* as the super block for the file system. Block 32 is always one of the alternate super blocks. Determine the location of other super blocks by running `newfs`(1M) with the `-Nv` options specified.

`c`
If the file system is in the old (static table) format, convert it to the new (dynamic table) format. If the file system is in the new format, convert it to the old format provided the old format can support the file system configuration. In interactive mode, `fsck` will

list the direction the conversion is to be made and ask whether the conversion should be done. If a negative answer is given, no further operations are done on the file system. In preen mode, the direction of the conversion is listed and done if possible without user interaction. Conversion in preen mode is best used when all the file systems are being converted at once. The format of a file system can be determined from the first line of output from `fstyp`(1M). Note: the `c` option is seldom used and is included only for compatibility with pre-4.1 releases. There is no guarantee that this option will be included in future releases.

`f`
Force checking of file systems regardless of the state of their super block clean flag.

`p`
Check and fix the file system non-interactively ("preen"). Exit immediately if there is a problem requiring intervention. This option is required to enable parallel file system checking.

`w`
Check writable file systems only.

FILES

`/etc/vfstab` list of default parameters for each file system

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`clri`(1M), `fsck`(1M), `fsdb_ufs`(1M), `fsirand`(1M), `fstyp`(1M), `mkfs`(1M), `mkfs_ufs`(1M), `mount_ufs`(1M), `mountall`(1M), `newfs`(1M), `reboot`(1M), `fs_ufs`(4), `vfstab`(4), `attributes`(5), `largefile`(5)

WARNINGS

The operating system buffers file system data. Running `fsck` on a mounted file system can cause the operating system's buffers to become out of date with respect to the disk. For this reason, the file system should be *unmounted* when `fsck` is used. If this is not possible, care should be taken that the system is quiescent and that it is rebooted immediately after `fsck` is run. Quite often, however, this will not be sufficient. A panic will probably occur if running `fsck` on a file system modifies the file system.

NOTES

It is usually faster to check the character special device than the block special device.

Running `fsck` on file systems larger than 2 Gb fails if the user chooses to use the block interface to the device:

```
fsck /dev/dsk/c?t?d?s?
```

rather than the raw (character special) device:

```
fsck /dev/rdsk/c?t?d?s?
```

NAME

fsdb – file system debugger

SYNOPSIS

fsdb [-F *FSType*] [-V] [-o *FSType-specific_options*] *special*

DESCRIPTION

fsdb is a file system debugger that allows for the manual repair of a file system after a crash. *special* is a special device used to indicate the file system to be debugged. fsdb is intended for experienced users only. *FSType* is the file system type to be debugged. Since different *FSTypes* have different structures and hence different debugging capabilities, the manual pages for the *FSType*-specific fsdb should be consulted for a more detailed description of the debugging capabilities.

OPTIONS

-F Specify the *FSType* on which to operate. The *FSType* should either be specified here or be determinable from /etc/vfstab by matching the *special* with an entry in the table, or by consulting /etc/default/fs.

-V Echo the complete command line, but do not execute the command. The command line is generated by using the options and arguments provided by the user and adding to them information derived from /etc/vfstab. This option may be used to verify and validate the command line.

-o Specify *FSType*-specific options.

USAGE

See largefile(5) for the description of the behavior of fsdb when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

FILES

/etc/default/fs default local file system type. Default values can be set for the following flags in /etc/default/fs. For example: LOCAL=ufs

LOCAL: The default partition for a command if no FSType is specified.

/etc/vfstab list of default parameters for each file system

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

vfstab(4), attributes(5), largefile(5) Manual pages for the *FSType*-specific modules of fsdb.

NOTES

This command may not be supported for all *FSTypes*.

NAME

fsdb_udfs – udfs file system debugger

SYNOPSIS

fsdb [-F] udfs [*generic_option*] [-o *specific_option*] *special*

DESCRIPTION

The fsdb_udfs command is an interactive tool that can be used to patch up a damaged udfs file system. fsdb_udfs has conversions to translate block and i-numbers into their corresponding disk addresses. Mnemonic offsets to access different parts of an inode are also included. Mnemonic offsets greatly simplify the process of correcting control block entries or descending the file system tree.

fsdb contains several error-checking routines to verify inode and block addresses. These can be disabled if necessary by invoking fsdb with the -o option or by using the o command.

fsdb reads one block at a time, and therefore works with raw as well as block I/O devices. A buffer management routine is used to retain commonly used blocks of data in order to reduce the number of read system calls. All assignment operations result in an immediate write-through of the corresponding block. In order to modify any portion of the disk, fsdb must be invoked with the -w option.

Wherever possible, adb-like syntax has been adopted to promote the use of fsdb through familiarity.

OPTIONS

The following options are supported:

-o *specific_option*
: Specify udfs file system specific options in a comma-separated list with no intervening spaces. The following specific options are supported:

  o
  : Override some error conditions.

  p=*string*
  : Set prompt to *string*.

  w
  : Open for write.

  ?
  : Display usage.

USAGE

Numbers are considered hexadecimal by default. The user has control over how data is to be displayed or accepted. The base command displays or sets the input and output base. Once set, all input defaults to this base and all output displays in this base. The base can be overriden temporarily for input by preceding hexadecimal numbers by 0x, preceding decimal numbers with a 0t, or octal numbers with a 0. Hexadecimal numbers beginning with a-f or A-F must be preceded with a 0x to distinguish them from commands.

Disk addressing by fsdb is at the byte level. However, fsdb offers many commands to convert a desired inode, directory entry, block, and so forth, to a byte address. After the address has been calculated, fsdb records the result in the current address (dot).

Several global values are maintained by `fsdb`:

- Current base (referred to as `base`)
- Current address (referred to as `dot`)
- Current inode (referred to as `inode`)
- Current count (referred to as `count`)
- Current type (referred to as `type`)

Most commands use the preset value of `dot` in their execution. For example,

```
> 2:inode
```

first sets the value of dot (`.`) to `2`, colon (`:`), signifies the start of a command, and the `inode` command sets `inode` to `2`. A count is specified after a comma (`,`). Once set, count remains at this value until a new command is encountered that resets the value back to `1` (the default).

So, if

```
> 2000,400/X
```

is entered, `400` hex longs are listed from `2000`, and when completed, the value of dot is `2000 + 400 * sizeof` (long). If a RETURN is then entered, the output routine uses the current values of `dot`, `count`, and `type` and displays `400` more hex longs. An asterisk (`*`) causes the entire block to be displayed. An example showing several commands and the use of RETURN would be:

```
> 2:ino; 0:dir?d
```

or

```
> 2:ino; 0:db:block?d
```

The two examples are synonymous for getting to the first directory entry of the root of the file system. Once there, subsequently entering a RETURN, plus (`+`), or minus (`-`) advances to subsequent entries. Notice that

```
> 2:inode; :ls
```

or

```
> :ls /
```

is again synonymous.

Expressions

The following symbols are recognized by `fsdb`:

RETURN	Update the value of dot by the current value of *type* and *display* using the current value of *count*.
#	Update the value of dot by specifying a numeric expression. Specify numeric expressions using addition, subtraction, mulitiplication, and division operators (`+`, `-`, `*`, and `%`). Numeric expressions are evaluated from left to right and can use parentheses. After evaluation, the value of dot is updated.
, *count*	Update the count indicator. The global value of *count* is updated to *count*. The value of *count* remains until a

	new command is run. A *count* specifier of `*` attempts to show a blocks's worth of information. The default for *count* is `1`.
? *f*	Display in structured style with format specifier `f`. See `Formatted Output`.
/ *f*	Display in unstructured style with format specifier `f`. See `Formatted Output`.
.	Display the value of dot.
+*e*	Increment the value of dot by the expression *e*. The amount actually incremented is dependent on the size of type: `dot = dot +` *e* `* sizeof` (*type*) The default for *e* is `1`.
–*e*	Decrement the value of dot by the expression *e* . See `+`.
**e*	Multiply the value of dot by the expression *e*. Multiplication and division don't use *type*. In the above calculation of dot, consider the `sizeof` (*type*) to be `1`.
%*e*	Divide the value of dot by the expression *e*. See `*`.
< *name*	Restore an address saved in register *name*. *name* must be a single letter or digit.
> *name*	Save an address in register *name*. *name* must be a single letter or digit.
= *f*	Display indicator. If *f* is a legitimate format specifier (see `Formatted Output`), then the value of dot is displayed using format specifier *f*. Otherwise, assignment is assumed. See `= [s] [e]`.
= [*s*] [*e*]	Change the value of dot using an assignment indicator. The address pointed to by dot has its contents changed to the value of the expression *e* or to the ASCII representation of the quoted (`"`) string *s*. This can be useful for changing directory names or ASCII file information.
=+ *e*	Change the value of dot using an incremental assignment. The address pointed to by dot has its contents incremented by expression *e*.
=- e	Change the value of dot using a decremental assignment. Decrement the contents of the address pointed to by dot by expression *e*.

Commands

A command must be prefixed by a colon (`:`). Only enough letters of the command to uniquely distinguish it are needed. Multiple commands can be entered on one line by separating them by a SPACE, TAB, or semicolon (`;`).

To view a potentially unmounted disk in a reasonable manner, `fsdb` supports the `cd`, `pwd`, `ls`, and `find` commands. The functionality of each of these commands basically matches that of its UNIX counterpart. See `cd`(1), `pwd`(1),`ls`(1), and`find`(1) for details. The `*`, `,`, `?`, and `-` wildcard characters are also supported.

The following commands are supported:

base[=*b*]	Display or set the base. All input and output is governed by the current base. Without the = *b*, displays the current base. Otherwise, sets the current base to *b*. Base is interpreted using the old value of base, so to ensure correctness use the `0`, `0t`, or `0x` prefix when changing the base. The default for base is hexadecimal.
block	Convert the value of dot to a block address.
cd [*dir*]	Change the current directory to directory *dir*. The current values of inode and dot are also updated. If `dir` is not specified, changes directories to inode 2, root (`/`).
directory	If the current inode is a directory, converts the value of dot to a directory slot offset in that directory, and dot now points to this entry.
file	Set the value of dot as a relative block count from the beginning of the file. The value of dot is updated to the first byte of this block.
find *dir* [`-name` *n*] | [`-inum` *i*]	Find files by name or i-number. Recursively searches directory *dir* and below for file names whose i-number matches `i` or whose name matches pattern *n*. Only one of the two options (`-name` or `-inum`) can be used at one time. The find `-print` is not necessary or accepted.
fill=*p*	Fill an area of disk with pattern *p*. The area of disk is delimited by dot and count.
inode	Convert the value of dot to an inode address. If successful, the current value of inode is updated as well as the value of dot. As a convenient shorthand, if `:inode`

	appears at the beginning of the line, the value of dot is set to the current inode and that inode is displayed in inode format.
ls [`-R`] [`-l`] *pat1 pat2...*	List directories or files. If no file is specified, the current directory is assumed. Either or both of the options can be used (but, if used, must be specified before the filename specifiers). Wild card characters are available and multiple arguments are acceptable. The long listing shows only the i-number and the name; use the inode command with `?i` to get more information.
override	Toggle the value of override. Some error conditions might be overridden if override is toggled to `on`.
prompt "*p*"	Change the `fsdb` prompt to *p*. *p* must be enclosed in quotes.
pwd	Display the current working directory.
quit	Quit `fsdb`.
tag	Convert the value of dot and if this is a valid tag, print the volume structure according to the tag.
!	Escape to the shell.

Inode Commands

In addition to the above commands, several other commands deal with inode fields and operate directly on the current inode (they still require the colon (`:`). They can be used to more easily display or change the particular fields. The value of dot is only used by the `:db` and `:ib` commands. Upon completion of the command, the value of dot is changed so that it points to that particular field. For example,

```
> :ln=+1
```

increments the link count of the current inode and sets the value of dot to the address of the link count field.

The following inode commands are supported:

at	Access time
bs	Block size
ct	Creation time
gid	Group id
ln	Link number
mt	Modification time

md Mode

maj Major device number

min Minor device number

nm This command actually operates on the directory name field. Once poised at the desired directory entry (using the `directory` command), this command allows you to change or display the directory name. For example,

```
> 7:dir:nm="foo"
```

gets the 7th directory entry of the current inode and changes its name to `foo`. Directory names cannot be made larger than the field allows. If an attempt is made to make a directory name larger than the field allows,, the string is truncated to fit and a warning message is displayed.

sz File size

uid User ID

uniq Unique ID

Formatted Output

Formatted output comes in two styles and many format types. The two styles of formatted output are: structured and unstructured. Structured output is used to display inodes, directories, and so forth. Unstructured output displays raw data.

Format specifiers are preceded by the slash (`/`) or question mark (`?`) character. *type* is updated as necessary upon completion.

The following format specifiers are preceded by the `?` character:

i Display as inodes in the current base.

d Display as directories in the current base.

The following format specifiers are preceded by the `/` character:

b Display as bytes in the current base.

c Display as characters.

o | O Display as octal shorts or longs.

d | D Display as decimal shorts or longs.

x | X Display as hexadecimal shorts or longs.

EXAMPLES

EXAMPLE 1 Using fsdb as a calculator for complex arithmetic

The following command displays `2010` in decimal format, and is an example of using `fsdb` as a calculator for complex arithmetic.

```
> 2000+400%(20+20)=D
```

EXAMPLE 2 Using fsdb to display an i-number in idode fomat

The following command displays the i-number `386` in inode format.`386` becomes the current inode.

```
> 386:ino?i
```

EXAMPLE 3 Using fsdb to change the link count

The following command changes the link count for the current inode to `4`.

```
> :ln=4
```

EXAMPLE 4 Using fsdb to increment the link count

The following command increments the link count by `1`.

```
> :ln=+1
```

EXAMPLE 5 Using fsdb to display the creation time as a hexadecimal long

The following command displays the creation time as a hexadecimal long.

```
> :ct=X
```

EXAMPLE 6 Using fsdb to display the modification time in time format

The following command displays the modification time in time format.

```
> :mt=t
```

EXAMPLE 7 Using fsdb to display in ASCII

The following command displays, in ASCII, block `0` of the file associated with the current inode.

```
> 0:file/c
```

EXAMPLE 8 Using fsdb to display the directory enteries for the root inode

The following command displays the first block's directory entries for the root inode of this file system. This command stops prematurely if the EOF is reached.

```
> 2:ino,*?d
```

EXAMPLE 9 Using fsdb to change the current inode

The following command changes the current inode to that associated with the `5`th directory entry (numbered from `0`) of the current inode. The first logical block of the file is then displayed in ASCII.

EXAMPLE 9 Using fsdb to change the current inode *(Continued)*

```
> 5:dir:inode; 0:file,*/c
```

EXAMPLE 10 Using fsdb to change the i-number

The following command changes the i-number for the 7th directory slot in the root directory to 3.

```
> 2:inode; 7:dir=3
```

EXAMPLE 11 Using fsdb to change the name field

The following command changes the *name* field in the directory slot to name.

```
 > 7:dir:nm="name"
```

EXAMPLE 12 Using fsdb to display the a block

The following command displays the 3rd block of the current inode as directory entries.

EXAMPLE 13 Using fsdb to set the contents of address

The following command sets the contents of address 2050 to 0xffffffff. 0xffffffff can be truncated, depending on the current type.

```
> 2050=0xffff
```

EXAMPLE 14 Using fsdb to place an ASCII string at an address

The following command places the ASCII string this is some text at address 1c92434.

```
> 1c92434="this is some text"
```

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWudf

SEE ALSO

clri(1M), fsck_udfs(1M), dir(4), attributes(5)

NAME

fsdb_ufs – ufs file system debugger

SYNOPSIS

fsdb `-F ufs` [*generic_options*] [*specific_options*] *special*

DESCRIPTION

The `fsdb_ufs` command is an interactive tool that can be used to patch up a damaged UFS file system. It has conversions to translate block and i-numbers into their corresponding disk addresses. Also included are mnemonic offsets to access different parts of an inode. These greatly simplify the process of correcting control block entries or descending the file system tree.

`fsdb` contains several error-checking routines to verify inode and block addresses. These can be disabled if necessary by invoking `fsdb` with the `-o` option or by the use of the `o` command.

`fsdb` reads a block at a time and will therefore work with raw as well as block I/O devices. A buffer management routine is used to retain commonly used blocks of data in order to reduce the number of read system calls. All assignment operations result in an immediate write-through of the corresponding block. Note that in order to modify any portion of the disk, `fsdb` must be invoked with the `w` option.

Wherever possible, `adb`-like syntax was adopted to promote the use of `fsdb` through familiarity.

OPTIONS

The following option is supported:

`-o` Specify UFS file system specific options. These options can be any combination of the following separated by commas (with no intervening spaces). The options available are:

`?`	Display usage
`o`	Override some error conditions
`p='string'`	set prompt to string
`w`	open for write

USAGE

Numbers are considered hexadecimal by default. However, the user has control over how data is to be displayed or accepted. The `base` command will display or set the input/output base. Once set, all input will default to this base and all output will be shown in this base. The base can be overridden temporarily for input by preceding hexadecimal numbers with '`0x`', preceding decimal numbers with '`0t`', or octal numbers with '`0`'. Hexadecimal numbers beginning with `a-f` or `A-F` must be preceded with '`0x`' to distinguish them from commands.

Disk addressing by `fsdb` is at the byte level. However, `fsdb` offers many commands to convert a desired inode, directory entry, block, superblock and so forth to a byte address. Once the address has been calculated, `fsdb` will record the result in dot (`.`).

Several global values are maintained by `fsdb`:

- the current base (referred to as `base`),

- the current address (referred to as `dot`),
- the current inode (referred to as `inode`),
- the current count (referred to as `count`),
- and the current type (referred to as `type`).

Most commands use the preset value of `dot` in their execution. For example,

```
> 2:inode
```

will first set the value of `dot` to 2, ':', will alert the start of a command, and the `inode` command will set `inode` to 2. A count is specified after a ',', Once set, `count` will remain at this value until a new command is encountered which will then reset the value back to 1 (the default). So, if

```
> 2000,400/X
```

is typed, 400 hex longs are listed from 2000, and when completed, the value of `dot` will be `2000 + 400 * sizeof (long)`. If a RETURN is then typed, the output routine will use the current values of `dot`, `count`, and `type` and display 400 more hex longs. A '*' will cause the entire block to be displayed.

End of fragment, block and file are maintained by `fsdb`. When displaying data as fragments or blocks, an error message will be displayed when the end of fragment or block is reached. When displaying data using the `db`, `ib`, `directory`, or `file` commands an error message is displayed if the end of file is reached. This is mainly needed to avoid passing the end of a directory or file and getting unknown and unwanted results.

An example showing several commands and the use of RETURN would be:

```
> 2:ino; 0:dir?d
      or
> 2:ino; 0:db:block?d
```

The two examples are synonymous for getting to the first directory entry of the root of the file system. Once there, any subsequent RETURN (or +, -) will advance to subsequent entries. Note that

```
> 2:inode; :ls
      or
> :ls /
```

is again synonymous.

Expressions

The symbols recognized by `fsdb` are:

RETURN — update the value of `dot` by the current value of `type` and display using the current value of `count`.

`#`	numeric expressions may be composed of +, -, *, and % operators (evaluated left to right) and may use parentheses. Once evaluated, the value of `dot` is updated.
`,` *count*	count indicator. The global value of `count` will be updated to `count`. The value of `count` will remain until a new command is run. A count specifier of '*' will attempt to show a *blocks's* worth of information. The default for `count` is 1.
`?` *f*	display in structured style with format specifier *f*. See `Formatted`Output.
`/` *f*	display in unstructured style with format specifier *f* See `Formatted`Output.
`.`	the value of `dot`.
`+`*e*	increment the value of `dot` by the expression *e.* The amount actually incremented is dependent on the size of `type`: `dot = dot + e * sizeof (type)` The default for *e* is `1`.
`-`*e*	decrement the value of `dot` by the expression *e*. See `+`.
`*`*e*	multiply the value of `dot` by the expression *e.* Multiplication and division don't use `type`. In the above calculation of `dot`, consider the `sizeof(type)` to be `1`.
`%`*e*	divide the value of `dot` by the expression *e*. See `*`.
`<` *name*	restore an address saved in register *name*. *name* must be a single letter or digit.
`>` *name*	save an address in register *name*. *name* must be a single letter or digit.
`=` *f*	display indicator. If *f* is a legitimate format specifier. then the value of `dot` is displayed using the format specifier *f*. See `Formatted`Output. Otherwise, assignment is assumed See `=`.
`=` `[`*s*`]` `[`*e*`]`	assignment indicator. The address pointed to by `dot` has its contents changed to the value of the expression *e* or to the ASCII representation of the quoted (") string *s*. This may be useful for changing directory names or ASCII file information.
`=+` *e*	incremental assignment. The address pointed to by `dot` has its contents incremented by expression *e*.
`=-` *e*	decremental assignment. The address pointed to by `dot` has its contents decremented by expression *e*.

Commands

A command must be prefixed by a ':' character. Only enough letters of the command to uniquely distinguish it are needed. Multiple commands may be entered on one line by separating them by a SPACE, TAB or ';'.

In order to view a potentially unmounted disk in a reasonable manner, `fsdb` offers the `cd`, `pwd`, `ls` and `find` commands. The functionality of these commands substantially matches those of its UNIX counterparts. See individual commands for details. The '*', '?', and '[-]' wild card characters are available.

`base=b`	display or set base. As stated above, all input and output is governed by the current `base`. If the `=b` is omitted, the current `base` is displayed. Otherwise, the current `base` is set to *b.* Note that this is interpreted using the old value of `base`, so to ensure correctness use the '0', '0t', or '0x' prefix when changing the `base`. The default for `base` is hexadecimal.
`block`	convert the value of `dot` to a block address.
`cd` *dir*	change the current directory to directory *dir*. The current values of `inode` and `dot` are also updated. If no *dir* is specified, then change directories to inode `2` ("/").
`cg`	convert the value of `dot` to a cylinder group.
`directory`	If the current `inode` is a directory, then the value of `dot` is converted to a directory slot offset in that directory and `dot` now points to this entry.
`file`	the value of `dot` is taken as a relative block count from the beginning of the file. The value of `dot` is updated to the first byte of this block.
`find` *dir* [`-name` *n*] [`-inum` *i*]	find files by name or i-number. `find` recursively searches directory `dir` and below for filenames whose i-number matches *i* or whose name matches pattern *n*. Note that only one of the two options (-name or -inum) may be used at one time. Also, the -print is not needed or accepted.
`fill=`*p*	fill an area of disk with pattern *p*. The area of disk is delimited by `dot` and `count`.

`fragment`	convert the value of *dot* to a fragment address. The only difference between the `fragment` command and the `block` command is the amount that is able to be displayed.
`inode`	convert the value of *dot* to an inode address. If successful, the current value of `inode` will be updated as well as the value of *dot*. As a convenient shorthand, if ':inode' appears at the beginning of the line, the value of *dot* is set to the current `inode` and that inode is displayed in inode format.
`log_chk`	run through the valid log entries without printing any information and verify the layout.
`log_delta`	count the number of deltas into the log, using the value of dot as an offset into the log. No checking is done to make sure that offset is within the head/tail offsets.
`log_head`	display the header information about the file system logging. This shows the block allocation for the log and the data structures on the disk.
`log_otodb`	return the physical disk block number, using the value of dot as an offset into the log.
`log_show`	display all deltas between the beginning of the log (BOL) and the end of the log (EOL).
`ls`	[`-R`] [`-l`] *pat1 pat2* . . . list directories or files. If no file is specified, the current directory is assumed. Either or both of the options may be used (but, if used, *must* be specified before the filename specifiers). Also, as stated above, wild card characters are available and multiple arguments may be given. The long listing shows only the i-number and the name; use the `inode` command with '?i' to get more information.
`override`	toggle the value of override. Some error conditions may be overriden if override is toggled on.

`prompt` *p*	change the `fsdb` prompt to *p*. *p* must be surrounded by (")s.
`pwd`	display the current working directory.
`quit`	quit `fsdb`.
`sb`	the value of *dot* is taken as a cylinder group number and then converted to the address of the superblock in that cylinder group. As a shorthand, ':sb' at the beginning of a line will set the value of *dot* to *the* superblock and display it in superblock format.
`shadow`	if the current inode is a shadow inode, then the value of *dot* is set to the beginning of the shadow inode data.
`!`	escape to shell

Inode Commands

In addition to the above commands, there are several commands that deal with inode fields and operate directly on the current `inode` (they still require the ':'). They may be used to more easily display or change the particular fields. The value of *dot* is only used by the ':`db`' and ':`ib`' commands. Upon completion of the command, the value of *dot* is changed to point to that particular field. For example,

```
> :ln=+1
```

would increment the link count of the current `inode` and set the value of *dot* to the address of the link count field.

`at` access time.

`bs` block size.

`ct` creation time.

`db` use the current value of *dot* as a direct block index, where direct blocks number from 0 - 11. In order to display the block itself, you need to 'pipe' this result into the `block` or `fragment` command. For example,

```
> 1:db:block,20/X
```

would get the contents of data block field 1 from the inode and convert it to a block address. 20 longs are then displayed in hexadecimal. See `Formatted`Output.

`gid` group id.

`ib` use the current value of *dot* as an indirect block index where indirect blocks number from 0 - 2. This will only get the indirect block itself (the block containing the pointers to the actual blocks). Use the `file` command and start at block 12 to get to the actual blocks.

`ln` link count.

`mt` modification time.

`md` mode.

`maj` major device number.

`min` minor device number.

`nm` although listed here, this command actually operates on the directory name field. Once poised at the desired directory entry (using the *directory* command), this command will allow you to change or display the directory name. For example,

```
> 7:dir:nm="foo"
```

will get the `7`th directory entry of the current `inode` and change its name to foo. Note that names cannot be made larger than the field is set up for. If an attempt is made, the string is truncated to fit and a warning message to this effect is displayed.

`si` shadow inode.

`sz` file size.

`uid` user id.

Formatted Output

There are two styles and many format types. The two styles are structured and unstructured. Structured output is used to display inodes, directories, superblocks and the like. Unstructured displays raw data. The following shows the different ways of displaying:

`?`

`c`	display as cylinder groups
`i`	display as inodes
`d`	display as directories
`s`	display as superblocks
`S`	display as shadow inode data

`/`

`b`	display as bytes
`c`	display as characters
`o O`	display as octal shorts or longs
`d D`	display as decimal shorts or longs
`x X`	display as hexadecimal shorts or longs The format specifier

immediately follows the '/' or '?' character. The values displayed by '/b' and all '?' formats are displayed in the current `base`. Also, `type` is appropriately updated upon completion.

EXAMPLES

EXAMPLE 1 Displaying in Decimal

The following command displays `2010` in decimal (use of `fsdb` as a calculator for complex arithmetic):

```
> 2000+400%(20+20)=D
```

EXAMPLE 2 Displaying an i-number in Inode Format

The following command displays i-number `386` in an inode format. This now becomes the current `inode`:

```
> 386:ino?i
```

EXAMPLE 3 Changing the Link Count

The following command changes the link count for the current `inode` to `4`:

```
> :ln=4
```

EXAMPLE 4 Incrementing the Link Count

The following command increments the link count by `1`:

```
> :ln=+1
```

EXAMPLE 5 Displaying the Creation Time

The following command displays the creation time as a hexadecimal long:

```
> :ct=X
```

EXAMPLE 6 Displaying the Modification Time

The following command displays the modification time in time format:

```
> :mt=t
```

EXAMPLE 7 Displaying in ASCII

The following command displays in ASCII, block zero of the file associated with the current `inode`:

```
> 0:file/c
```

EXAMPLE 8 Displaying the First Block's Worth of Directorty Entries

The following command displays the first block's worth of directory entries for the root inode of this file system. It will stop prematurely if the EOF is reached:

```
> 2:ino,*?d
```

EXAMPLE 9 Displaying Changes to the Current Inode

The following command displays changes the current inode to that associated with the `5`th directory entry (numbered from zero) of the current `inode`. The first logical block of the file is then displayed in ASCII:

```
> 5:dir:inode; 0:file,*/c
```

EXAMPLE 10 Displaying the Superblock

The following command displays the superblock of this file system:

```
> :sb
```

EXAMPLE 11 Displaying the Cylinder Group

The following command displays cylinder group information and summary for cylinder group `1`:

```
> 1:cg?c
```

EXAMPLE 12 Changing the i-number

The following command changes the i-number for the seventh directory slot in the root directory to `3`:

```
> 2:inode; 7:dir=3
```

EXAMPLE 13 Displaying as Directory Entries

The following command displays the third block of the current `inode` as directory entries:

```
> 2:db:block,*?d
```

EXAMPLE 14 Changing the Name Field

The following command changes the name field in the directory slot to *name*:

```
> 7:dir:nm="name"
```

EXAMPLE 15 Getting and Filling Elements

The following command gets fragment `3c3` and fill `20 type` elements with `0x20`:

```
> 3c3:fragment,20:fill=0x20
```

EXAMPLE 16 Setting the Contents of an Address

The following command sets the contents of address `2050` to `0xffffffff`. `0xffffffff` may be truncated depending on the current `type`:

```
> 2050=0xffff
```

EXAMPLE 17 Placing ASCII

The following command places the ASCII for the string at `1c92434`:

```
> 1c92434="this is some text"
```

EXAMPLE 18 Displaying Shadow Inode Data

The following command displays all of the shadow inode data in the shadow inode associated with the root inode of this file system:

```
> 2:ino:si:ino;0:shadow,*?S
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`clri`(1M), `fsck_ufs`(1M), `dir_ufs`(4), `fs_ufs`(4), `attributes`(5)

WARNINGS

Since `fsdb` reads the disk raw, extreme caution is advised in determining its availability of `fsdb` on the system. Suggested permissions are 600 and owned by bin.

NOTES

The old command line syntax for clearing i-nodes using the ufs-specific `'-z i-number'` option is still supported by the new debugger, though it is obsolete and will be removed in a future release. Use of this flag will result in correct operation, but an error message will be printed warning of the impending obsolesence of this option to the command. The equivalent functionality is available using the more flexible `clri`(1M) command.

NAME

fsirand – install random inode generation numbers

SYNOPSIS

fsirand [-p] *special*

DESCRIPTION

`fsirand` installs random inode generation numbers on all the inodes on device *special*, and also installs a file system ID in the superblock. This helps increase the security of file systems exported by NFS.

`fsirand` must be used only on an unmounted file system that has been checked with `fsck`(1M) The only exception is that it can be used on the root file system in single-user mode, if the system is immediately re-booted afterwards.

OPTIONS

`-p` Print out the generation numbers for all the inodes, but do not change the generation numbers.

USAGE

See `largefile`(5) for the description of the behavior of `fsirand` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`fsck`(1M), `attributes`(5), `largefile`(5)

NAME

fssnap – create temporary snapshots of a file system

SYNOPSIS

/usr/sbin/fssnap [-F *FSType*] [-V] -o *special_options* [*mount-point* | *special*]

/usr/sbin/fssnap -d [-F *FSType*] [-V] -o *special_options* [*mount-point* | *special*]

/usr/sbin/fssnap -i [-F *FSType*] [-V] -o *special_options* [*mount-point* | *special*]

DESCRIPTION

The `fssnap` command creates a stable, read-only snapshot of a file system when given either an active mount point or a special device containing a mounted file system, as in the first form of the synopsis. A snapshot is a temporary image of a file system intended for backup operations.

While the snapshot file system is stable and consistent, an application updating files when the snapshot is created might leave these files in an internally inconsistent, truncated, or otherwise unusable state. In such a case, the snapshot will contain these partially written or corrupted files. It is a good idea to ensure active applications are suspended or checkpointed and their associated files are also consistent during snapshot creation.

Note that file access times are not updated while the snapshot is being created.

A path to the virtual device that contains this snapshot is printed to standard output when a snapshot is created.

OPTIONS

The following options are supported:

`-d`	Deletes the snapshot associated with the given file system.
`-F` *FSType*	Specifies the file system type to be used. The `FSType` should either be specified here or be determined by matching the block special device with an entry in the `/etc/vfstab` table, or by consulting `/etc/default/fs`.
`-i`	Displays the state of any given `FSType` snapshot. If a mount-point or device is not given, a list of all snapshots on the system is displayed. When a mount-point or device is specified, detailed information is provided for the specified file system snapshot by default. The format and meaning of this information is file-system dependent. See the `FSType`-specific `fssnap` man page for details.
`-o` *special_options*	See the `FSType`-specific man page for `fssnap`.
`-V`	Echoes the complete command line, but does not execute the command.

OPERANDS

The following operands are supported:

mount-point The directory where the file system resides.

special The physical device for the system, such as `/dev/dsk/c0t0d0s7`.

EXAMPLES

See `FSType`-specific man pages for examples.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

FILES

`/etc/vfstab` Specifies file system type.

`/etc/default/fs` Specifies the default local file system type.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`fssnap_ufs`(1M), `attributes`(5)

NOTES

This command might not be supported for all `FSTypes`.

NAME

fssnap_ufs – create a temporary snapshot of a UFS file system

SYNOPSIS

fssnap [-F] [ufs] [*generic-options*] -o *backing-store=path*, [*specific-options*] *mount-point* | *special*

fssnap [-F ufs] [-d] [*generic-options*] [-o *specific-options*] *mount-point* | *special*

fssnap [-F ufs] [-i] [*generic-options*] [-o *specific-options*] *mount-point* | *special*

DESCRIPTION

The `fssnap` command queries, creates, or deletes a temporary snapshot of a UFS file system. A snapshot is a point-in-time image of a file system that provides a stable and unchanging device interface for backups.

When creating a file system snapshot, you must specify the file system to be captured and the backing-store file.

The backing-store file is one in which the snapshot subsystem saves old file system data before it is overwritten. The destination path must have enough free space to hold the backing-store file, whose size varies with the amount of activity on the file system. This location must be different from the file system that is being captured in a snapshot. The backing-store file can reside on any type of file system, including another UFS file system or an NFS–mounted file system.

OPTIONS

The following options are supported:

`-d`
Deletes the snapshot associated with the given file system.

`-i`
Displays the state of one or all UFS snapshots. If a mount-point or device is not specified, a list of all snapshots on the system is displayed. When a mount-point or device is specified, detailed information is provided for the specified file system snapshot by default.

Use the `-o` options with the `-i` option to specify what snapshot information is displayed. Since this feature is provided primarily for use in scripts and on the command line, no labels are displayed for the data. Sizes are all in bytes, and the output is not internationalized or localized. The information is displayed on one line per option. Unrecognized options display a single `?` on the line. One line per option guarantees that there are the same number of lines as options specified and there is a one–to-one correspondence between an output line and an option.

The following `-o` options display specific information for a given snapshot. See the EXAMPLES section for examples of how to use these options.

`snapnumber`
Display the snapshot number.

`blockdevname`
Display the block device path.

`rawdevname`
Display the raw device path.

`mountpoint`
Display the mount point of the master file system.

`state`
Display the state of the snapshot device.

`backing-store`
Display the location of the backing-store file.

`backing-store-len`
Display the size of the backing-store file.

`maxsize`
Display the max size of the backing-store file.

`createtime`
Display the time that the snapshot was created.

`chunksize`
Display the copy-on-write granularity.

`-o` *specific-options*
Without `-d` or `-i`, the default action is to create a snapshot. Specify the following options when creating a snapshot. All of these options are discretionary, except for the backing-store file (`bs`), which is required.

`backing-store=`*path*
Uses *path* as the backing-store file. *path* must not reside on the file system that is being captured in a snapshot.

path must exist, and must be either a directory, a regular file, or a raw device. If *path* is a directory, then a temporary file is created and held open. That device is then used as-is. The option can be abbreviated as `bf=` *path* or `bs=`*path*.

unlink
Unlinks the backing-store file after the snapshot is created. This option specifies that the backing-store file does not need to be removed manually when the snapshot is deleted. This might make administration more difficult since the file is not visible in the file system. If this option is not specified, the backing-store files should be removed manually after the snapshot is deleted.

chunksize=*n* [k,m,g]
Uses *n* for the chunk size. Chunk size is the granularity of the data that is sent to the backing store.

Specify `chunksize` in the following units: `k` for kilobytes, `m` for megabytes, or `g` for gigabytes. By default, chunk size is four times the block size of the file system (typically `32k`).

maxsize=*n*[k,m,g]
Does not allow the size of the backing-store file to exceed *n*, where *n* is the unit specified. The snapshot is deleted automatically when the backing-store file exceeds `maxsize`.

Specify `maxsize` in the following units: `k` for kilobytes, `m` for megabytes, or `g` for gigabytes.

raw
Displays to standard output the name of the raw device instead of the block device when a snapshot is created. The block device is printed by default (when `raw` is not specified). This option makes it easier to embed `fssnap` commands in the command line for commands that require the raw device instead. Both devices are always created. This option affects only the output.

OPERANDS

The following operands are supported:

mount-point The directory where the file system resides.

special The physical device for the file system, such as `/dev/dsk/c0t0d0s7`.

EXAMPLES

EXAMPLE 1 Creating a Snapshot of a File System

The following example creates a snapshot of a file system. The block special device created for the snapshot is `/dev/fssnap/0`.

```
# fssnap -F ufs -o backing-store=/var/tmp /export/home
/dev/fssnap/0
```

EXAMPLE 2 Backing Up a File System Snapshot Without Having To Unmount the File System

The following example backs up a file system snapshot without having to unmount the file system. Since `ufsdump` requires the path to a raw device, the `raw` option is used. The `/export/home` file system snapshot is removed in the second command.

```
# ufsdump 0uf /dev/rmt/0 `fssnap -F ufs
      -o raw,bs=/dev/rdsk/c0t3d0s6 /export/home`
<output from ufsdump>
# fssnap -F ufs -d /export/home
```

EXAMPLE 3 Backing Up a File System

When backing up a file system, do not let the backing-store file exceed `400` Mbytes. The second command removes the `/export/home` file system snapshot.

```
# ufsdump 0uf /dev/rmt/0 `fssnap -F ufs
      -o maxsize=400m,backing-store=/export/snap,raw
      /export/home`
# fssnap -F ufs -d /export/home
```

EXAMPLE 3 Backing Up a File System *(Continued)*

EXAMPLE 4 Performing an Incremental Dump of a Snapshot

The following example uses `ufsdump` to back up a snapshot of `/dev/rdsk/c0t3d0s2`. Note the use of the `N` option to `ufsdump`, which writes the name of the device being dumped, rather than the name of the snapshot device, to `/etc/dumpdates` file. See `ufsdump`(1M) for details on the `N` flag.

```
# ufsdump lfNu /dev/rmt/0 /dev/rdsk/c0t3d0s2 'fssnap -F ufs
-o raw,bs=/export/scratch,unlink /dev/rdsk/c0t3d0s2'
```

EXAMPLE 5 Finding Out What Snapshots Currently Exist

The following command displays the currently existing snapshots.

```
# fssnap -i
0  /src
1  /export/home
<output continues>
```

EXAMPLE 6 Mounting a File System Snapshot

The following example creates a file system snapshot. After you create a file system snapshot, mount it on `/tmp/mount` for temporary read-only access.

```
# fssnap -F ufs -o backing-store=/nfs/server/scratch /export/home
/dev/fssnap/1
# mkdir /tmp/mount
# mount -F ufs -o ro /dev/fssnap/1 /tmp/mount
```

EXAMPLE 7 Creating a File System Snapshot and Unlinking the Backing-store File

The following example creates a file system snapshot and unlinks the backing-store file. After creating a file system snapshot and unlinking the backing-store file, check the state of the snapshot.

```
# fssnap -o bs=/scratch,unlink /src
/dev/fssnap/0
# fssnap -i /src
Snapshot number               : 0
Block Device                  : /dev/fssnap/0
Raw Device                    : /dev/rfssnap/0
Mount point                   : /src
Device state                  : active
Backing store path            : /scratch/snapshot2 <UNLINKED>
Backing store size            : 192 KB
Maximum backing store size    : Unlimited
Snapshot create time          : Sat May 06 10:55:11 2000
Copy-on-write granularity     : 32 KB
```

EXAMPLE 7 Creating a File System Snapshot and Unlinking the Backing-store File *(Continued)*

EXAMPLE 8 Displaying the Size and Location of the Backing-store File and the Creation Time for the Snapshot

The following example displays the size of the backing-store file in bytes, the location of the backing store, and the creation time for the snapshot of the `/test` file system.

```
# fssnap -i -o backing-store-len,backing-store,createtime /test
196608
/snapshot2
Sat May 6 10:55:11 2000
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

The script-readable output mode is a stable interface that can be added to, but will not change. All other interfaces are subject to change.

NOTES

The `fssnap` device files should be treated like a regular disk block or character device.

The association between a file system and the snapshot is lost when the snapshot is deleted or the system reboots. Snapshot persistence across reboots is not currently supported.

To avoid unnecessary performance impacts, perform the snapshot and system backup when the system is least active.

NAME

fstyp – determine file system type

SYNOPSIS

`fstyp` `[-v]` *special*

DESCRIPTION

`fstyp` allows the user to determine the file system type of unmounted file systems using heuristic programs.

An `fstyp` module for each file system type to be checked is executed; each of these modules applies an appropriate heuristic to determine whether the supplied *special* file is of the type for which it checks. If it is, the program prints on standard output the usual file system identifier for that type (for example, ''ufs'') and exits with a return code of `0`; if none of the modules succeed, the error message `unknown_fstyp (no matches)` is returned and the exit status is `1`. If more than one module succeeds, the error message `unknown_fstyp (multiple matches)` is returned and the exit status is `2`.

OPTIONS

`-v` Produce verbose output. This is usually information about the file systems superblock and varies across different *FSTypes*. See `fs_ufs`(4), `mkfs_ufs`(1M), and `tunefs`(1M) for details.

USAGE

See `largefile`(5) for the description of the behavior of `fstyp` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`mkfs_ufs`(1M), `tunefs`(1M), `fs_ufs`(4), `attributes`(5), `largefile`(5), `hsfs`(7FS), `pcfs`(7FS)

NOTES

The use of heuristics implies that the result of `fstyp` is not guaranteed to be accurate.

NAME

ftpaddhost – set up a virtual FTP host

SYNOPSIS

ftpaddhost -c | -l [-b] [-x *xferlog*] *hostname root_dir*

DESCRIPTION

The ftpaddhost script is executed by the super user to set up virtual FTP hosts. The ftpaddhost command configures the virtual host *hostname* under directory *root_dir*. The value of *hostname* can be an IP address or the name of a host.

OPTIONS

The ftpaddhost script supports the following options:

-b — Create a banner for the virtual host. This option is useful to confirm that the virtual host is working.

-c — Configure complete virtual hosting. This option allows each virtual host to have its own version of the ftpaccess, ftpconversions, ftpgroups, ftphosts, and ftpusers files. The master version of each of these configuration files is copied from the /etc/ftpd directory and placed in the /etc/ftpd/virtual-ftpd/*hostname* directory. If the /etc/ftpusers file exists it is appended to the virtual ftpusers file. If a virtual host lacks its own version of a configuration file, the master version is used.

-l — Configure limited virtual hosting. This option allows a small number of parameters to be configured differently for a virtual host. See the virtual keyword on the ftpaccess(4) manual page.

-x *xferlog* — Create a logfile entry such that the transfer logs for the virtual host are written to the specified file. An absolute path must be specified for the *xferlog* file.

OPERANDS

The following operands are supported:

hostname — The host name or IP address of the virtual server.

root_dir — The absolute pathname of the directory under which the virtual server is set up.

EXIT STATUS

The following exit values are returned:

0 — Successful completion

1 — Improper usage of the command

2 — Command failed

FILES

/etc/ftpd/virtual-ftpd/*hostname*
The configuration files directory for the virtual host *hostname*.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWftpu

SEE ALSO `ftpconfig`(1M), `in.ftpd`(1M), `ftpaccess`(4), `ftpconversions`(4), `ftpgroups`(4), `ftphosts`(4), `ftpusers`(4), `xferlog`(4), `attributes`(5)

NAME

ftpconfig – set up anonymous FTP

SYNOPSIS

ftpconfig [*ftpdir*]

ftpconfig -d *ftpdir*

DESCRIPTION

The ftpconfig script is executed by the super user to set up anonymous FTP. Anonymous FTP allows users to remotely log on to the FTP server by specifying the user name ftp or anonymous and the user's email address as password. The anonymous users are logged on to the FTP Server and given access to a restricted file area with its own file system root. See chroot(2). The FTP area has its own minimal system files.

This command will copy and set up all the components needed to operate an anonymous FTP server, including creating the *ftp* user account, creating device nodes, copying /usr/lib files, and copying timezone data. The passwd and group files set up have been stripped down to prevent malicious users from finding login names on the server. The anonymous file area will be placed under ftpdir. If the *ftp* user account already exists, then the current FTP area is used, and the system files in it are updated. All other files are left untouched. This command should be run to update the anonymous FTP area's configuration whenever a system patch is installed, or the system is upgraded.

OPTIONS

-d Create a new or update an existing *ftpdir* without creating or updating the *ftp* user account. Use this option when creating guest FTP user accounts.

OPERANDS

The following operands are supported:

ftpdir The absolute pathname of the directory under which the anonymous FTP area is set up.

EXIT STATUS

The following exit values are returned:

0 Successful completion

1 Improper usage of the command

2 Command failed

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWftpu

SEE ALSO

ftpaddhost(1M), in.ftpd(1M), useradd(1M), chroot(2), attributes(5)

NAME

ftprestart – restart previously shutdown FTP Servers

SYNOPSIS

```
ftprestart [-V]
```

DESCRIPTION

Use the `ftprestart` command to restart an FTP Server previously shut down by means of `ftpshut`(1M). The `ftprestart` command reads the `shutdown` capability from the `ftpaccess`(4) file to determine the path of the shutdown message files. It then reenables the FTP Server by removing any shutdown message files in the anonymous and virtual FTP Server area, as well as the system wide shutdown message file.

OPTIONS

The `ftprestart` command supports the following options:

`-V` Display program copyright and version information, then terminate.

EXAMPLES

EXAMPLE 1 Sample Output from `ftprestart`

The following example shows sample output from the `ftprestart` command:

```
example% ftprestart
ftprestart: /export/home/ftp/etc/ftpd/shutdown.msg removed.
ftprestart: /export/home/virtual1/etc/ftpd/shutdown.msg removed.
ftprestart: /etc/ftpd/shutdown.msg removed.
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

FILES

`/etc/ftpd/ftpaccess`

`/etc/ftpd/ftpservers`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWftpu

SEE ALSO

`ftpshut`(1M), `in.ftpd`(1M), `ftpaccess`(4), `ftpservers`(4), `attributes`(5)

NAME

ftpshut – close down the FTP Servers at a given time

SYNOPSIS

`ftpshut` `[-V] [-v] [-l` *min*`] [-d` *min*`]` *time* `[`*warning-message*`...]`

DESCRIPTION

The `ftpshut` command provides an automated shutdown procedure that the superuser can use to notify FTP users when the FTP Server is shutting down.

Ten minutes before shutdown, or immediately if the value of *time* is less than ten minutes, any new FTP Server connections will be disabled. You may adjust the shutdown of new FTP Server connections by means of the `-l` option.

Five minutes before shutdown, or immediately if the value of *time* is less than five minutes, all current FTP connections will be disconnected. You may adjust the shutdown of current FTP connections by means of the `-d` option.

The `ftpshut` command creates shutdown message files that the FTP Server uses to determine when to shutdown. Separate shutdown message files are created in the anonymous and virtual host FTP Server areas, in addition to the system wide shutdown message file. Once the shutdown occurs, the server continues to refuse connections until the appropriate shutdown message file is removed. This normally is done by using the `ftprestart`(1M) command. The location of the shutdown message file is specified by the `shutdown` capability in the `ftpaccess` file.

The following magic cookies are available:

`%s`	The time system is going to shut down.
`%r`	The time new connections will be denied.
`%d`	The time current connections will be dropped.
`%C`	The current working directory.
`%E`	The maintainer's email address as defined in the `ftpaccess` file.
`%F`	The free space in the partition of CWD, in kilobytes.
`%L`	The local host name.
`%M`	The maximum allowed number of users in this class.
`%N`	The current number of users in this class.
`%R`	The remote host name.
`%T`	The local time (form Thu Nov 15 17:12:42 1990).
`%U`	The username given at login time.

OPTIONS

The `ftpshut` command supports the following options:

`-V`	Display program copyright and version information, then terminate.
`-d` *min*	The time ahead of shutdown, in minutes, that existing connections will be disconnected upon completion of their current or next (if idle) FTP request. .

`-l` *min* The time ahead of shutdown, in minutes, that new connections will be refused.

`-v` Verbose. Output the pathname of the shutdown message files created.

OPERANDS

The `ftpshut` command supports the following operands:

time The *time* at which `ftpshut` will bring the FTP Servers down. *time* can have a value of `now`, which indicates an immediate shutdown. Alternatively, *time* can specify a future time in one of two formats: *+number* or *HHMM*. The first form brings the FTP Server down in *number* minutes. The second brings the FTP Server down at the time of day indicated, using a 24-hour clock format. When using the absolute time format, ou can only specify times between now and 23:59.

warning-message The message to display that warns of the imminent shutdown. The *warning-message* will be formatted at 70 characters wide. `ftpshut` knows the actual string length of the magic cookies. If no warning-message is supplied, the default message "`System shutdown at %s`" is used.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

FILES

`/etc/ftpd/ftpaccess`

`/etc/ftpd/ftpservers`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWftpu

SEE ALSO

`in.ftpd`(1M), `ftprestart`(1M), `shutdown`(1M), `ftpaccess`(4), `ftpservers`(4), `attributes`(5)

NAME

fuser – identify processes using a file or file structure

SYNOPSIS

```
/usr/sbin/fuser [-c | -f] [-nu] [-k | -s sig]files [ [- ] [-c | -f]
     [-nu] [-k | -s sig]files] ...
```

DESCRIPTION

`fuser` displays the process IDs of the processes that are using the *files* specified as arguments.

Each process ID is followed by a letter code. These letter codes are interpreted as follows: if the process is using the file as

c — Indicates that the process is using the file as its current directory.

m — Indicates that the process is using a file mapped with `mmap`(2). See `mmap`(2) for details.

n — Indicates that the process is holding a non-blocking mandatory lock on the file.

o — Indicates that the process is using the file as an open file.

r — Indicates that the process is using the file as its root directory.

t — Indicates that the process is using the file as its text file.

y — Indicates that the process is using the file as its controlling terminal.

For block special devices with mounted file systems, all processes using any file on that device are listed. For all types of files (text files, executables, directories, devices, and so forth), only the processes using that file are reported.

If more than one group of files are specified, the options may be respecified for each additional group of files. A lone dash cancels the options currently in force.

The process IDs are printed as a single line on the standard output, separated by spaces and terminated with a single new line. All other output is written on standard error.

Any user can run fuser, but only the superuser can terminate another user's process.

OPTIONS

The following options are supported:

`-c` — Reports on files that are mount points for file systems, and any files within that mounted file system.

`-f` — Print a report for the named file, not for files within a mounted file system.

`-k` — Sends the `SIGKILL` signal to each process. Since this option spawns kills for each process, the kill messages may not show up immediately (see `kill`(2)).

`-n` — Lists only processes with non-blocking mandatory locks on a file.

`-s` *sig* — Send a signal to each process. The *sig* option argument specifies one of the symbolic names defined in the `<signal.h>` header, or a decimal integer

signal number. If *sig* is a symbolic name, it is recognized in a case-independent fashion, without the `SIG` prefix. The `-k` option is equivalent to `-s KILL` or `-s 9`.

`-u` Displays the user login name in parentheses following the process ID.

EXAMPLES

EXAMPLE 1 Reporting on the Mount Point and Files

The following example reports on the mount point and files within the mounted file system.

```
fuser -c /export/foo
```

EXAMPLE 2 Restricting Output When Reporting on the Mount Point and Files

The following example reports on the mount point and files within the mounted file system, but the output is restricted to processes that hold non-blocking mandatory locks.

```
fuser -cn /export/foo
```

EXAMPLE 3 Sending SIGTERM to Processes Holding a Non-Blocking Mandatory Lock

The following command sends `SIGTERM` to any processes that hold a non-blocking mandatory lock on file `/export/foo/my_file`.

```
fuser -fn -s term /export/foo/my_file
```

ENVIRONMENT VARIABLES

See `environ`(5) for descriptions of the following environment variables that affect the execution of `fuser`: `LANG`, `LC_ALL` `LC_CTYPE`, `LC_MESSAGES`, and `NLSPATH`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`ps`(1), `mount`(1M), `kill`(2), `mmap`(2), `signal`(3C), `attributes`(5), `environ`(5)

NOTES

Because `fuser` works with a snapshot of the system image, it may miss processes that begin using a file while `fuser` is running. Also, processes reported as using a file may have stopped using it while `fuser` was running. These factors should discourage the use of the `-k` option.

NAME

fwtmp, wtmpfix – manipulate connect accounting records

SYNOPSIS

/usr/lib/acct/fwtmp [-ic]

/usr/lib/acct/wtmpfix [*file*...]

DESCRIPTION

`fwtmp` reads from the standard input and writes to the standard output, converting binary records of the type found in `/var/adm/wtmpx` to formatted ASCII records. The ASCII version is useful when it is necessary to edit bad records.

`wtmpfix` examines the standard input or named files in `utmpx` format, corrects the time/date stamps to make the entries consistent, and writes to the standard output. A hyphen (–) can be used in place of *file* to indicate the standard input. If time/date corrections are not performed, `acctcon`(1M) will fault when it encounters certain date-change records.

Each time the date is set, a pair of date change records are written to `/var/adm/wtmpx`. The first record is the old date denoted by the string "`old time`" placed in the `line` field and the flag `OLD_TIME` placed in the `type` field of the `utmpx` structure. The second record specifies the new date and is denoted by the string `new time` placed in the `line` field and the flag `NEW_TIME` placed in the `type` field. `wtmpfix` uses these records to synchronize all time stamps in the file.

In addition to correcting time/date stamps, `wtmpfix` will check the validity of the `name` field to ensure that it consists solely of alphanumeric characters or spaces. If it encounters a name that is considered invalid, it will change the login name to `INVALID` and write a diagnostic to the standard error. In this way, `wtmpfix` reduces the chance that `acctcon` will fail when processing connect accounting records.

OPTIONS

`-ic` Denotes that input is in ASCII form, and output is to be written in binary form.

FILES

`/var/adm/wtmpx` history of user access and administration information

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWaccu

SEE ALSO

`acctcom`(1), `ed`(1), `acct`(1M), `acctcms`(1M), `acctcon`(1M), `acctmerg`(1M), `acctprc`(1M), `acctsh`(1M), `runacct`(1M), `acct`(2), `acct`(3HEAD), `utmpx`(4), `attributes`(5)

System Administration Guide: Basic Administration

NAME

getdev – lists devices based on criteria

SYNOPSIS

getdev [-ae] [*criteria*...] [*device*...]

DESCRIPTION

getdev generates a list of devices that match certain criteria. The criteria includes a list of attributes (given in expressions) and a list of devices. If no criteria are given, all devices are included in the list.

Devices must satisfy at least one of the criteria in the list unless the -a option is used. Then, only those devices which match all of the criteria in a list will be included.

Devices which are defined on the command line and which match the criteria are included in the generated list. However, if the -e option is used, the list becomes a set of devices to be *excluded* from the list. See OPTIONS and OPERANDS.

OPTIONS

The following options are supported:

-a Specifies that a device must match all criteria to be included in the list generated by this command. The option has no effect if no criteria are defined.

-e Specifies that the list of devices which follows on the command line should be *excluded* from the list generated by this command. Without the -e the named devices are *included* in the generated list. The flag has no effect if no devices are defined.

OPERANDS

The following operands are supported:

criteria Defines the criteria that a device must match to be included in the generated list. *criteria* is specified by expressions.

There are four possible expression types which the criteria specified in the *criteria* argument may follow:

attribute=*value* Selects all devices whose attribute *attribute* is defined and is equal to *value*.

attribute!=*value* Selects all devices whose attribute *attribute* is defined and does not equal *value*.

attribute:* Selects all devices which have the attribute *attribute* defined.

attribute!:* Selects all devices which do not have the attribute *attribute* defined.

See the putdev(1M) manual page for a complete listing and description of available attributes.

device Defines the devices which should be included in the generated list. This can be the pathname of the device or the device alias.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` Command syntax was incorrect, invalid option was used, or an internal error occurred.

`2` Device table could not be opened for reading.

FILES

`/etc/device.tab`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`devattr`(1M), `getdgrp`(1M), `putdev`(1M), `putdgrp`(1M), `attributes`(5)

NAME

getdgrp – lists device groups which contain devices that match criteria

SYNOPSIS

/usr/sbin/getdgrp [-ael] [*criteria*...] [*dgroup*...]

DESCRIPTION

`getdgrp` generates a list of device groups that contain devices matching the given criteria. The criteria is given in the form of expressions.

OPTIONS

The following options are supported:

`-a` Specifies that a device must match all criteria to be included in the list generated by this command. The option has no effect if no criteria are defined.

`-e` Specifies that the list of device groups on the command line should be *excluded* from the list generated by this command. Without the `-e` option the named device groups are *included* in the generated list. The flag has no effect if no devices are defined.

`-l` Specifies that all device groups (subject to the `-e` option and the *dgroup* list) should be listed even if they contain no valid device members. This option has no affect if *criteria* is specified on the command line.

OPERANDS

The following operands are supported:

criteria Defines criteria that a device must match before a device group to which it belongs can be included in the generated list. Specify *criteria* as an expression or a list of expressions which a device must meet for its group to be included in the list generated by `getdgrp`. If no criteria are given, all device groups are included in the list.

Devices must satisfy at least one of the criteria in the list. However, the `-a` option can be used to define that a "logical and" operation should be performed. Then, only those groups containing devices which match all of the criteria in a list will be included.

There are four possible expressions types which the criteria specified in the *criteria* argument may follow:

attribute=*value* Selects all device groups with a member whose attribute *attribute* is defined and is equal to *value*.

attribute`!=`*value* Selects all device groups with a member whose attribute *attribute* is defined and does not equal *value*.

attribute`:*` Selects all device groups with a member which has the attribute *attribute* defined.

attribute`!:*` — Selects all device groups with a member which does not have the attribute *attribute* defined.

See `putdev`(1M) for a complete listing and description of available attributes.

dgroup — Defines a set of device groups which should be included in or excluded from the generated list. Device groups that are defined and which contain devices matching the criteria are included.

If the `-e` option is used, this list defines a set of device groups to be excluded. When the `-e` option is used and *criteria* is also defined, the generated list will include device groups containing devices which match the criteria and are not in the command line list.

EXIT STATUS

The following exit values are returned:

`0` — Successful completion of the task.

`1` — Command syntax was incorrect, invalid option was used, or an internal error occurred.

`2` — Device table or device group table could not be opened for reading.

FILES

`/etc/device.tab`

`/etc/dgroup.tab`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`devattr`(1M), `getdev`(1M), `putdev`(1M), `putdgrp`(1M), `attributes`(5)

NAME

getent – get entries from administrative database

SYNOPSIS

`getent` *database* [*key*...]

DESCRIPTION

`getent` gets a list of entries from the administrative database specified by *database*. The information generally comes from one or more of the sources that are specified for the *database* in `/etc/nsswitch.conf`.

database is the name of the database to be examined. This can be `passwd`, `group`, `hosts`,`ipnodes`, `services`, `protocols`, `ethers`, `networks`, or `netmasks`. For each of these databases, `getent` uses the appropriate library routines described in `getpwnam`(3C), `getgrnam`(3C), `gethostbyaddr`(3NSL), `gethostbyname`(3NSL), `getipnodebyaddr`(3SOCKET), `getipnodebyname`(3SOCKET), `getservbyname`(3SOCKET), `getprotobyname`(3SOCKET), `ethers`(3SOCKET), and `getnetbyname`(3SOCKET), respectively.

Each *key* must be in a format appropriate for searching on the respective database. For example, it can be a *username* or *numeric-uid* for `passwd`; *hostname* or *IP address* for `hosts`; or *service*, *service/protocol*, *port*, or *port/proto* for `services`.

`getent` prints out the database entries that match each of the supplied keys, one per line, in the format of the matching administrative file: `passwd`(4), `group`(4), `hosts`(4), `ipnodes`(4),`services`(4), `protocols`(4), `ethers`(3SOCKET), `networks`(4), or `netmasks`(4). If no key is given, all entries returned by the corresponding enumeration library routine, for example, `getpwent()` or `gethostent()`, are printed. Enumeration is not supported on `ipnodes`.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` Command syntax was incorrect, an invalid option was used, or an internal error occurred.

`2` At least one of the specified entry names was not found in the database.

`3` There is no support for enumeration on this database.

FILES

`/etc/nsswitch.conf`	name service switch configuration file
`/etc/passwd`	password file
`/etc/group`	group file
`/etc/inet/hosts`	IPv4 host name database
`/etc/inet/ipnodes`	IPv4 and IPv6 host name database
`/etc/services`	Internet services and aliases
`/etc/protocols`	protocol name database
`/etc/ethers`	Ethernet address to hostname database or domain
`/etc/networks`	network name database

`/etc/netmasks`	network mask database

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`ethers`(3SOCKET), `getgrnam`(3C), `gethostbyaddr`(3NSL), `gethostbyname`(3NSL), `gethostent`(3NSL), `getipnodebyaddr`(3SOCKET), `getipnodebyname`(3SOCKET), `getnetbyname`(3SOCKET), `getprotobyname`(3SOCKET), `getpwnam`(3C), `getservbyname`(3SOCKET), `group`(4), `hosts`(4), `ipnodes`(4), `netmasks`(4), `networks`(4), `nsswitch.conf`(4), `passwd`(4), `protocols`(4), `services`(4), `attributes`(5)

NAME

gettable – get DoD Internet format host table from a host

SYNOPSIS

`/usr/sbin/gettable` *host*

DESCRIPTION

`gettable` is a simple program used to obtain the DoD Internet host table from a "hostname" server. The specified *host* is queried for the table. The table is placed in the file `hosts.txt`.

`gettable` operates by opening a TCP connection to the port indicated in the service specification for "hostname". A request is then made for all names and the resultant information is placed in the output file.

`gettable` is best used in conjunction with the `htable`(1M) program which converts the DoD Internet host table format to that used by the network library lookup routines.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`htable`(1M), `attributes`(5) Harrenstien, Ken, Mary Stahl, and Elizabeth Feinler, HOSTNAME *Server*, RFC 953, Network Information Center, SRI International, Menlo Park, California, October 1985.

NOTES

Should allow requests for only part of the database.

NAME

getty – set terminal type, modes, speed, and line discipline

SYNOPSIS

/usr/lib/saf/ttymon [-h] [-t *timeout*] *line* [*speed* [*type* [*linedisc*]]]

/usr/lib/saf/ttymon -c *file*

DESCRIPTION

`getty` sets terminal type, modes, speed, and line discipline. `getty` is a symbolic link to `/usr/lib/saf/ttymon`. It is included for compatibility with previous releases for the few applications that still call `getty` directly.

`getty` can only be executed by the super-user, (a process with the user ID `root`). Initially `getty` prints the login prompt, waits for the user's login name, and then invokes the `login` command. `getty` attempts to adapt the system to the terminal speed by using the options and arguments specified on the command line.

Without optional arguments, `getty` specifies the following: The *speed* of the interface is set to 300 baud, either parity is allowed, NEWLINE characters are converted to carriage return-line feed, and tab expansion is performed on the standard output. `getty` types the login prompt before reading the user's name a character at a time. If a null character (or framing error) is received, it is assumed to be the result of the user pressing the BREAK key. This will cause `getty` to attempt the next *speed* in the series. The series that `getty` tries is determined by what it finds in `/etc/ttydefs` .

OPTIONS

The following options are supported:

`-h` — If the `-h` flag is not set, a hangup will be forced by setting the speed to zero before setting the speed to the default or a specified speed.

`-t` *timeout* — Specifies that `getty` should exit if the open on the line succeeds and no one types anything in *timeout* seconds.

`-c` *file* — The `-c` option is no longer supported. Instead use `/usr/sbin/sttydefs -l` to list the contents of the `/etc/ttydefs` file and perform a validity check on the file.

OPERANDS

The following operands are supported:

line — The name of a TTY line in `/dev` to which `getty` is to attach itself. `getty` uses this string as the name of a file in the `/dev` directory to open for reading and writing.

speed — The *speed* argument is a label to a speed and TTY definition in the file `/etc/ttydefs`. This definition tells `getty` at what speed to run initially, what the initial TTY settings are, and what speed to try next, (should the user press the BREAK key to indicate that the speed is inappropriate). The default *speed* is 300 baud.

type and *linedisc* — These options are obsolete and will be ignored.

FILES `/etc/ttydefs`

ATTRIBUTES See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsr

SEE ALSO `ct`(1C), `login`(1), `sttydefs`(1M), `ttymon`(1M), `ioctl`(2), `attributes`(5), `tty`(7D)

NAME

getvol – verifies device accessibility

SYNOPSIS

/usr/bin/getvol `-n [-l` *label*`]` *device*

/usr/bin/getvol `[-f | -F] [-ow] [-l` *label* `| -x` *label*`]` *device*

DESCRIPTION

`getvol` verifies that the specified device is accessible and that a volume of the appropriate medium has been inserted. The command is interactive and displays instructional prompts, describes errors, and shows required label information.

OPTIONS

The following options are supported:

`-n` Runs the command in non-interactive mode. The volume is assumed to be inserted upon command invocation.

`-l` *label* Specifies that the label *label* must exist on the inserted volume (can be overridden by the `-o` option).

`-f` Formats the volume after insertion, using the format command defined for this device in the device table.

`-F` Formats the volume after insertion and places a file system on the device. Also uses the format command defined for this device in the device table.

`-o` Allows the administrator to override a label check.

`-w` Allows administrator to write a new label on the device. User is prompted to supply the label text. This option is ineffective if the `-n` option is enabled.

`-x` *label* Specifies that the label *label* must exist on the device. This option should be used in place of the `-l` option when the label can only be verified by visual means. Use of the option causes a message to be displayed asking the administrator to visually verify that the label is indeed *label*.

OPERANDS

The following operands are supported:

device Specifies the device to be verified for accessibility.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` Command syntax was incorrect, invalid option was used, or an internal error occurred.

`3` Device table could not be opened for reading.

FILES

`/etc/device.tab`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`attributes`(5)

NOTES

This command uses the device table to determine the characteristics of the device when performing the volume label checking.

NAME

gkadmin – Kerberos database administration GUI, SEAM Administration Tool

SYNOPSIS

/usr/sbin/gkadmin

DESCRIPTION

`gkadmin` is an interactive graphical user interface (GUI) that enables you to maintain Kerberos principals and policies. `gkadmin` provides much the same functionality as the `kadmin`(1M) command.

`gkadmin` does not support the management of keytabs. You must use `kadmin` for keytabs management. `gkadmin` uses Kerberos authentication and an encrypted RPC to operate securely from anywhere on the network.

When `gkadmin` is invoked, the login window is populated with default values. For the principal name, `gkadmin` determines your user name from the `USER` environment variable. It appends `/admin` to the name (*username*`/admin`) to create a default user instance in the same manner as `kadmin`. It also selects appropriate defaults for realm and master KDC (*admin_server*) from the `/etc/krb5/krb5.conf` file.

You can change these defaults on the login window. When you enter your password, a session is started with `kadmind`. Operations performed are subject to permissions that are granted or denied to the chosen user instance by the Kerberos ACL file. See `kadm5.acl`(4).

After the session is started, a tabbed folder is displayed that contains a principal list and a policy list. The functionality is mainly the same as `kadmin`, with addition, deletion, and modification of principal and policy data available.

In addition, `gkadmin` provides the following features:

- New principal or policy records can be added either from default values or from the settings of an existing principal.
- A comment field is available for principals.
- Default values are saved in `$HOME/.gkadmin`.
- A logout option permits you to log back in as another user instance without exiting the tool.
- Principal and policy lists and attributes can be printed or saved to a file.
- Online context-sensitive help and general help is available in the `Help` menu.

FILES

`/etc/krb5/krb5.conf`	Kerberos configuration information on a Kerberos client. Used to search for default realm and master KDC (`admin_server`), including a port number for the master KDC.
`$HOME/.gkadmin`	Default parameters used to initialize new principals created during the session.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWkdcu

SEE ALSO

`kpasswd`(1), `kadmin`(1M), `kadmind`(1M), `kadmin.local`(1M), `kdb5_util`(1M), `kadm5.acl`(4), `kdc.conf`(4), `krb5.conf`(4), `attributes`(5), `SEAM`(5)

DIAGNOSTICS

The `gkadmin` interface is currently incompatible with the MIT `kadmind` daemon interface, so you cannot use this interface to administer an MIT-based Kerberos database. However, SEAM-based Kerberos clients can still use an MIT-based KDC.

NAME

groupadd – add (create) a new group definition on the system

SYNOPSIS

/usr/sbin/groupadd [-g *gid* [-o]] *group*

DESCRIPTION

The groupadd command creates a new group definition on the system by adding the appropriate entry to the /etc/group file.

OPTIONS

The following options are supported:

-g *gid* Assigns the group id *gid* for the new group. This group id must be a non-negative decimal integer below MAXUID as defined in /usr/include/sys/param.h. The group ID defaults to the next available (unique) number above the highest number currently assigned. For example, if groups 100, 105, and 200 are assigned as groups, the next default group number will be 201. (Group IDs from 0–99 are reserved by SunOS for future applications.)

-o Allows the *gid* to be duplicated (non-unique).

OPERANDS

The following operands are supported:

group A string consisting of characters from the set of lower case alphabetic characters and numeric characters. A warning message will be written if the string exceeds MAXGLEN, which is usually set at eight characters. The *group* field must contain at least one character; it accepts lower case or numeric characters or a combination of both, and must not contain a colon (:) or NEWLINE.

EXIT STATUS

The following exit values are returned:

0 Successful completion.

2 Invalid command syntax. A usage message for the groupadd command is displayed.

3 An invalid argument was provided to an option.

4 The *gid* is not unique (when -o option is not used).

9 The *group* is not unique.

10 The /etc/group file cannot be updated.

FILES

/etc/group

/usr/include/userdefs.h

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO `users`(1B), `groupdel`(1M), `groupmod`(1M), `grpck`(1M), `logins`(1M), `pwck`(1M), `useradd`(1M), `userdel`(1M), `usermod`(1M), `group`(4), `attributes`(5)

NOTES `groupadd` only adds a group definition to the local system. If a network name service such as NIS or NIS+ is being used to supplement the local `/etc/group` file with additional entries, `groupadd` cannot change information supplied by the network name service. However, `groupadd` will verify the uniqueness of group name and group ID against the external name service.

NAME

groupdel – delete a group definition from the system

SYNOPSIS

/usr/sbin/groupdel *group*

DESCRIPTION

The `groupdel` utility deletes a group definition from the system. It deletes the appropriate entry from the `/etc/group` file.

OPERANDS

group An existing group name to be deleted.

EXIT STATUS

The following exit values are returned:

`0` Success.

`2` Invalid command syntax. A usage message for the `groupdel` command is displayed.

`6` *group* does not exist.

`10` Cannot update the `/etc/group` file.

FILES

`/etc/group` system file containing group definitions

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`users`(1B), `groupadd`(1M), `groupmod`(1M), `logins`(1M), `useradd`(1M), `userdel`(1M), `usermod`(1M), `attributes`(5)

NOTES

The `groupdel` utility only deletes a group definition that is in the local `/etc/group` file. If a network nameservice such as NIS or NIS+ is being used to supplement the local `/etc/group` file with additional entries, `groupdel` cannot change information supplied by the network nameservice.

NAME

groupmod – modify a group definition on the system

SYNOPSIS

/usr/sbin/groupmod [-g *gid* [-o]] [-n *name*] *group*

DESCRIPTION

The groupmod command modifies the definition of the specified group by modifying the appropriate entry in the /etc/group file.

OPTIONS

The following options are supported:

-g *gid* Specify the new group ID for the group. This group ID must be a non-negative decimal integer less than MAXUID, as defined in <param.h>. The group ID defaults to the next available (unique) number above 99. (Group IDs from 0-99 are reserved by SunOS for future applications.)

-n *name* Specify the new name for the group. The *name* argument is a string of no more than eight bytes consisting of characters from the set of lower case alphabetic characters and numeric characters. A warning message will be written if these restrictions are not met. A future Solaris release may refuse to accept group fields that do not meet these requirements. The *name* argument must contain at least one character and must not include a colon (:) or NEWLINE (\n).

-o Allow the *gid* to be duplicated (non-unique).

OPERANDS

The following operands are supported:

group An existing group name to be modified.

EXIT STATUS

The groupmod utility exits with one of the following values:

0 Success.

2 Invalid command syntax. A usage message for the groupmod command is displayed.

3 An invalid argument was provided to an option.

4 *gid* is not unique (when the -o option is not used).

6 *group* does not exist.

9 *name* already exists as a group name.

10 Cannot update the /etc/group file.

FILES

/etc/group group file

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO `users`(1B), `groupadd`(1M), `groupdel`(1M), `logins`(1M), `useradd`(1M), `userdel`(1M), `usermod`(1M), `group`(4), `attributes`(5)

NOTES The `groupmod` utility only modifies group definitions in the `/etc/group` file. If a network name service such as NIS or NIS+ is being used to supplement the local `/etc/group` file with additional entries, `groupmod` cannot change information supplied by the network name service. The `groupmod` utility will, however, verify the uniqueness of group name and group ID against the external name service.

NAME

growfs – non-destructively expand a UFS file system

SYNOPSIS

/usr/sbin/growfs [-M *mount-point*] [*newfs-options*] [*raw-device*]

DESCRIPTION

`growfs` non-destructively expands a mounted or unmounted UNIX file system (UFS) to the size of the file system's slice(s).

Typically, disk space is expanded by first adding a slice to a metadevice, then running the `growfs` command. When adding space to a mirror, you expand each submirror before expanding the file system. On a trans metadevice, the master device is expanded, not the trans metadevice. Then the `growfs` command is run on the trans metadevice. (You can add space to a logging device, but you do not need to run the `growfs` command. The new space is automatically recognized.)

`growfs` will ''write-lock'' (see `lockfs`(1M)) a mounted file system when expanding. The length of time the file system is write-locked can be shortened by expanding the file system in stages. For instance, to expand a 1 Gbyte file system to 2 Gbytes, the file system can be grown in 16 Mbyte stages using the `-s` option to specify the total size of the new file system at each stage. The argument for `-s` is the number of sectors, and must be a multiple of the cylinder size. Note: The file system cannot be grown if a cylinder size of less than 2 is specified. Refer to the `newfs`(1M) man page for information on the options available when growing a file system.

`growfs` displays the same information as `mkfs` during the expansion of the file system.

If `growfs` is aborted, recover any lost free space by unmounting the file system and running the `fsck` command, or run the `growfs` command again.

OPTIONS

Root privileges are required for all of the following options.

`-M` *mount-point*
: The file system to be expanded is mounted on *mount-point*. File system locking (`lockfs`) will be used.

newfs-options
: The options are documented in the `newfs` man page.

raw-device
: Specifies the name of a raw metadevice or raw special device, residing in `/dev/md/rdsk`, or `/dev/rdsk`, respectively, including the disk slice, where you want the file system to be grown.

EXAMPLES

EXAMPLE 1 Expanding nonmetadevice slice for `/export` file system

The following example expands a nonmetadevice slice for the `/export` file system. In this example, the existing slice, `/dev/dsk/c1t0d0s3`, is converted to a metadevice so additional slices can be concatenated.

```
# metainit -f d8 2 1 c1t0d0s3 1 c2t0d0s3
# umount /export
```

EXAMPLE 2 Associate `/export` with new metadevice

Edit the `/etc/vfstab` file to change the entry for `/export` to the newly defined metadevice, `d8`.

```
# mount /export
# growfs -M /export /dev/md/rdsk/d8
```

The first example starts by running the `metainit` command with the `-f` option to force the creation of a new concatenated metadevice `d8`, which consists of the existing slice `/dev/dsk/c1t0d0s3` and a new slice `/dev/dsk/c2t0d0s3`. Next, the file system on `/export` must be unmounted. The `/etc/vfstab` file is edited to change the entry for `/export` to the newly defined metadevice name, rather than the slice name. After the file system is remounted, the `growfs` command is run to expand the file system. The file system will span the entire metadevice when `growfs` completes. The `-M` option enables the `growfs` command to expand a mounted file system. During the expansion, write access for `/export` is suspended until `growfs` unlocks the file system. Read access is not affected, though access times are not kept when the lock is in effect.

EXAMPLE 3 Dynamic Expansion of `/export` file system

The following example picks up from the previous one. Here, the `/export` file system mounted on metadevice `d8` is dynamically expanded.

```
# metattach d8 c0t1d0s2
# growfs -M /export /dev/md/rdsk/d8
```

This example begins by using the `metattach` command to dynamically concatenate a new slice, `/dev/dsk/c0t1d0s2`, to the end of an existing metadevice, `d8`. Next, the `growfs` command specifies that the mount-point is `/export` and that it is to be expanded onto the raw metadevice `/dev/md/rdsk/d8`. The file system will span the entire metadevice when `growfs` completes. During the expansion, write access for `/export` is suspended until `growfs` unlocks the file system. Read access is not affected, though access times are not kept when the lock is in effect.

EXAMPLE 4 Expanding mounted file system to existing mirror

The following example expands a mounted file system `/files`, to an existing mirror, `d80`, which contains two submirrors, `d9` and `d10`.

```
# metattach d9 c0t2d0s5
# metattach d10 c0t3d0s5
# growfs -M /files /dev/md/rdsk/d80
```

In this example, the `metattach` command dynamically concatenates the new slices to each submirror. The `metattach` command must be run for each submirror. The mirror will automatically grow when the last submirror is dynamically concatenated. The mirror will grow to the size of the smallest submirror. The `growfs` command then expands the file system. The `growfs` command specifies that the mount-point is `/files` and that it is to be expanded onto the raw metadevice `/dev/md/rdsk/d80`.

EXAMPLE 4 Expanding mounted file system to existing mirror *(Continued)*

The file system will span the entire mirror when the `growfs` command completes. During the expansion, write access for the file system is suspended until `growfs` unlocks the file system. Read access is not affected, though access times are not kept when the lock is in effect.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdu

SEE ALSO

`fsck`(1M), `lockfs`(1M), `mkfs`(1M), `metattach`(1M), `newfs`(1M), `attributes`(5)

Solaris Volume Manager Administration Guide

LIMITATIONS

Only UFS file systems (either mounted or unmounted) can be expanded using the `growfs` command. Once a file system is expanded, it cannot be decreased in size. The following conditions prevent you from expanding file systems: When `acct` is activated and the accounting file is on the target device. When C2 security is activated and the logging file is on the target file system. When there is a local `swap` file in the target file system. When the file system is root (`/`), `/usr`, or `swap`.

NAME

gsscred – add, remove and list gsscred table entries

SYNOPSIS

gsscred [-n *user* [-o *oid*] [-u *uid*]] [-c *comment*] -m *mech* -a

gsscred [-n *user* [-o *oid*]] [-u *uid*] [-m *mech*] -r

gsscred [-n *user* [-o *oid*]] [-u *uid*] [-m *mech*] -l

DESCRIPTION

The `gsscred` utility is used to create and maintain a mapping between a security principal name and a local UNIX *uid*. The format of the user name is assumed to be `GSS_C_NT_USER_NAME`. You can use the `-o` option to specify the object identifier of the *name* type. The `OID` must be specified in dot-separated notation, for example: `1.2.3.45464.3.1`

The `gsscred` table is used on server machines to lookup the *uid* of incoming clients connected using `RPCSEC_GSS`.

When adding users, if no *user* name is specified, an entry is created in the table for each user from the `passwd` table. If no *comment* is specified, the `gsscred` utility inserts a comment that specifies the user name as an ASCII string and the GSS-APIsecurity mechanism that applies to it. The security mechanism will be in string representation as defined in the `/etc/gss/mech` file.

The parameters are interpreted the same way by the `gsscred` utility to delete users as they are to create users. At least one of the following options must be specified: `-n`, `-u`, or `-m`. If no security mechanism is specified, then all entries will be deleted for the user identified by either the *uid* or *user* name. If only the security mechanism is specified, then all *user* entries for that security mechanism will be deleted.

Again, the parameters are interpreted the same way by the `gsscred` utility to search for users as they are to create users. If no options are specified, then the entire table is returned. If the *user* name or *uid* is specified, then all entries for that *user* are returned. If a security mechanism is specified, then all *user* entries for that security mechanism are returned.

OPTIONS

`-a`	Add a table entry.
`-c` *comment*	Insert comment about this table entry.
`-l`	Search table for entry.
`-m` *mech*	Specify the mechanism for which this name is to be translated.
`-n` *user*	Specify the optional principal name.
`-o` *oid*	Specify the OID indicating the name type of the user.
`-r`	Remove the entry from the table.
`-u` *uid*	Specify the *uid* for the *user* if the *user* is not local.

EXAMPLES

EXAMPLE 1 Creating a `gsscred` Table for the Kerberos v5 Security Mechanism

The following shows how to create a `gsscred` table for the kerberos v5 security mechanism. `gsscred` obtains *user* names and *uid*'s from the `passwd` table to populate the table.

```
example% gsscred -m kerberos_v5 -a
```

EXAMPLE 2 Adding an Entry for `root/host1` for the Kerberos v5 Security Mechanism

The following shows how to add an entry for `root/host1` with a specified *uid* of `0` for the kerberos v5 security mechanism.

```
example% gsscred -m kerberos_v5 -n root/host1 -u 0 -a
```

EXAMPLE 3 Listing All User Mappings for the Kerberos v5 Security Mechanism

The following lists all user mappings for the kerberos v5 security mechanism.

```
example% gsscred -m kerberos_v5 -l
```

EXAMPLE 4 Listing All Mappings for All Security Mechanism for a Specified User

The following lists all mappings for all security mechanisms for the user bsimpson..

```
example% gsscred -n bsimpson -l
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWgss

SEE ALSO

`gssd`(1m),`attributes`(5)

NAME

gssd – generates and validates GSS-API tokens for kernel RPC

SYNOPSIS

```
/usr/lib/gss/gssd
```

DESCRIPTION

`gssd` is the user mode daemon that operates between the kernel rpc and the Generic Security Service Application Program Interface (GSS-API) to generate and validate GSS-API security tokens. In addition, `gssd` maps the GSS-API principal names to the local user and group ids. By default, all groups that the requested user belongs to will be included in the grouplist credential. `gssd` is invoked by the Internet daemon `inetd`(1m) the first time that the kernel RPC requests GSS-API services.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWgssk

SEE ALSO

`gsscred`(1m),`attributes`(5)

RFC 2078

NAME

halt, poweroff – stop the processor

SYNOPSIS

/usr/sbin/halt [-dlnqy]

/usr/sbin/poweroff [-dlnqy]

DESCRIPTION

The `halt` and `poweroff` utilities write any pending information to the disks and then stop the processor. The `poweroff` utility will have the machine remove power, if possible.

The `halt` and `poweroff` utilities normally log the system shutdown to the system log daemon, `syslogd`(1M), and place a shutdown record in the login accounting file `/var/adm/wtmpx`. These actions are inhibited if the `-n` or `-q` options are present.

OPTIONS

The following options are supported:

`-d` Force a system crash dump before rebooting. See `dumpadm`(1M) for information on configuring system crash dumps.

`-l` Suppress sending a message to the system log daemon, `syslogd`(1M), about who executed `halt`.

`-n` Prevent the `sync`(1M) before stopping.

`-q` Quick halt. No graceful shutdown is attempted.

`-y` Halt the system, even from a dialup terminal.

FILES

`/var/adm/wtmpx` History of user access and administration information.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`dumpadm`(1M), `init`(1M), `reboot`(1M), `shutdown`(1M), `sync`(1M), `syslogd`(1M), `attributes`(5)

NOTES

The `halt` utility does not execute the scripts in `/etc/rc`*<num>*`.d` or execute shutdown actions in `inittab`(4). To ensure a complete shutdown of system services, use `shutdown`(1M) or `init`(1M) to reboot a Solaris system.

The `poweroff` utility is equivalent to `init 5`.

NAME

hostconfig – configure a system's host parameters

SYNOPSIS

```
/usr/sbin/hostconfig -p protocol [-d] [ -h] [-n] [-v] [-i interface]
     [-f hostname]
```

DESCRIPTION

The `hostconfig` program uses a network protocol to acquire a machine's *host parameters* and set these parameters on the system.

The program selects which protocol to use based on the argument to the required `-p` flag. Different protocols may set different host parameters. Currently, only one protocol (`bootparams`) is defined.

OPTIONS

The following options are supported:

`-d`	Enable `debug` output.
`-f` *hostname*	Run the protocol as if this machine were named `hostname`.
`-h`	Echo the received `hostname` to `stdout`, rather than setting `hostname` using the system name directly.
`-i` *interface*	Use only the named network interface to run the protocol.
`-n`	Run the network protocol, but do not set the acquired parameters into the system.
`-p` *protocol*	Run `hostconfig` using *protocol*. Currently, only one protocol (`bootparams`) is available. This option is required. Specifying the `-p bootparams` option uses the `whoami` call of the RPC `bootparams` protocol. This sets the system's `hostname`, `domainname`, and default IP router parameters.
`-v`	Enable verbose output.

EXAMPLES

EXAMPLE 1 Configuring Host Parameters with Verbose Output

The following command configures a machine's host parameters using the `whoami` call of the RPC `bootparams` protocol with a verbose output.

```
example# hostconfig -p bootparams -v
```

EXAMPLE 2 Displaying Host Parameters

The following command displays the parameters that would be set using the `whoami` call of the RPC `bootparams` protocol.

```
example# hostconfig -p bootparams -n -v
```

EXAMPLE 3 Configuring Host Parameters Less the System Name

The following command configures a machine's host parameters, less the system name, using the `whoami` call of the RPC `bootparams` protocol.

```
example# hostconfig='hostconfig -p bootparams -h'
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`hostname`(1), `domainname`(1M), `route`(1M), `attributes`(5)

NAME

htable – convert DoD Internet format host table

SYNOPSIS

/usr/sbin/htable *filename*

DESCRIPTION

`htable` converts a host table in the format specified by RFC 952 to the format used by the network library routines. Three files are created as a result of running `htable`: `hosts`, `networks`, and `gateways`. The `hosts` file is used by the `gethostbyname`(3NSL) routines in mapping host names to addresses. The `networks` file is used by the `getnetbyname`(3SOCKET) routines in mapping network names to numbers. The `gateways` file is used by the routing daemon to identify "passive" Internet gateways.

If any of the files `localhosts`, `localnetworks`, or `localgateways` are present in the current directory, the file's contents is prepended to the output file without interpretation. This allows sites to maintain local aliases and entries which are not normally present in the master database.

`htable` is best used in conjunction with the `gettable`(1M) program which retrieves the DoD Internet host table from a host.

FILES

`localhosts`

`localnetworks`

`localgateways`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`gettable`(1M), `gethostbyname`(3NSL), `getnetbyname`(3SOCKET), `attributes`(5) Harrenstien, Ken, Mary Stahl, and Elizabeth Feinler, *DoD Internet Host Table Specification*, RFC 952, Network Information Center, SRI International, Menlo Park, California, October 1985.

NOTES

`htable` does not properly calculate the `gateways` file.

NAME

id – return user identity

SYNOPSIS

/usr/bin/id [-p] [*user*]

/usr/bin/id -a [-p] [*user*]

/usr/xpg4/bin/id [-p] [*user*]

/usr/xpg4/bin/id -G [-n] [*user*]

/usr/xpg4/bin/id -g [-nr] [*user*]

/usr/xpg4/bin/id -u [-nr] [*user*]

DESCRIPTION

If no *user* operand is provided, the `id` utility writes the user and group IDs and the corresponding user and group names of the invoking process to standard output. If the effective and real IDs do not match, both are written. If multiple groups are supported by the underlying system, `/usr/xpg4/bin/id` also writes the supplementary group affiliations of the invoking process.

If a *user* operand is provided and the process has the appropriate privileges, the user and group IDs of the selected user are written. In this case, effective IDs are assumed to be identical to real IDs. If the selected user has more than one allowable group membership listed in the group database, `/usr/xpg4/bin/id` writes them in the same manner as the supplementary groups described in the preceding paragraph.

Formats

The following formats are used when the `LC_MESSAGES` locale category specifies the "C" locale. In other locales, the strings `uid`, `gid`, `euid`, `egid`, and `groups` may be replaced with more appropriate strings corresponding to the locale.

`"uid=%u(%s) gid=%u(%s)\n"` <*real user ID*>, <*user-name*>, <*real group ID*>, <*group-name*>

If the effective and real user IDs do not match, the following are inserted immediately before the `\n` character in the previous format:

`" euid=%u(%s)"`

with the following arguments added at the end of the argument list:

<*effective user ID*>, <*effective user-name*>

If the effective and real group IDs do not match, the following is inserted directly before the `\n` character in the format string (and after any addition resulting from the effective and real user IDs not matching):

`" egid=%u(%s)"`

with the following arguments added at the end of the argument list:

<*effectivegroup-ID*>, <*effectivegroupname*>

If the process has supplementary group affiliations or the selected user is allowed to belong to multiple groups, the first is added directly before the NEWLINE character in the format string:

```
" groups=%u(%s)"
```

with the following arguments added at the end of the argument list:

<*supplementary group ID*>, <*supplementary group name*>

and the necessary number of the following added after that for any remaining supplementary group IDs:

```
",%u(%s)"
```

and the necessary number of the following arguments added at the end of the argument list:

<*supplementary group ID*>, <*supplementary group name*>

If any of the user ID, group ID, effective user ID, effective group ID or supplementary/multiple group IDs cannot be mapped by the system into printable user or group names, the corresponding (`%s`) and name argument is omitted from the corresponding format string.

When any of the options are specified, the output format is as described under `OPTIONS`.

OPTIONS

The following option is supported by both `/usr/bin/id` and `/usr/xpg4/bin/id`. For `/usr/xpg4/bin/id`, `-p` is invalid if specified with any of the `-G`, `-g`, or `-u` options.

`-p` Reports additionally the current project membership of the invoking process. The project is reported using the format:

`"projid=%u(%s)"`

which is inserted prior to the `\n` character of the default format described in the `Formats` section. The arguments

<*project ID*>, <*project name*>

are appended to the end of the argument list. If the project ID cannot be mapped by the system into a printable project name, the corresponding `(%s)` and name argument is omitted from the corresponding format string.

/usr/bin/id

The following option is supported for `/usr/bin/id` only:

`-a` Reports user name, user ID and all the groups to which the user belongs.

/usr/xpg4/bin/id

The following options are supported for `/usr/xpg4/bin/id` only:

`-G` Output all different group IDs (effective, real and supplementary) only, using the format `"%u\n"`. If there is more than one distinct group affiliation, output each such affiliation, using the format `" %u"`, before the NEWLINE character is output.

`-g` Output only the effective group ID, using the format `"%u\n"`.

`-n` Output the name in the format `"%s"` instead of the numeric ID using the format `"%u"`.

`-r` Output the real ID instead of the effective ID.

`-u` Output only the effective user ID, using the format `"%u\n"`.

OPERANDS

The following operand is supported:

user The user (login) name for which information is to be written.

ENVIRONMENT VARIABLES

See `environ`(5) for descriptions of the following environment variables that affect the execution of `id`: `LC_CTYPE`, `LC_MESSAGES`, and `NLSPATH`.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

/usr/bin/id

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu
	SUNWcar

/usr/xpg4/bin/id

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWxcu4

SEE ALSO

`fold`(1), `logname`(1), `who`(1), `getgid`(2), `getgroups`(2), `getprojid`(2), `getuid`(2), `attributes`(5), `environ`(5), `XPG4`(5)

NOTES

Output produced by the `-G` option and by the default case could potentially produce very long lines on systems that support large numbers of supplementary groups.

NAME

idsconfig – prepare an iPlanet Directory Server (iDS) to be populated with data and serve LDAP clients

SYNOPSIS

/usr/lib/ldap/idsconfig [-v] [-i *input_configfile*] [-o *output_configfile*]

DESCRIPTION

Use the idsconfig tool to set up an iPlanet Directory Server (iDS). You can specify the input configuration file with the -i option on the command line. Alternatively, the tool will prompt the user for configuration information. The input configuration file is created by idsconfig with the -o option on a previous run.

The first time a server is set up, the user is prompted for all the required information. Future installations on that machine can use the configuration file previously generated by idsconfig using the -o option.

The output configuration file contains the directory administrator's password in clear text. Thus, if you are creating an output configuration file, take appropriate security precautions.

You should back up the directory server's configuration and data prior to running this command.

OPTIONS

-i *input_configfile*	Specify the file name for idsconfig to use as a configuration file. This file will be read by idsconfig, and the values in the file will be used to configure the server. Do not manually edit *input_configfile*. The *input_configfile* is only partially validated, as idsconfig assumes that the file was created by a previous invocation of the command.
-o *output_configfile*	Create a configuration file.
-v	Verbose output.

OPERANDS

The following operands are supported:

input_configfile	Name of configuration file for idsconfig to use.
output_configfile	Configuration file created by idsconfig.

EXAMPLES

EXAMPLE 1 Prompting the User for Input

In the following example, the user is prompted for information to set up iDS.

```
example# idsconfig
```

EXAMPLE 2 Creating an Output Configuration File

In the following example, the user is prompted for information to set up iDS, and an output configuration file, config.1, is created when completed.

```
example# idsconfig -o config.1
```

EXAMPLE 3 Setting up iDS Using the Specified Configuration File

In the following example, iDS is set up by using the values specified in the configuration file, `config.1`. The verbose mode is specified, so detailed information will print to the screen.

```
example# idsconfig -v -i config.1
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu
Interface Stability	Evolving

SEE ALSO

`ldap`(1), `ldapadd`(1), `ldapdelete`(1), `ldaplist`(1), `ldapmodify`(1), `ldapmodrdn`(1), `ldapsearch`(1), `ldap_cachemgr`(1M), `ldapaddent`(1M), `ldapclient`(1M), `suninstall`(1M), `resolv.conf`(4), `attributes`(5)

NAME

ifconfig – configure network interface parameters

SYNOPSIS

/sbin/ifconfig *interface* [*address_family*] [*address* [*/prefix_length*] [*dest_address*]] [addif *address* [*/prefix_length*]] [removeif *address* [*/prefix_length*]] [arp | -arp] [auth_algs *authentication algorithm*] [encr_algs *encryption algorithm*] [encr_auth_algs *authentication algorithm*] [auto-revarp] [broadcast *address*] [deprecated | -deprecated] [destination *dest_address*] [[failover] | [-failover]] [group [[*name*] | ""]] [index {*if_index*}] [metric *n*] [modlist] [modinsert *mod_name@pos*] [modremove *mod_name@pos*] [mtu *n*] [netmask *mask*] [plumb] [unplumb] [private | -private] [nud | -nud] [set [*address*] [*/netmask*]] [[standby] | [-standby]] [subnet *subnet_address*] [tdst *tunnel_dest_address*] [token *address/prefix_length*] [tsrc *tunnel_src_address*] [trailers | -trailers] [up] [down] [xmit | -xmit]

/usr/sbin/ifconfig *interface* [*address_family*] [*address* [*/prefix_length*] [*dest_address*]] [addif *address* [*/prefix_length*]] [removeif *address* [*/prefix_length*]] [arp | -arp] [auth_algs *authentication algorithm*] [encr_algs *encryption algorithm*] [encr_auth_algs *authentication algorithm*] [auto-revarp] [broadcast *address*] [deprecated | -deprecated] [destination *dest_address*] [[failover] | [-failover]] [group [[*name*] | ""]] [index {*if_index*}] [metric *n*] [modlist] [modinsert *mod_name@pos*] [modremove *mod_name@pos*] [mtu *n*] [netmask *mask*] [plumb] [unplumb] [private | -private] [nud | -nud] [set [*address*] [*/netmask*]] [[standby] | [-standby]] [subnet *subnet_address*] [tdst *tunnel_dest_address*] [token *address/prefix_length*] [tsrc *tunnel_src_address*] [trailers | -trailers] [up] [down] [xmit | -xmit]

/sbin/ifconfig *interface* {auto-dhcp | dhcp} [primary] [wait *seconds*] drop | extend | inform | ping | release | start | status

/usr/sbin/ifconfig *interface* {auto-dhcp | dhcp} [primary] [wait *seconds*] drop | extend | inform | ping | release | start | status

DESCRIPTION

The command ifconfig is used to assign an address to a network interface and to configure network interface parameters. The ifconfig command must be used at boot time to define the network address of each interface present on a machine; it may also be used at a later time to redefine an interface's address or other operating parameters. If no option is specified, ifconfig displays the current configuration for a network interface. If an address family is specified, ifconfig reports only the details specific to that address family. Only the superuser may modify the configuration of a network interface. Options appearing within braces ({ }) indicate that one of the options must be specified.

The two versions of ifconfig, /sbin/ifconfig and /usr/sbin/ifconfig, behave differently with respect to name services. The order in which names are looked up by /sbin/ifconfig when the system is booting is fixed and cannot be changed.

In contrast, changing `/etc/nsswitch.conf` may affect the behavior of `/usr/sbin/ifconfig`. The system administrator may configure the source and lookup order in the tables by means of the name service switch. See `nsswitch.conf`(4) for more information.

DHCP Configuration

The third and fourth forms of this command are used to control the Dynamic Host Configuration Protocol ("DHCP") configuring of the interface. DHCP is only available on interfaces for which the address family is `inet`. In this mode, `ifconfig` is used to control operation of `dhcpagent`(1M), the DHCP client daemon. Once an interface is placed under DHCP control by using the `start` operand, `ifconfig` should not, in normal operation, be used to modify the address or characteristics of the interface. If the address of an interface under DHCP is changed, `dhcpagent` will remove the interface from its control.

OPTIONS

The following options are supported:

`addif` *address*
Create the next unused logical interface on the specified physical interface.

`arp`
Enable the use of the Address Resolution Protocol ("ARP") in mapping between network level addresses and link level addresses (default). This is currently implemented for mapping between IPv4 addresses and 10Mb/s Ethernet addresses.

`-arp`
Disable the use of the ARP.

auth_algs *authentication algorithm*
For a tunnel, enable IPsec AH with the authentication algorithm specified. The algorithm can be either a number or an algorithm name, including *any* to express no preference in algorithm. All IPsec tunnel properties must be specified on the same command line. To disable tunnel security, specify an `auth_alg` of *none*.

auto-dhcp
Use DHCP to automatically acquire an address for this interface. This option has a completely equivalent alias called `dhcp`.

primary	Defines the interface as the `primary`. The interface is defined as the preferred one for the delivery of client-wide configuration data. Only one interface can be the primary at any given time. If another interface is subsequently selected as the primary, it replaces the previous one. Nominating an interface as the primary one will not have much significance once the client work station has booted, as many applications will already have started and been configured with data read from the previous primary interface.
wait *seconds*	The `ifconfig` command will wait until the operation either completes or for the interval

specified, whichever is the sooner. If no wait interval is given, and the operation is one that cannot complete immediately, `ifconfig` will wait 30 seconds for the requested operation to complete. The symbolic value `forever` may be used as well, with obvious meaning.

drop	Remove the specified interface from DHCP control. Additionally, set the IP address to zero and mark the interface as “down”.
extend	Attempt to extend the lease on the interface’s IPv4 address. This is not required, as the agent will automatically extend the lease well before it expires.
inform	Obtain network configuration parameters from DHCP without obtaining a lease on an IP address. This is useful in situations where an IP address is obtained through mechanisms other than DHCP.
ping	Check whether the interface given is under DHCP control, which means that the interface is managed by the DHCP agent and is working properly. An exit status of `0` means success. This subcommand has no meaning when the named interface represents more than one interface.
release	Relinquish the IPv4 address on the interface, and mark the interface as “down.”
start	Start DHCP on the interface.
status	Display the DHCP configuration status of the interface.

`auto-revarp`
Use the Reverse Address Resolution Protocol (“RARP”) to automatically acquire an address for this interface.

`broadcast` *address*
For IPv4 only. Specify the address to use to represent broadcasts to the network. The default broadcast address is the address with a host part of all `1`’s. A "+" (plus sign) given for the broadcast value causes the broadcast address to be reset to a default appropriate for the (possibly new) address and netmask. The arguments of `ifconfig` are interpreted left to right. Therefore

```
example% ifconfig -a netmask + broadcast +
```

and

```
example% ifconfig -a broadcast + netmask +
```

may result in different values being assigned for the broadcast addresses of the interfaces.

`deprecated`
Marks the address as a deprecated address. Addresses marked as deprecated will not be used as source address for outbound packets unless either there are no other addresses available on this interface or the application has bound to this address explicitly. The status display shows `DEPRECATED` as part of flags.

`-deprecated`
Marks the address as not deprecated.

`destination` *dest_address*
Set the destination address for a point-to point interface.

`dhcp`
This option is an alias for option `auto-dhcp`

`down`
Mark an interface "down". When an interface is marked "down", the system does not attempt to transmit messages through that interface. If possible, the interface is reset to disable reception as well. This action does not automatically disable routes using the interface.

`encr_auth_algs` *authentication algorithm*
For a tunnel, enable IPsec ESP with the authentication algorithm specified. It can be either a number or an algorithm name, including `any` or `none`, to indicate no algorithm preference. If an ESP encryption algorithm is specified but the authentication algorithm is not, the default value for the ESP authentication algorithm will be `any`.

`encr_algs` *encryption algorithm*
For a tunnel, enable IPsec ESP with the encryption algorithm specified. It can be either a number or an algorithm name. Note that all IPsec tunnel properties must be specified on the same command line. To disable tunnel security, specify the value of `encr_alg` as `none`. If an ESP authentication algorithm is specified, but the encryption algorithm is not, the default value for the ESP encryption will be `null`.

`-failover`
Mark the address as a non-failover address.Addresses marked this way will not failover when the interface fails. Status display shows "NOFAILOVER" as part of flags.

`failover`
Mark the address as a failover address. This address will failover when the interface fails. Status display does not show "NOFAILOVER" as part of flags.

`group` [*name* | `""`]
Insert the interface in the multipathing group specified by *name*. To delete an interface from a group, use a null string `""`. When invoked on the logical interface with id zero, the status display shows the group name.

`index` *n*
Change the interface index for the interface. The value of *n* must be an interface index (*if_index*) that is not used on another interface. *if_index* will be a non-zero positive number that uniquely identifies the network interface on the system.

`metric` *n*
Set the routing metric of the interface to *n*; if no value is specified, the default is `0`. The routing metric is used by the routing protocol. Higher metrics have the effect of making a route less favorable; metrics are counted as addition hops to the destination network or host.

`modinsert` *mod_name@pos*
Insert a module with name *mod_name* to the stream of the device at position *pos*. The position is relative to the stream head. Position `0` means directly under stream head.

Based upon the example in the `modlist` option, use the following command to insert a module with name `ipqos` under the `ip` module and above the firewall module:

```
example% ifconfig hme0 modinsert ipqos@2
```

A subsequent listing of all the modules in the stream of the device follows:

```
example% ifconfig hme0 modlist
0 arp
1 ip
2 ipqos
3 firewall
4 hme
```

`modlist`
List all the modules in the stream of the device.

The following example lists all the modules in the stream of the device:

```
example% ifconfig hme0 modlist
0 arp
1 ip
2 firewall
4 hme
```

`modremove` *mod_name@pos*
Remove a module with name *mod_name* from the stream of the device at position *pos*. The position is relative to the stream head.

Based upon the example in the `modinsert` option, use the following command to remove the firewall module from the stream after inserting the `ipqos` module:

```
example% ifconfig hme0 modremove firewall@3
```

A subsequent listing of all the modules in the stream of the device follows:

```
example% ifconfig hme0 modlist
0 arp
1 ip
2 ipqos
3 hme
```

Note that the core IP stack modules, for example, `ip` and `tun` modules, cannot be removed.

`mtu` *n*
Set the maximum transmission unit of the interface to *n*. For many types of networks, the `mtu` has an upper limit, for example, `1500` for Ethernet.

`netmask` *mask*
For IPv4 only. Specify how much of the address to reserve for subdividing networks into subnetworks. The mask includes the network part of the local address and the subnet part, which is taken from the host field of the address. The mask contains 1's for the bit positions in the 32-bit address which are to be used for the network and subnet parts, and 0's for the host part. The mask should contain at least the standard network portion, and the subnet field should be contiguous with the network portion. The mask can be specified in one of four ways:

1. with a single hexadecimal number with a leading 0x,
2. with a dot-notation address,
3. with a "+" (plus sign) address, or
4. with a pseudo host name/pseudo network name found in the network database `networks`(4).

If a "+" (plus sign) is given for the netmask value, the mask is looked up in the `netmasks`(4) database. This lookup finds the longest matching netmask in the database by starting with the interface's IPv4 address as the key and iteratively masking off more and more low order bits of the address. This iterative lookup ensures that the `netmasks`(4) database can be used to specify the netmasks when variable length subnetmasks are used within a network number.

If a pseudo host name/pseudo network name is supplied as the netmask value, netmask data may be located in the `hosts` or `networks` database. Names are looked up by first using `gethostbyname`(3NSL). If not found there, the names are looked up in `getnetbyname`(3SOCKET). These interfaces may in turn use `nsswitch.conf`(4) to determine what data store(s) to use to fetch the actual value.

For both `inet` and `inet6`, the same information conveyed by *mask* can be specified as a *prefix_length* attached to the *address* parameter.

`nud`
Enables the neighbor unreachability detection mechanism on a point-to-go interface.

`-nud`
Disables the neighbor unreachability detection mechanism on a point-to-go interface.

`plumb`
Open the device associated with the physical interface name and set up the streams needed for IP to use the device. When used with a logical interface name, this command is used to create a specific named logical interface. An interface must be separately plumbed for use by IPv4 and IPv6. The *address_family* parameter controls whether the `ifconfig` command applies to IPv4 or IPv6.

Before an interface has been plumbed, , the interface will not show up in the output of the `ifconfig -a` command.

`private`
Tells the `in.routed` routing daemon that the interface should not be advertised.

`-private`
Specify unadvertised interfaces.

`removeif` *address*
Remove the logical interface on the physical interface specified that matches the *address* specified.

`set`
Set the *address*, *prefix_length* or both, for an interface.

`standby`
Marks the physical interface as a standby interface. If the interface is marked `STANDBY` and is part of the multipathing group, the interface will not be selected to send out packets unless some other interface in the group has failed and the network access has been failed over to this standby interface.

The status display shows "`STANDBY, INACTIVE`" indicating that that the interface is a standby and is also inactive. `IFF_INACTIVE` will be cleared when some other interface belonging to the same multipathing group fails over to this interface. Once a failback happens, the status display will return to `INACTIVE`.

`-standby`
Turns off standby on this interface.

`subnet`
Set the subnet *address* for an interface.

`tdst` *tunnel_dest_address*
Set the destination address of a tunnel. The address should not be the same as the `dest_address` of the tunnel, because no packets leave the system over such a tunnel.

`token` *address/prefix_length*
Set the IPv6 token of an interface to be used for address autoconfiguration.

```
example% ifconfig hme0 inet6 token ::1/64
```

`trailers`
This flag previously caused a nonstandard encapsulation of `inet` packets on certain link levels. Drivers supplied with this release no longer use this flag. It is provided for compatibility, but is ignored.

`-trailers`
Disable the use of a "trailer" link level encapsulation.

`tsrc` *tunnel_src_address*
Set the source address of a tunnel. This is the source address on an outer encapsulating IP header. It must be an address of another interface already configured using `ifconfig`.

`unplumb`
Close the device associated with this physical interface name and any streams that `ifconfig` set up for IP to use the device. When used with a logical interface name, the logical interface is removed from the system. After this command is executed, the device name will no longer appear in the output of `ifconfig -a`.

`up`
Mark an interface "`up`". This happens automatically when setting the first address on an interface. The `up` option enables an interface after an `ifconfig down`, which reinitializes the hardware.

`xmit`
Enable an interface to transmit packets. This is the default behavior when the interface is up.

`-xmit`
Disable transmission of packets on an interface. The interface will continue to receive packets.

OPERANDS

The *interface* operand, as well as address parameters that affect it, are described below.

interface
A string of the form, *name physical-unit*, for example, `le0` or `ie1`; or of the form *name physical-unit*`:`*logical-unit*, for example, `le0:1`; or of the form `ip.tun`*N*, for tunnels.

If the interface name starts with a dash (-), it is interpreted as a set of options which specify a set of interfaces. In such a case, `-a` must be part of the options and any of the additional options below can be added in any order. If one of these interface names is given, the commands following it are applied to all of the interfaces that match.

`-a`
Apply the commands to all interfaces in the system.

`-d`
Apply the commands to all "down" interfaces in the system.

	`-D` Apply the commands to all interfaces not under DHCP (Dynamic Host Configuration Protocol) control.
	`-u` Apply the commands to all "up" interfaces in the system.
	`-4` Apply the commands to all IPv4 interfaces.
	`-6` Apply the commands to all IPv6 interfaces.
address_family	The address family is specified by the *address_family* parameter. The `ifconfig` command currently supports the following families: `ether`, `inet`, and `inet6`. If no address family is specified, the default is `inet`.
address	For the IPv4 family (`inet`), the *address* is either a host name present in the host name data base (see `hosts`(4)) or in the Network Information Service (NIS) map `hosts`, or an IPv4 address expressed in the Internet standard "dot notation". For the IPv6 family (`inet6`), the *address* is either a host name present in the host name data base (see `ipnodes`(4)) or in the Network Information Service (NIS) map ipnode, or an IPv6 address expressed in the Internet standard colon-separated hexadecimal format represented as *x*:*x*:*x*:*x*:*x*:*x*:*x*:*x* where *x* is a hexadecimal number between `0` and `FFFF`. For the `ether` address family, the address is an Ethernet address represented as *x*:*x*:*x*: *x*:*x*:*x* where *x* is a hexadecimal number between `0` and `FF`. Some, though not all, of the Ethernet interface cards have their own addresses. To use cards that do not have their own addresses, refer to section 3.2.3(4) of the IEEE 802.3 specification for a definition of the locally administered address space. The use of interface groups should be restricted to those cards with their own addresses (see `INTERFACE GROUPS`).
prefix_length	For the IPv4 and IPv6 families (`inet` and `inet6`), the *prefix_length* is a number between 0 and the number of bits in the address. For `inet`, the number of bits in the address is 32; for `inet6`, the number of bits in the address is 128. The *prefix_length* denotes the number of leading set bits in the netmask.

dest_address	If the *dest_address* parameter is supplied in addition to the *address* parameter, it specifies the address of the correspondent on the other end of a point-to-point link.
tunnel_dest_address	An address that is or will be reachable through an interface other than the tunnel being configured. This tells the tunnel where to send the tunneled packets. This address must not be the same as the *tunnel_dest_address* being configured.
tunnel_src_address	As address that is attached to an already configured interface that has been configured "up" with `ifconfig`.

LOGICAL INTERFACES

Solaris TCP/IP allows multiple logical interfaces to be associated with a physical network interface. This allows a single machine to be assigned multiple IP addresses, even though it may have only one network interface. Physical network interfaces have names of the form *driver-name physical-unit-number*, while logical interfaces have names of the form *driver-name physical-unit-number*`:`*logical-unit-number*. A physical interface is configured into the system using the `plumb` command. For example:

```
example% ifconfig le0 plumb
```

Once a physical interface has been "plumbed", logical interfaces associated with the physical interface can be configured by separate `plumb` or `addif` options to the `ifconfig` command.

```
example% ifconfig le0:1 plumb
```

allocates a specific logical interface associated with the physical interface `le0`. The command

```
example% ifconfig le0 addif 192.9.200.1/24 up
```

allocates the next available logical unit number on the `le0` physical interface and assigns an *address* and *prefix_length*.

A logical interface can be configured with parameters (*address*,*prefix_length*, and so on) different from the physical interface with which it is associated. Logical interfaces that are associated with the same physical interface can be given different parameters as well. Each logical interface must be associated with an existing and "up" physical interface. So, for example, the logical interface `le0:1` can only be configured after the physical interface `le0` has been plumbed.

To delete a logical interface, use the `unplumb` or `removeif` options. For example,

```
example% ifconfig le0:1 down unplumb
```

will delete the logical interface `le0:1`.

INTERFACE GROUPS

If a physical interface shares an IP prefix with another interface, these interfaces are collected into an `interface group`. IP uses an interface group to rotate source address selection when the source address is unspecified, and in the case of multiple physical interfaces in the same group, to scatter traffic across different IP addresses on a per-IP-destination basis. See `netstat`(1M) for per-IP-destination information.

This feature may be enabled by using `ndd`(1M).

One can also use the `group` keyword to form a multipathing group. When multipathing groups are used, the functionality of the `interface group` is subsumed into the functionality of the multipathing group. A multipathing group provides failure detection and repair detection for the interfaces in the group. See `in.mpathd`(1M) and *System Administration Guide, Volume 3*.

The interface groups formed using `ndd`(1M) will be made obsolete in the future. Accordingly, it is advisable to use form multipathing groups using the `group` keyword.

CONFIGURING IPv6 INTERFACES

When an IPv6 physical interface is plumbed and configured "up" with `ifconfig`, it is automatically assigned an IPv6 link-local address for which the last 64 bits are calculated from the MAC address of the interface.

```
ifconfig le0 inet6 plumb up
```

The following example shows that the link-local address has a prefix of `fe80::/10`.

```
example% ifconfig le0 inet6
le0: flags=2000841<UP,RUNNING,MULTICAST,IPv6>
           mtu 1500 index 2
        inet6 fe80::a00:20ff:fe8e:f3ad/10
```

If an advertising IPv6 router exists on the link advertising prefixes, then the newly plumbed IPv6 interface will autoconfigure logical interface(s) depending on the prefix advertisements. For example, for prefix advertisements `fec0:0:0:55::/64` and `3ff0:0:0:55::/64`, the autoconfigured interfaces will look like:

```
le0:1: flags=2080841<UP,RUNNING,MULTICAST,ADDRCONF,IPv6>
          mtu 1500 index 2
        inet6 fec0::55:a00:20ff:fe8e:f3ad/64
le0:2: flags=2080841<UP,RUNNING,MULTICAST,ADDRCONF,IPv6>
          mtu 1500 index 2
        inet6 3ff0::55:a00:20ff:fe8e:f3ad/64
```

Even if there are no prefix advertisements on the link, you can still assign site-local and global addresses manually, for example:

```
example% ifconfig le0 inet6 addif fec0::55:a00:20ff:fe8e:f3ad/64 up
example% ifconfig le0 inet6 addif 3ff0::55:a00:20ff:fe8e:f3ad/64 up
```

To configure boot-time defaults for the interface `le0`, place the following entries in the `/etc/hostname6.le0` file:

```
addif  fec0::55:a00:20ff:fe8e:f3ad/64 up
addif  3ff0::55:a00:20ff:fe8e:f3ad/64 up
```

Link-local addresses are only used for on-link communication and are not visible to other subnets.

Configuring IPv6/IPv4 tunnels

An IPv6 over IPv4 tunnel interface can send and receive IPv6 packets encapsulated in an IPv4 packet. Create tunnels at both ends pointing to each other. IPv6 over IPv4 tunnels require the tunnel source and tunnel destination IPv4 and IPv6 addresses. Solaris 8 supports both automatic and configured tunnels. For automatic tunnels, an IPv4-compatible IPv6 address is used. The following demonstrates auto-tunnel configuration:

```
example% ifconfig ip.atun0 inet6 plumb
example% ifconfig ip.atun0 inet6 tsrc <IPv4-address> \
   ::<IPv4 address>/96 up
```

where `IPv4-address` is the IPv4 address of the interface through which the tunnel traffic will flow, and `IPv4-address`, `::<IPv4-address>`, is the corresponding IPv4-compatible IPv6 address.

The following is an example of a configured tunnel:

```
example% ifconfig ip.tun0 inet6 plumb tsrc <my-ipv4-address> \
   tdst <peer-ipv4-address> up
```

This creates a configured tunnel between `my-ipv4-address` and `peer-ipv4-address` with corresponding link-local addresses. For tunnels with global or site-local addresses, the logical tunnel interfaces need to be configured in the following form:

```
ifconfig ip.tun0 inet6 addif <my-v6-address> <peer-v6-address> up
```

For example,

```
example% ifconfig ip.tun0 inet6 plumb tsrc 109.146.85.57 \
   tdst 109.146.85.212 up
example% ifconfig ip.tun0 inet6 addif 2::45 2::46 up
```

To show all IPv6 interfaces that are up and configured:

```
example% ifconfig -au6
ip.tun0: flags=2200851<UP,POINTOPOINT,RUNNING,MULTICAST,NONUD,IPv6>
          mtu 1480 index 3
        inet tunnel src 109.146.85.57   tunnel dst 109.146.85.212
        inet6 fe80::6d92:5539/10 --> fe80::6d92:55d4
```

```
ip.tun0:1: flags=2200851<UP,POINTOPOINT,RUNNING,MULTICAST,NONUD,IPv6>
          mtu 1480 index 3
        inet6 2::45/128 --> 2::46
```

EXAMPLES

EXAMPLE 1 Using the `ifconfig` Command

If your workstation is not attached to an Ethernet, the `le0` interface should be marked "down" as follows:

```
example% ifconfig le0 down
```

EXAMPLE 2 Printing Addressing Information

To print out the addressing information for each interface, use the following command:

```
example% ifconfig -a
```

EXAMPLE 3 Resetting the Broadcast Address

To reset each interface's broadcast address after the netmasks have been correctly set, use the next command:

```
example% ifconfig -a broadcast +
```

EXAMPLE 4 Changing the Ethernet Address

To change the Ethernet address for interface `le0`, use the following command:

```
example% ifconfig le0 ether aa:1:2:3:4:5
```

EXAMPLE 5 Configuring an IP-in-IP Tunnel

To configure an IP-in-IP tunnel, first plumb it with the following command:

```
example% ifconfig ip.tun0 plumb
```

Then configure it as a point-to-point interface, supplying the tunnel source and the tunnel destination:

```
example% ifconfig ip.tun0 myaddr mydestaddr tsrc another_myaddr \
          tdst a_dest_addr up
```

Tunnel security properties must be configured on one invocation of `ifconfig`:

```
example% ifconfig ip.tun0 encr_auth_algs md5 encr_algs 3des
```

EXAMPLE 6 Requesting a Service Without Algorithm Preference

To request a service without any algorithm preferences, specify `any`:

```
example% ifconfig ip.tun0 encr_auth_algs any encr_algs any
```

EXAMPLE 7 Disabling All Security

To disable all security, specify any security service with `none` as the algorithm value:

```
example% ifconfig ip.tun0 auth_algs none
```

or

```
example% ifconfig ip.tun0 encr_algs none
```

FILES

`/etc/netmasks` netmask data

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

/usr/sbin

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu
Interface Stability for options `modlist`, `modinsert`, and `modremove`	Evolving

/sbin

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsr
Interface Stability for options `modlist`, `modinsert`, and `modremove`	Evolving

SEE ALSO

`dhcpinfo`(1), `dhcpagent`(1M), `in.mpathd`(1M), `in.routed`(1M), `ndd`(1M), `netstat`(1M), `ethers`(3SOCKET), `gethostbyname`(3NSL), `getnetbyname`(3SOCKET), `hosts`(4), `netmasks`(4), `networks`(4), `nsswitch.conf`(4), `attributes`(5), `arp`(7P),`ipsecah`(7P),`ipsecesp`(7P),`tun`(7M)

System Administration Guide, Volume 3

DIAGNOSTICS

`ifconfig` sends messages that indicate if:

- the specified interface does not exist

- the requested address is unknown
- the user is not privileged and tried to alter an interface's configuration

NOTES

It is recommended that the names `broadcast`, `down`, `private`, `trailers`, `up`, and the other possible option names not be selected when choosing host names. Choosing any one of these names as host names will cause bizarre problems that can be extremely difficult to diagnose.

NAME

if_mpadm – change operational status of interfaces within a multipathing group

SYNOPSIS

/usr/sbin/if_mpadm `-d` *interface_name*

/usr/sbin/if_mpadm `-r` *interface_name*

DESCRIPTION

Use the `if_mpadm` utility to change the operational status of interfaces that are part of a multipathing group. If the interface is operational, you can use `if_mpadm -d` to detach or off-line the interface. If the interface is off-lined, use `if_mpadm -r` to revert it to its original state.

When a network interface is off-lined, all network access fails over to a different interface in the multipathing group. Any addresses that do not failover are brought down. Network access includes unicast, broadcast, and multicast for IPv4 and unicast and multicast for IPv6. Addresses marked with `IFF_NOFAILOVER` do not failover; they are marked down. After an interface is off-lined, the system will not use the interface for any outbound or inbound traffic, and the interface can be safely removed from the system without any loss of network access.

The `if_mpadm` utility can be applied only to interfaces that are part of a multipathing group.

OPTIONS

The `if_mpadm` utility supports the following options:

`-d` *interface_name*	Detach or off-line the interface specified by *interface_name*.
`-r` *interface_name*	Reattach or undo the previous detach or off-line operation on the interface specified by *interface_name*. Unless the `-d` option was used to detach or off-line the interface, this option will fail.

EXAMPLES

EXAMPLE 1 Detaching an Interface

Use the following command to off-line or detach the interface. All network access will failback to `hme0` from another interface in the group.

```
example% if_mpadm -d hme0
```

EXAMPLE 2 Reattaching an Off-line Interface

Use the following command to undo the previous operation. Network access will failback over to `hme0` from an interface in the same multipathing group.

```
example% if_mpdadm -r hme0
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu
Interface Stability	Unstable

SEE ALSO `ifconfig`(1M), `in.mpathd`(1M), `attributes`(5)

DIAGNOSTICS

`off-line failed as there is no other functional interface available in the multipathing group for failing over the network access.`

This message means that other interfaces in the group are failed over already or the multipathing configuration was not suitable for completing a failover.

`off-line cannot be undone as failback has been disabled.`

`FAILBACK` is set to "`no`" in `/etc/default/mpathd`.

`off-line cannot be undone because multipathing configuration is not consistent across all the interfaces in the group.`

This message means that some interfaces in the multipathing group are not configured consistently with other interfaces in the group, for example, one of the interfaces in the group does not have an `IFF_NOFAILOVER` address.

NAME

ifparse – parse `ifconfig` command line

SYNOPSIS

`/sbin/ifparse` `[-fs]` *addr_family commands*

DESCRIPTION

Use the `ifparse` command to parse the `ifconfig`(1M) command line options and output substrings, one per line, as appropriate. If no options are specified, `ifparse` returns the entire `ifconfig` command line as a series of substrings, one per line.

OPTIONS

The `ifparse` command supports the following options:

`-f` Lists only substrings of the `ifconfig` command line that are relevant to IP network multipath failover

`-s` Lists only substrings of the `ifconfig` command line that are not relevant to IP network multipath failover

OPERANDS

The `ifparse` command *does not* support the *interface* operand of the `ifconfig` command.

EXAMPLES

EXAMPLE 1 Parsing Command Line Options Relevant to Failover

The following example shows the use of the `ifparse` command to parse the command line options relevant to IP network multipath failover:

```
example# ifparse -f inet 1.2.3.4 up group one addif 1.2.3.5 -failover up
set 1.2.3.4 up
```

EXAMPLE 2 Parsing Command Line Options That Are Not Relevant to Failover

The following example shows the use of the `ifparse` command to parse the command line options that are not relevant to IP network multipath failover:

```
example# ifparse -s inet 1.2.3.4 up group one addif 1.2.3.5 -failover up
group one
addif 1.2.3.5 -failover up
```

EXAMPLE 3 Parsing the Command Line For All Options

The following example shows the use of the `ifparse` command to parse the command line for all `ifconfig` options:

```
example# ifparse inet 1.2.3.4 up group one addif 1.2.3.5 -failover up
group one
set 1.2.3.4 up
addif 1.2.3.5 -failover up
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsr
Stability Level	Obsolete

SEE ALSO

`ifconfig`(1M), `attributes`(5)

DIAGNOSTICS

`usage: -fs <addr_family> <commands>`
This message indicates an invalid command line.

`ifparse: Not enough space`
This message indicates insufficient memory.

`ifparse: dhcp not supported for inet6`
DHCP operations are not supported for the `inet6` address family.

`ifparse: Operation <operation> not supported for <addr_family>`
Most operations cannot be used with all address families. For example, the `broadcast` operation is not supported on the `inet6` address family.

`ifparse: no argument for <operation>`
Some operations, for example `broadcast`, require an argument.

NOTES

The `ifparse` command is classified as an obsolete interface. It will likely be removed in a future release. You should not develop applications that depend upon this interface.

NAME

ikeadm – manipulate Internet Key Exchange (IKE) parameters and state

SYNOPSIS

```
ikeadm [-np]
ikeadm [-np] get [debug | priv | stats]
ikeadm [-np] set [debug | priv] [level] [file]
ikeadm [-np] [get | del] [p1 | rule | preshared] [id]
ikeadm [-np] add [rule | preshared]{ description }
ikeadm [-np] [read | write] [rule | preshared]file
ikeadm [-np] [dump | pls | rule | preshared]
ikeadm [-np] flush pls
ikeadm help [get | set | add | del | read | write | dump | flush]
```

DESCRIPTION

The `ikeadm` utility retrieves information from and manipulates the configuration of the Internet Key Exchange (IKE) protocol daemon, `in.iked`(1M).

`ikeadm` supports a set of operations, which may be performed on one or more of the supported object types. When invoked without arguments, `ikeadm` enters interactive mode which prints a prompt to the standard output and accepts commands from the standard input until the end-of-file is reached.

Because `ikeadm` manipulates sensitive keying information, you must be superuser to use this command. Additionally, some of the commands available require that the daemon be running in a privileged mode, which is established when the daemon is started.

For details on how to use this command securely see `SECURITY`.

OPTIONS

The following options are supported:

`-n` Prevent attempts to print host and network names symbolically when reporting actions. This is useful, for example, when all name servers are down or are otherwise unreachable.

`-p` Paranoid. Do not print any keying material, even if saving Security Associations. Instead of an actual hexadecimal digit, print an `X` when this flag is turned on.

Commands

The following commands are supported:

add Add the specified object. This option can be used to add a new policy rule or a new preshared key to the current (running) in.iked configuration. When adding a new preshared key, the command cannot be invoked from the command line, as it will contain keying material. The rule or key being added is specified using appropriate id-value pairs as described in the `ID FORMATS` section.

	del	Delete a specific object from `in.iked`'s current configuration. This operation is available for IKE (Phase 1) SAs, policy rules, and preshared keys. The object to be deleted is specified as described in the `ID FORMATS`.
	dump	Display all objects of the specified type known to `in.iked`. This option can be used to display all Phase 1 SAs, policy rules, or preshared keys. A large amount of output may be generated by this command.
	flush	Remove all IKE (Phase 1) SAs from `in.iked`.
	get	Lookup and display the specified object. May be used to view the current debug or privilege level, global statistics for the daemon, or a specific IKE (Phase 1) SA, policy rule, or preshared key. The latter three object types require that identifying information be passed in; the appropriate specification for each object type is described below.
	help	Print a brief summary of commands, or, when followed by a command, prints information about that command.
	read	Update the current `in.iked` configuration by reading the policy rules or preshared keys from either the default location or from the file specified.
	set	Adjust the current debug or privilege level. If the debug level is being modified, an output file may optionally be specified; the output file *must* be specified if the daemon is running in the background and is not currently printing to a file. When changing the privilege level, adjustments may only be made to lower the access level; it cannot be increased using ikeadm.
	write	Write the current `in.iked` policy rule set or preshared key set to the specified file. A destination file must be specified. This command should not be used to overwrite the existing configuration files.
OBJECT TYPES	debug	Specifies the daemon's debug level. This determines the amount and type of output provided by the daemon about its operations. The debug level is actually a bitmask, with individual bits enabling different types of information.

Description	Flag	Nickname
Certificate management	0x0001	cert
Key management	0x0002	key
Operational	0x0004	op

Description	Flag	Nickname
Phase 1 SA creation	0x0008	phase1
Phase 2 SA creation	0x0010	phase2
PF_KEY interface	0x0020	pfkey
Policy management	0x0040	policy
Proposal construction	0x0080	prop
Door interface	0x0100	door
Config file processing	0x0200	config
All debug flags	0x3ff	all

When specifying the debug level, either a number (decimal or hexadecimal) or a string of nicknames may be given. For example, `88`, `0x58`, and `phase1+phase2+policy` are all equivalent, and will turn on debug for `phase 1 sa` creation, `phase 2 sa` creation, and policy management. A string of nicknames may also be used to remove certain types of information; `all-op` has the effect of turning on all debug *except* for operational messages; it is equivalent to the numbers `1019` or `0x3fb`.

priv

Specifies the daemon's access privilege level. The possible values are:

Description	Level	Nickname
Base level	0	base
Access to preshared key info	1	modkeys
Access to keying material	2	keymat

By default, `in.iked` is started at the base level. A command-line option can be used to start the daemon at a higher level. `ikeadm` can be used to lower the level, but it cannot be used to raise the level.

Either the numerical level or the nickname may be used to specify the target privilege level.

In order to get, add, delete, dump, read, or write preshared keys, the privilege level must at least give access to preshared key information. However, when viewing preshared keys (either using the get or dump command), the key itself will only be available if the privilege level gives access to keying material. This is also the case when viewing Phase 1 SAs.

stats — Global statistics from the daemon, covering both successful and failed Phase 1 SA creation.

Reported statistics include:

- Count of current P1 SAs which the local entity initiated
- Count of current P1 SAs where the local entity was the responder
- Count of all P1 SAs which the local entity initiated since boot
- Count of all P1 SAs where the local entity was the responder since boot
- Count of all attempted P1 SAs since boot, where the local entity was the initiator; includes failed attempts
- Count of all attempted P1 SAs since boot, where the local entity was the responder; includes failed attempts
- Count of all failed attempts to initiate a P1 SA, where the failure occurred because the peer did not respond
- Count of all failed attempts to initiate a P1 SA, where the peer responded
- Count of all failed P1 SAs where the peer was the initiator

p1 — An IKE Phase 1 SA. A `p1` object is identified by an IP address pair or a cookie pair; identification formats are described below.

rule — An IKE policy rule, defining the acceptable security characteristics for Phase 1 SAs between specified local and remote identities. A rule is identified by its label; identification formats are described below.

preshared — A preshared key, including the local and remote identification and applicable IKE mode. A preshared key is identified by an IP address pair or an identity pair; identification formats are described below.

ID FORMATS

Commands like `add`, `del`, and `get` require that additional information be specified on the command line. In the case of the delete and get commands, all that is required is to minimally identify a given object; for the add command, the full object must be specified.

Minimal identification is accomplished in most cases by a pair of values. For IP addresses, the local addr and then the remote addr are specified, either in dot-notation for IPv4 addresses, colon-separated hexadecimal format for IPv6 addresses, or a host name present in the host name database. If a host name is given that expands to more than one address, the requested operation will be performed multiple times, once for each possible combination of addresses.

Identity pairs are made up of a local type-value pair, followed by the remote type-value pair. Valid types are:

prefix	An address prefix.
fqdn	A fully-qualified domain name.
domain	Domain name, synonym for fqdn.
user_fqdn	User identity of the form *user*@fqdn.
mailbox	Synonym for `user_fqdn`.

A cookie pair is made up of the two cookies assigned to a Phase 1 Security Association (SA) when it is created; first is the initiator's, followed by the responder's. A cookie is a 64-bit number.

Finally, a label (which is used to identify a policy rule) is a character string assigned to the rule when it is created.

Formatting a rule or preshared key for the add command follows the format rules for the in.iked configuration files. Both are made up of a series of id-value pairs, contained in curly braces ({ and }). See `ike.config`(4) and `ike.preshared`(4) for details on the formatting of rules and preshared keys.

SECURITY

The `ikeadm` command allows a privileged user to enter cryptographic keying information. If an adversary gains access to such information, the security of IPsec traffic is compromised. The following issues should be taken into account when using the `ikeadm` command.

- Is the TTY going over a network (interactive mode)?

  If it is, then the security of the keying material is the security of the network path for this TTY's traffic. Using `ikeadm` over a clear-text telnet or rlogin session is risky. Even local windows may be vulnerable to attacks where a concealed program that reads window events is present.
- Is the file accessed over the network or readable to the world (read/write commands)?

  A network-mounted file can be sniffed by an adversary as it is being read. A world-readable file with keying material in it is also risky.

If your source address is a host that can be looked up over the network, and your naming system itself is compromised, then any names used will no longer be trustworthy.

Security weaknesses often lie in misapplication of tools, not the tools themselves. It is recommended that administrators are cautious when using the `ikeadm` command. The safest mode of operation is probably on a console, or other hard-connected TTY.

For additional information regarding this subject, see the afterward by Matt Blaze in Bruce Schneier's *Applied Cryptography: Protocols, Algorithms, and Source Code in C.*

EXAMPLES

EXAMPLE 1 Emptying out all Phase 1 Security Associations

The following command empties out all Phase 1 Security Associations:

```
example# ikeadm flush p1s
```

EXAMPLE 2 Displaying all Phase 1 Security Associations

The following command displays all Phase 1 Security Associations:

```
example# ikeadm dump p1s
```

EXAMPLE 3 Deleting a Specific Phase 1 Security Association

The following command deletes the specified Phase 1 Security Associations:

```
example# ikeadm get p1 local_ip remote_ip
```

EXAMPLE 4 Adding a Rule From a File

The following command adds a rule from a file:

```
example# ikeadm add rule rule_file
```

EXAMPLE 5 Adding a Preshared Key

The following command adds a preshared key:

```
example# ikeadm
     ikeadm> add preshared { localidtype ip localid local_ip
             remoteidtype ip remoteid remote_ip ike_mode main
             key 1234567890abcdef1234567890abcdef }
```

EXAMPLE 6 Saving All Preshared Keys to a File

The following command saves all preshared keys to a file:

```
example# ikeadm write preshared target_file
```

EXAMPLE 7 Viewing a Particular Rule

The following command views a particular rule:

```
example# ikeadm get rule rule_label
```

EXAMPLE 8 Reading in New Rules from `ike.config`

The following command reads in new rules from the ike.config file:

```
example# ikeadm read rules
```

EXAMPLE 9 Lowering the Privilege Level

The following command lowers the privilege level:

```
example# ikeadm set priv base
```

EXAMPLE 10 Viewing the debug level

The following command the current debug level

```
example# ikeadm get debug
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`non-zero` An error occurred. Writes an appropriate error message to standard error.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu
Interface Stability	Evolving

SEE ALSO

`in.iked`(1M), `ike.config`(4), `ike.preshared`(4), `attributes`(5), `ipsec`(7P)

Schneier, Bruce, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, Second Edition, John Wiley & Sons, New York, NY, 1996.

NAME

ikecert – manipulates the machine's on-filesystem public-key certificate databases

SYNOPSIS

ikecert [certlocal] [-a | -e | -h | -k | -l | -r] [*option_specific_arguments*...]

ikecert [certdb] [-a | -e | -h | -l | -r] [*option_specific_arguments*...]

ikecert [certldb] [-a | -e | -h | -l | -r] [*option_specific_arguments*...]

DESCRIPTION

The ikecert command manipulates the machine's on-filesystem public-key certificate databases. See FILES.

ikecert has three subcommands, one for each of the three major repositories:

- certlocal deals with the private-key repository,
- certdb deals with the public-key repository, and
- certrldb deals with the certificate revocation list (CRL) repository.

OPTIONS

Each subcommand requires one option, possibly followed by one or more option-specific arguments.

The following options are supported:

-a

certlocal — When specified with the certlocal subcommand, this option installs (adds) a private key into the Internet Key Exchange (IKE) local ID database. The key data is read from standard input, and is in a Solaris-only format.

certdb — When specified with the certdb subcommand, this option reads a certificate from standard input and adds it to the IKE certificate database. The certificate must be a X.509 certificate in PEM Base64 or ASN.1 BER encoding. The certificate adopts the name of its identity.

certrldb — When specified with the certrldb subcommand, this option installs (adds) a CRL into the IKE database. The CRL reads from standard input.

-e *slot*

certlocal — When specified with the certlocal subcommand, this option extracts a private key from the IKE local ID database. The key data are written to standard output. The lot specifies which private key to extract. Private keys are only extracted in binary/ber format.

Use this option with extreme caution. See SECURITY.

-e [-f *output-format*] certspec

certdb — When specified with the certdb subcommand, this option extracts a certificate from the IKE certificate database which matches the certspec and writes it to standard output. The *output-format* option specifies the

encoding format. Valid options are `PEM` and `BER`. This extracts the first matching identity. The default output format is `PEM`.

certrldb — When specified with the `certrldb` subcommand, this option extracts a CRL from the IKE database. The key data are written to standard output. The `certspec` specifies which CRL that is extracted. The first one that matches in the database is extracted. See `PARAMETERS` for details on `certspec` patterns.

`-kc -m` *keysize* `-t` *keytype* `-D` *dname* `-A` *altname*[...]

certlocal — When specified with the `certlocal` subcommand, this option generates a IKE public/private key pair and adds it into the local ID database. It also generates a certificate request and sends that to standard output. For details on the above options see PARAMETERS for details on the *dname* argument and see ALTERNATIVE NAMES for details on the *altname* argument(s) to this command.

`-ks -m` *keysize* `-t` *keytype* `-D` *dname* `-A` *altname*[...] [`-f` *output-format*]

certlocal — When specified with the `certlocal` subcommand, generates a public/private key pair and adds it into the local ID database. This option also generates a self-signed certificate and installs it into the certificate database. See `PARAMETERS` for details on the *dname* and *altname* arguments to this command.

`-l` [`-v`] [*slot*]

certlocal — When specified with the `certlocal` subcommand, this option lists private keys in the local ID database. The `-v` option switches output to a verbose mode where the entire certificate is printed.

Use the `-v` *option with extreme caution.* See `SECURITY`.

`-l` [`-v`] [certspec]

certdb — When specified with the `certdb` subcommand, this option lists certificates in the IKE certificate database matching the certspec, if any pattern is given. The list displays the identity string of the certificates, as well as, the private key if in the key database. The `-v` switches the output to a verbose mode where the entire certificate is printed.

certrldb — When specified with the `certrldb` subcommand, this option lists the CRLs in the IKE database along with any certificates that reside in the database and match the Issuer Name. `certspec` can be used to specify to list a specific CRL. The `-v`option switches the output to a verbose mode where the entire certificate is printed. See `PARAMETERS` for details on `certspec` patterns.

`-r` *slot*

certlocal — When specified with the `certlocal` subcommand, deletes the local ID in the specified slot. If there is a corresponding public key, it is not

be deleted.

`-r` certspec

certdb — Removes certificates from the IKE certificate database. Certificates matching the specified certificate pattern are deleted. Any private keys in the `certlocal` database corresponding to these certificates are not deleted. This removes the first matching identity.

certrldb — When specified with the `certrldb` subcommand, this option deletes the CRL with the given `certspec`.

PARAMETERS

The following parameters are supported:

certspec — Specifies the pattern matching of certificate specifications. Valid `certspec`s are the Subject Name, Issuer Name, and Subject Alternative Names.

These can be specified as certificates that match the given `certspec` values and that don't match other `certspec` values. To signify a `certspec` value that is not supposed to be present in a certificate, place an `!` in front of the tag.

Valid `certspec`s are:

```
<Subject Names>
SUBJECT=<Subject Names>
ISSUER=<Issuer Names>
SLOT=<Slot Number in the certificate database>

Example:"ISSUER=C=US, O=SUN" IP=1.2.3.4 !DNS=example.com
Example:"C=US,   O=CALIFORNIA" IP=5.4.2.1 DNS=sun.com
```

Valid arguments to the alternative names are as follows:

```
IP=<IPv4 address>
DNS=<Domain Name Server address>
EMAIL=<email (RFC 822) address>
URI=<Uniform Resource Indicator value>
DN=<LDAP Directory Name value>
RID=<Registered Identifier value>
```

Valid Slot numbers can be specified without the keyword tag. Alternative name can also be issued with keyword tags.

`-A` — Subject Alternative Names the certificate. The argument that follows the `-A` option should be in the form of *tag=value*. Valid tags are `IP`, `DNS`, `EMAIL`, `URI`, `DN`, and `RID` (See example below).

`-D` — `X.509` distinguished name for the certificate subject. It typically has the form of: `C`=country, `O`=organization, `OU`=organizational unit, `CN`=common name. Valid tags are: `C`, `O`, `OU`, and `CN`.

`-f`	Encoding output format. `pem` for `PEM Base64` or `ber` for `ASN.1 BER`. If `-f` is not specified, `pem` is assumed.
`-m`	Key size. It can be `512`, `1024`, `2048`, `3072`, or `4096`.
`-t`	Key type. It can be `rsa-sha1`, `rsa-md5`, or `dsa-sha1`.

SECURITY

This command can save private keys of a public-private key pair into a file. Any exposure of a private key may lead to compromise if the key is somehow obtained by an adversary.

Refer to the afterword by Matt Blaze in Bruce Schneier's *Applied Cryptography: Protocols, Algorithms, and Source Code in C* for additional information.

EXAMPLES

EXAMPLE 1 Generating a Self-Signed Certificate

The following is an example of a self-signed certificate:

```
# ikecert certlocal -ks -m 512 -t rsa-md5 -D "C=US, O=SUN" -A
IP=1.2.3.4
Generating, please wait...
Certificate generated.
Certificate added to database.
-----BEGIN X509 CERTIFICATE-----
MIIBRDCB76ADAgECAgEBMA0GCSqGSIb3DQEBBAUAMBsxCzAJBgNVBAYTAlVTMQww
CgYDVQQKEwNTVU4wHhcNMDEwMzE0MDEzMDM1WhcNMDUwMzE0MDEzMDM1WjAbMQsw
CQYDVQQGEwJVUzEMMAoGA1UEChMDU1VOMFowDQYJKoZIhvcNAQEBBQADSQAwRgJB
APDhqpKgjgRoRUr6twTMTtSuNsReEnFoReVer!ztpXpQK6ybYlRH18JIqU/uCV/r
26R/cVXTy5qc5NbMwA40KzcCASOjIDAeMAsGA1UdDwQEAwIFoDAPBgNVHREECDAG
hwQBAgMEMA0GCSqGSIb3DQEBBAUAA0EApTRD23KzN95GMvPD71hwwCluk slKLVg8
f1xm9ZsHLPJLRxHFwsqqjAad4j4wwwriiUmGAHLTGB0lJM18xsgxag==
-----END X509 CERTIFICATE-----
```

EXAMPLE 2 Generating a CA Request

Generating a `CA` request appears the same as the self-signed certificate. The only differences between the two is the option `-c` instead of `-s`, and the certificate data is a `CA` request.

```
# ikecert certlocal -kc -m 512 -t rsa-md5 \
   -D "C=US, O=SUN" -A IP=1.2.3.4
```

EXIT STATUS

The following exit values are returned:

`0`	Successful completion.
`non-zero`	An error occurred. Writes an appropriate error message to standard error.

FILES

`/etc/inet/secret/ike.privatekeys/*`
Private keys. A private key *must* have a matching public-key certificate with the same filename in `/etc/inet/ike/publickeys/`.

`/etc/inet/ike/publickeys/*`
Public-key certificates. The names are only important with regard to matching private key names.

`/etc/inet/ike/crls/*`
Public key certificate revocation lists.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`in.iked`(1M), `attributes`(5)

Schneier, Bruce, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, Second Edition, John Wiley & Sons, New York, NY, 1996.

NAME

in.comsat, comsat – biff server

SYNOPSIS

/usr/sbin/in.comsat

DESCRIPTION

`comsat` is the server process which listens for reports of incoming mail and notifies users who have requested to be told when mail arrives. It is invoked as needed by `inetd`(1M), and times out if inactive for a few minutes.

`comsat` listens on a datagram port associated with the `biff` service specification (see `services`(4)) for one line messages of the form

user@mailbox-offset

If the *user* specified is logged in to the system and the associated terminal has the owner execute bit turned on (by a `biff y`), the *offset* is used as a seek offset into the appropriate mailbox file, and the first 7 lines or 560 characters of the message are printed on the user's terminal. Lines which appear to be part of the message header other than the `From`, `To`, `Date`, or `Subject` lines are not printed when displaying the message.

FILES

`/var/adm/utmpx` user access and administration information

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWrcmds

SEE ALSO

`inetd`(1M), `services`(4), `attributes`(5)

NOTES

The message header filtering is prone to error.

NAME

in.dhcpd – Dynamic Host Configuration Protocol server

SYNOPSIS

/usr/lib/inet/in.dhcpd [-denv] [-h *relay_hops*] [-i *interface, ...*] [-l *syslog_local_facility*] [-b automatic | manual] [-o *DHCP_offer_time*] [-t *dhcptab_rescan_interval*]

/usr/lib/inet/in.dhcpd [-dv] [-h *relay_hops*] [-i *interface,...*] [-l *syslog_local_facility*] -r *IP_address* | *hostname, ...*

DESCRIPTION

in.dhcpd is a daemon that responds to Dynamic Host Configuration Protocol (DHCP) requests and optionally to BOOTP protocol requests. The daemon forks a copy of itself that runs as a background process. It must be run as root. The daemon has two run modes, DHCP server (with optional BOOTP compatibility mode) and BOOTP relay agent mode.

The first line in the SYNOPSIS section illustrates the options available in the DHCP/BOOTP server mode. The second line in the SYNOPSIS section illustrates the options available when the daemon is run in BOOTP relay agent mode.

The DHCP and BOOTP protocols are used to provide configuration parameters to Internet hosts. Client machines are allocated their IP addresses as well as other host configuration parameters through this mechanism.

The DHCP/BOOTP daemon manages two types of DHCP data tables: the dhcptab configuration table and the DHCP network tables.

See dhcptab(4) regarding the dhcptab configuration table and dhcp_network(4) regarding the DHCP network tables.

The dhcptab contains macro definitions defined using a termcap-like syntax which permits network administrators to define groups of DHCP configuration parameters to be returned to clients. However, a DHCP/BOOTP server always returns hostname, network broadcast address, network subnet mask, and IP maximum transfer unit (MTU) if requested by a client attached to the same network as the server machine. If those options have not been explicitly configured in the dhcptab, in.dhcpd returns reasonable default values.

The dhcptab is read at startup, upon receipt of a SIGHUP signal, or periodically as specified by the -t option. A SIGHUP (sent using the command pkill -HUP in.dhcpd) causes the DHCP/BOOTP daemon to reread the dhcptab within an interval from 0-60 seconds (depending on where the DHCP daemon is in its polling cycle). For busy servers, users should run /etc/init.d/dhcp stop, followed by /etc/init.d/dhcp start to force the dhcptab to be reread.

The DHCP network tables contain mappings of client identifiers to IP addresses. These tables are named after the network they support and the datastore used to maintain them.

The DHCP network tables are consulted during runtime. A client request received from a network for which no DHCP network table exists is ignored.

This command may change in future releases of Solaris software. Scripts, programs, or procedures that use this command might need modification when upgrading to future Solaris software releases.The command line options provided with the `in.dhcpd` daemon are used only for the current session, and include only some of the server options you can set. The `dhcpsvc.conf`(4) contains all the server default settings, and can be modified by using the `dhcpmgr` utility. See `dhcpsvc.conf`(4) for more details.

OPTIONS

The following options are supported:

`-b automatic | manual`
: This option enables BOOTP compatibility mode, allowing the DHCP server to respond to BOOTP clients. The option argument specifies whether the DHCP server should automatically allocate permanent lease IP addresses to requesting BOOTP clients if the clients are not registered in the DHCP network tables (`automatic`) or respond only to BOOTP clients who have been manually registered in the DHCP network tables (`manual`). This option only affects DHCP server mode.

`-d`
: Debugging mode. The daemon remains as a foreground process, and displays verbose messages as it processes DHCP and/or BOOTP datagrams. Messages are displayed on the current TTY. This option can be used in both DHCP/BOOTP server mode and BOOTP relay agent mode.

`-h` *relay_hops*
: Specifies the maximum number of relay agent hops that can occur before the daemon drops the DHCP/BOOTP datagram. The default number of relay agent hops is 4. This option affects both DHCP/BOOTP server mode and BOOTP relay agent mode.

`-i` *interface, . . .*
: Selects the network interfaces that the daemon should monitor for DHCP/BOOTP datagrams. The daemon ignores DHCP/BOOTP datagrams on network interfaces not specified in this list. This option is only useful on machines that have multiple network interfaces. If this option is not specified, then the daemon listens for DHCP/BOOTP datagrams on all network interfaces. The option argument consists of a comma-separated list of interface names. It affects both DHCP/BOOTP server and BOOTP relay agent run modes.

`-l` *syslog_local_facility*
: The presence of this option turns on transaction logging for the DHCP server or BOOTP relay agent. The value specifies the `syslog` local facility (an integer from `0` to `7` inclusive) the DHCP daemon should use for tagging the transactions. Using a facility separate from the `LOG_DAEMON` facility allows the network administrator to capture these transactions separately from other DHCP daemon events for such purposes as generating transaction reports. See `syslog`(3C), for details about local facilities. Transactions are logged using a record with 9 space-separated fields as follows:

  1. Protocol:

Relay mode: "BOOTP"
Server mode: "BOOTP" or "DHCP" based upon client type.

2. Type:

   Relay mode: "RELAY-CLNT", "RELAY-SRVR"
   Server mode: "ASSIGN", "EXTEND", "RELEASE", "DECLINE", "INFORM", "NAK" "ICMP-ECHO."

3. Transaction time: absolute time in seconds (unix time)

4. Lease time:

   Relay mode: Always 0.
   Server mode: 0 for ICMP-ECHO events, absolute time in seconds (unix time) otherwise

5. Source IP address: Dotted Internet form

   Relay mode: Relay interface IP on RELAY-CLNT, INADDR_ANY on RELAY-SRVR.
   Server mode: Client IP.

6. Destination IP address: Dotted Internet form

   Relay mode: Client IP on RELAY-CLNT, Server IP on RELAY-SRVR.
   Server mode: Server IP.

7. Client Identifier: Hex representation (0-9, A-F)

   Relay mode: MAC address
   Server mode: BOOTP - MAC address; DHCP - client id

8. Vendor Class identifier (white space converted to periods (.)).

   Relay mode: Always "N/A"
   Server mode: Vendor class ID tokenized by converting white space characters to periods (.)

9. MAC address: Hex representation (0-9, A-F)

   Relay mode: MAC address
   Server mode: MAC address

The format of this record is subject to change between releases.

Transactions are logged to the console if daemon is in debug mode (`-d`).

Logging transactions impact daemon performance.

It is suggested that you manage log file size periodically using a script run by `cron`(1M) and sending `syslogd`(1M) a `SIGHUP` signal. You could, for example, clone `/usr/lib/newsyslog` and alter it to match your DHCP logging requirements.

`-n`
Disable automatic duplicate IP address detection. When this option is specified, the DHCP server does not attempt to verify that an IP address it is about to offer a client is not in use. By default, the DHCP server pings an IP address before offering it to a DHCP/BOOTP client, to verify that the address is not in use by another machine.

`-o` *DHCP_offer_time*
Specifies the number of seconds the DHCP server should cache the offers it has extended to discovering DHCP clients. The default setting is `10` seconds. On slow network media, this value can be increased to compensate for slow network performance. This option affects only DHCP server mode.

`-r` *IP_address* | *hostname*, . . .
This option enables BOOTP relay agent mode. The option argument specifies a comma-separated list of IP addresses or hostnames of DHCP or BOOTP servers to which the relay agent is to forward BOOTP requests. When the daemon is started in this mode, any DHCP tables are ignored, and the daemon simply acts as a BOOTP relay agent.

A BOOTP relay agent listens to UDP port 68, and forwards BOOTP request packets received on this port to the destinations specified on the command line. It supports the `BROADCAST` flag described in RFC 1542. A BOOTP relay agent can run on any machine that has knowledge of local routers, and thus does not have to be an Internet gateway machine.

Note that the proper entries must be made to the `netmasks` database so that the DHCP server being served by the BOOTP relay agents can identify the subnet mask of the foreign BOOTP/DHCP client's network. See `netmasks`(4) for the format and use of this database.

`-t` *dhcptab_rescan_interval*
Specifies the interval in minutes that the DHCP server should use to schedule the automatic rereading of the `dhcptab` information. Typically, you would use this option if the changes to the `dhcptab` are relatively frequent. Once the contents of the `dhcptab` have stabilized, you can turn off this option to avoid needless reinitialization of the server.

`-v`
Verbose mode. The daemon displays more messages than in the default mode. Note that verbose mode can reduce daemon efficiency due to the time taken to display messages. Messages are displayed to the current TTY if the debugging option is used; otherwise, messages are logged to the `syslogd` facility. This option can be used in both DHCP/BOOTP server mode and BOOTP relay agent mode.

EXAMPLES

EXAMPLE 1 Starting a DHCP Server in BOOTP Compatibility Mode

The following command starts a DHCP server in BOOTP compatibility mode, permitting the server to automatically allocate permanent IP addresses to BOOTP clients which are not registered in the server's table; limits the server's attention to incoming datagrams on network devices `le2` and `tr0`; drops BOOTP packets whose hop count exceeds 2; configures the DHCP server to cache extended DHCP offers for 15 seconds; and schedules `dhcptab` rescans to occur every 10 minutes:

```
# in.dhcpd -i le2,tr0 -h 2 -o 15 -t 10 -b automatic
```

EXAMPLE 2 Starting the Daemon in BOOTP Relay Agent Mode

The following command starts the daemon in BOOTP relay agent mode, registering the hosts `bladerunner` and `10.0.0.5` as relay destinations, with debugging and verbose modes enabled, and drops BOOTP packets whose hop count exceeds 5:

```
# in.dhcpd -d -v -h 5 -r bladerunner,10.0.0.5
```

FILES

`/etc/inet/dhcpsvc.conf`

`/etc/init.d/dhcp`

`/etc/init/hosts`

`/usr/lib/inet/dhcp/nsu/rfc2136.so.1`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWdhcsu
Interface Stability	Evolving

SEE ALSO

`cron`(1M), `dhcpmgr`(1M), `dhtadm`(1M), `pntadm`(1M), `syslogd`(1M), `syslog`(3C), `dhcpsvc.conf`(4), `dhcp_network`(4), `dhcptab`(4), `ethers`(4), `hosts`(4), `netmasks`(4), `nsswitch.conf`(4), `attributes`(5), `dhcp`(5)

System Administration Guide: IP Services

Alexander, S., and R. Droms, *DHCP Options and BOOTP Vendor Extensions*, RFC 2132, Silicon Graphics, Inc., Bucknell University, March 1997.

Droms, R., *Interoperation Between DHCP and BOOTP*, RFC 1534, Bucknell University, October 1993.

Droms, R., *Dynamic Host Configuration Protocol*, RFC 2131, Bucknell University, March 1997.

Wimer, W., *Clarifications and Extensions for the Bootstrap Protocol*, RFC 1542, Carnegie Mellon University, October 1993.

NAME

inetd – Internet services daemon

SYNOPSIS

inetd [-d] [-s] [-t] [-r *count interval*] [*configuration-file*]

DESCRIPTION

inetd is the server process for the Internet standard services. It usually starts up at system boot time. The *configuration-file* lists the services that inetd is to provide. If no *configuration-file* is given on the command line, inetd reads the configuration information from the /etc/inetd.conf file. If /etc/inetd.conf is not present, inetd reads the configuration information from /etc/inet/inetd.conf. See inetd.conf(4) for more information on the format of this file.

inetd listens for service requests on the TCP or UDP ports associated with each of the services listed in the configuration file. When a request arrives, inetd executes the server program associated with the service.

A service can be configured to have "wait" wait-status, in which case, inetd waits for the server process to exit before starting a second server process. RPC services can also be started by inetd.

inetd provides a number of simple Internet services internally. These include echo, discard, chargen (character generator), daytime (human-readable time), and time (machine-readable time, in the form of the number of seconds since midnight, January 1, 1900).

inetd reads the *configuration-file* and the default settings in /etc/default/inetd once when it starts up and rereads them again whenever it receives a hangup signal, SIGHUP. New services can be activated and existing services can be deleted or modified by editing the *configuration-file* and then sending inetd a SIGHUP signal.

After it receives the SIGHUP signal, inetd reads the *configuration-file* and, for each service listed, attempts to bind() to that service's port. The attempt might fail if another standalone server or "wait" wait-status server started by inetd is already listening on the same port. Such a server has to be killed before inetd can bind to the service's port. inetd defers implementing a newly read configuration for a service whose port is busy and periodically attempts to start listening, after logging an error on console. The retry interval is currently 10 minutes.

If you want a "wait" wait-status server that is started by inetd to be controlled by that daemon following a kill and restart of inetd, you must do one of the following:

- Kill the server before restarting inetd.
- Restart inetd, kill the server, and wait till the retry interval elapses. After this time, inetd attempts to restart the server upon the next request for service.

The /etc/default/inetd file contains the following default parameter settings. See FILES.

ENABLE_CONNECTION_LOGGING — Specifies whether incoming TCP connections are traced. The value ENABLE_CONNECTION_LOGGING=YES is equivalent to the -t command-line option. The

	default value for ENABLE_CONNECTION_LOGGING is NO.
ENABLE_TCPWRAPPERS	Specifies the TCP wrappers facility will be used to control access to TCP services. The value YES enables checking. The default value for ENABLE_TCPWRAPPERS is NO. If the ENABLE_TCPWRAPPERS parameter is turned on, then all "streams, nowait" services will be automatically wrapped by the TCP wrappers facility. The stability level of the TCP wrappers facility and its configuration files is External. As the TCP wrappers facility is not controlled by Sun, intrarelease incompatibilities are not uncommon. See attributes(5).

For more information about configuring TCP wrappers, you can refer to the following man pages, which are delivered as part of Solaris at /usr/sfw/man: tcpd(1M), hosts_access(5).

OPTIONS

-d Runs inetd in the foreground and enables debugging output.

-s Allows you to run inetd "stand-alone'' outside the Service Access Facility (SAF). If the -s option is omitted, inetd will attempt to contact the service access controller (SAC) and will exit if SAC is not already running. See sac(1M).

-t Instructs inetd to trace the incoming connections for all of its TCP services. It does this by logging the client's IP address and TCP port number, along with the name of the service, using the syslog(3C) facility. "Wait" wait-status services cannot be traced. When tracing is enabled, inetd uses the syslog facility code "daemon'' and "notice'' priority level. See FILES.

-r Allows inetd to detect and then suspend "broken'' wait services servers and connectionless datagram services servers, for example, UDP and RPC/CLTS. Without this detection, a buggy server that fails before consuming the service request is continuously restarted and taxes system resources too much. The -r flag has the form:

-r *count interval*

count and *interval* are decimal numbers that represent the maximum *count* of invocations per *interval* of seconds a service can be started before the service is considered "broken.''

After being considered "broken,'' a server is suspended for ten minutes. After ten minutes, inetd again enables service, trusting the server to operate correctly.

If the -r flag is not specified, inetd considers -r40 60 to be specified.

OPERANDS

configuration-file Lists the services `inetd` is to provide.

EXIT STATUS

`inetd` does not return an exit status.

FILES

`/etc/default/inetd` Contains default settings. `inetd` reads the *configuration-file* and the default settings in `/etc/default/inetd` once when it starts up and rereads them again whenever it receives a hangup signal, `SIGHUP`. You can override some of the settings by command-line options.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`in.ftpd`(1M), `in.rexecd`(1M), `in.rshd`(1M), `in.tftpd`(1M), `sac`(1M), `syslog`(3C), `inetd.conf`(4), `attributes`(5)

Postel, Jon. *RFC 862: Echo Protocol*. Network Information Center, SRI International, Menlo Park, CA, May 1983.

Postel, Jon. *RFC 863: Discard Protocol*. Network Information Center, SRI International, Menlo Park, CA, May 1983.

Postel, Jon. *RFC 864: Character Generator Protocol*. Network Information Center, SRI International, Menlo Park, CA, May 1983.

Postel, Jon. *RFC 867: Daytime Protocol*. Network Information Center, SRI International, Menlo Park, CA, May 1983.

Postel, Jon, and Ken Harrenstien. *RFC 868: Time Protocol*. Network Information Center, SRI International, Menlo Park, CA, May 1983.

The following man pages are delivered as part of the SUNWtcpd package: `tcpd`(1M), `hosts_access`(4)

WARNINGS

Do not configure `udp` services as `nowait`. This can cause a race condition where the `inetd` program selects on the socket and the server program reads from the socket. Many server programs will fork and performance will be severely compromised.

If you kill and restart `inetd`, be aware that any environment variables in your shell are inherited by a shell for an incoming `telnet` session. For example, if you have `USER=root` in your environment, a user who connects to your machine with `telnet` inherits `USER=root`.

NOTES

For RPC services, `inetd` listens on all the transports, not only `tcp` and `udp`, as specified for each service in the `inetd.conf`(4) file.

NAME

in.fingerd, fingerd – remote user information server

SYNOPSIS

```
/usr/sbin/in.fingerd
```

DESCRIPTION

`fingerd` implements the server side of the Name/Finger protocol, specified in *RFC 742*. The Name/Finger protocol provides a remote interface to programs which display information on system status and individual users. The protocol imposes little structure on the format of the exchange between client and server. The client provides a single command line to the finger server which returns a printable reply.

`fingerd` waits for connections on TCP port 79. Once connected, it reads a single command line terminated by RETURN-LINEFEED and passes the arguments to `finger`(1), prepended with `-s`. `fingerd` closes its connections as soon as the output is finished.

You must invoke `fingerd` from `inetd`. See `inetd`(1M) for more information.

FILES

`/var/adm/utmpx`	User and accounting information.
`/etc/passwd`	System password file.
`/var/adm/lastlog`	Last login times.
`$HOME/.plan`	User's plans.
`$HOME/.project`	User's projects.

USAGE

`fingerd` and `in.fingerd` are IPv6–enabled. See `ip6`(7P).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWrcmds

SEE ALSO

`finger`(1), `inetd`(1M), `inetd.conf`(4), `attributes`(5), `ip6`(7P)

Harrenstien, Ken, *RFC 742, NAME/FINGER*, Network Information Center, SRI International, Menlo Park, Calif., December 1977.

NOTES

Connecting directly to the server from a TIP or an equally narrow-minded TELNET-protocol user program can result in meaningless attempts at option negotiation being sent to the server, which will foul up the command line interpretation. `fingerd` should be taught to filter out IAC's and perhaps even respond negatively (IAC will not) to all option commands received.

NAME

infocmp – compare or print out terminfo descriptions

SYNOPSIS

```
/usr/bin/infocmp [-d] [-c] [-n] [-I] [-L] [-C] [-r] [-u] [-s | d |
     i | l | c] [-v] [-V] [-1] [-w width] [-A  directory] [-B directory]
     [termname...]
```

DESCRIPTION

`infocmp` compares a binary `terminfo` entry with other terminfo entries, rewrites a `terminfo` description to take advantage of the `use=` terminfo field, or prints out a `terminfo` description from the binary file (`term`) in a variety of formats. It displays boolean fields first, then numeric fields, followed by the string fields. If no options are specified and zero, or one *termname* is specified, the `-I` option is assumed. If more than one *termname* is specified, the `-d` option is assumed.

OPTIONS

The `-d` , `-c` , and `-n` options can be used for comparisons. `infocmp` compares the `terminfo` description of the first terminal *termname* with each of the descriptions given by the entries for the other terminal's *termname*. If a capability is defined for only one of the terminals, the value returned will depend on the type of the capability: `F` for boolean variables, `–1` for integer variables, and `NULL` for string variables.

`-d` Produce a list of each capability that is different between two entries. This option is useful to show the difference between two entries, created by different people, for the same or similar terminals.

`-c` Produce a list of each capability that is common between two entries. Capabilities that are not set are ignored. This option can be used as a quick check to see if the `-u` option is worth using.

`-n` Produce a list of each capability that is in neither entry. If no *termname* is given, the environment variable `TERM` will be used for both of the *termname*s. This can be used as a quick check to see if anything was left out of a description.

The `-I` , `-L` , and `-C` options will produce a source listing for each terminal named.

`-I` Use the `terminfo` names.

`-L` Use the long C variable name listed in < `term.h` >.

`-C` Use the `termcap` names. The source produced by the `-C` option may be used directly as a `termcap` entry, but not all of the parameterized strings may be changed to the `termcap` format. `infocmp` will attempt to convert most of the parameterized information, but anything not converted will be plainly marked in the output and commented out. These should be edited by hand.

`-r` When using `-C` , put out all capabilities in `termcap` form.

If no *termname* is given, the environment variable `TERM` will be used for the terminal name.

All padding information for strings will be collected together and placed at the beginning of the string where `termcap` expects it. Mandatory padding (padding information with a trailing '/') will become optional.

All `termcap` variables no longer supported by `terminfo` , but are derivable from other `terminfo` variables, will be displayed. Not all `terminfo` capabilities will be translated; only those variables which were part of `termcap` will normally be displayed. Specifying the `-r` option will take off this restriction, allowing all capabilities to be displayed in `termcap` form.

Note that because padding is collected to the beginning of the capability, not all capabilities are displayed. Mandatory padding is not supported. Because `termcap` strings are not as flexible, it is not always possible to convert a `terminfo` string capability into an equivalent `termcap` format. A subsequent conversion of the `termcap` file back into `terminfo` format will not necessarily reproduce the original `terminfo` source.

Some common `terminfo` parameter sequences, their `termcap` equivalents, and some terminal types which commonly have such sequences, are:

```
terminfo     termcap      Representative Terminals
%p1%c    %.    adm
%p1%d    %d    hp, ANSI standard, vt100
%p1%'x'%+%c    %+x    concept
%i    %i    ANSI standard, vt100
%p1%?%'x'%>%t%p1%'y'%+%;    %>xy    concept
%p2 is printed before %p1    %r    hp
```

`-u` Produce a `terminfo` source description of the first terminal *termname* which is relative to the sum of the descriptions given by the entries for the other terminals' *termname*s. It does this by analyzing the differences between the first *termname* and the other *termnames* and producing a description with `use=` fields for the other terminals. In this manner, it is possible to retrofit generic `terminfo` entries into a terminal's description. Or, if two similar terminals exist, but were coded at different times, or by different people so that each description is a full description, using `infocmp` will show what can be done to change one description to be relative to the other.

A capability is displayed with an at-sign (@) if it no longer exists in the first *termname*, but one of the other *termname* entries contains a value for it. A capability's value is displayed if the value in the first *termname* is not found in any of the other *termname* entries, or if the first of the other *termname* entries that has this capability gives a different value for that capability.

The order of the other *termname* entries is significant. Since the `terminfo` compiler `tic` does a left-to-right scan of the capabilities, specifying two `use=` entries that contain differing entries for the same capabilities will produce different results, depending on the order in which the entries are given. `infocmp` will flag any such inconsistencies between the other *termname* entries as they are found.

Alternatively, specifying a capability *after* a `use=` entry that contains, it will cause the second specification to be ignored. Using `infocmp` to recreate a description can be a useful check to make sure that everything was specified correctly in the original source description.

Another error that does not cause incorrect compiled files, but will slow down the compilation time, is specifying superfluous `use=` fields. `infocmp` will flag any superfluous `use=` fields.

`-s` Sorts the fields within each type according to the argument below:

- `d` Leave fields in the order that they are stored in the `terminfo` database.
- `i` Sort by `terminfo` name.
- `l` Sort by the long C variable name.
- `c` Sort by the `termcap` name.

If the `-s` option is not given, the fields are sorted alphabetically by the `terminfo` name within each type, except in the case of the `-C` or the `-L` options, which cause the sorting to be done by the `termcap` name or the long C variable name, respectively.

`-v` Print out tracing information on standard error as the program runs.

`-V` Print out the version of the program in use on standard error and exit.

`–1` Print the fields one to a line. Otherwise, the fields are printed several to a line to a maximum width of 60 characters.

`-w`*width* Changes the output to *width* characters.

The location of the compiled `terminfo` database is taken from the environment variable `TERMINFO`. If the variable is not defined, or the terminal is not found in that location, the system `terminfo` database, usually in `/usr/share/lib/terminfo`, is used. The options `-A` and `-B` may be used to override this location.

`-A` *directory* Set `TERMINFO` for the first *termname*.

`-B` *directory* Set `TERMINFO` for the other *termname*s. With this, it is possible to compare descriptions for a terminal with the same name located in two different databases. This is useful for comparing descriptions for the same terminal created by different people.

FILES

`/usr/share/lib/terminfo/?/*` Compiled terminal description database.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`captoinfo`(1M), `tic`(1M), `curses`(3CURSES), `terminfo`(4), `attributes`(5)

NAME

in.ftpd, ftpd – File Transfer Protocol Server

SYNOPSIS

```
in.ftpd [-A] [-a] [-d] [-I] [-i] [-L] [-l] [-o] [-P dataport] [-p
     ctrlport] [-Q] [-q] [-r rootdir] [-S] [-s] [-T maxtimeout] [-t timeout]
     [-u umask] [-V] [-v] [-W] [-w] [-X]
```

DESCRIPTION

`in.ftpd` is the Internet File Transfer Protocol (FTP) server process. The server may be invoked by the Internet daemon `inetd`(1M) each time a connection to the FTP service is made or run as a standalone server. See `services`(4).

OPTIONS

`in.ftpd` supports the following options:

`-A` Disable use of the `ftpaccess`(4) file. Use of `ftpaccess` is disabled by default.

`-a` Enable use of the `ftpaccess`(4) file.

`-d` Write debugging information to `syslogd`(1M).

`-I` Disable the use of `AUTH` and `ident` to determine the username on the client. See *RFC 931*. The FTP Server is built not to use `AUTH` and `ident`.

`-i` Log the names of all files received by the FTP Server to `xferlog`(4). You can override the `-i` option through use of the `ftpaccess`(4) file.

`-L` Log all commands sent to `in.ftpd` to `syslogd`(1M). When the `-L` option is used, command logging will be on by default, once the FTP Server is invoked. Because the FTP Server includes `USER` commands in those logged, if a user accidentally enters a password instead of the username, the password will be logged. You can override the `-L` option through use of the `ftpaccess`(4) file.

`-l` Log each FTP session to `syslogd`(1M).

`-o` Log the names of all files transmitted by the FTP Server to `xferlog`(4). You can override the `-o` option through use of the `ftpaccess`(4) file.

`-P` *dataport* The FTP Server determines the port number by looking in the `services`(4) file for an entry for the `ftp-data` service. If there is no entry, the daemon uses the port just prior to the control connection port. Use the `-P` option to specify the data port number.

`-p` *ctrlport* When run in standalone mode, the FTP Server determines the control port number by looking in the `services`(4) file for an entry for the `ftp` service. Use the `-p` option to specify the control port number.

`-Q`	Disable `PID` files. This disables user limits. Large, busy sites that do not want to impose limits on the number of concurrent users can use this option to disable `PID` files.
`-q`	Use `PID` files. The `limit` directive uses `PID` files to determine the number of current users in each access class. By default, `PID` files are used.
`-r` *rootdir*	`chroot`(2) to *rootdir* upon loading. Use this option to improve system security. It limits the files that can be damaged should a break in occur through the daemon. This option is similar to anonymous FTP. Additional files are needed, which vary from system to system.
`-S`	Place the daemon in standalone operation mode. The daemon runs in the background. This is useful for startup scripts that run during system initialization. See `init.d`(4).
`-s`	Place the daemon in standalone operation mode. The daemon runs in the foreground. This is useful when run from `/etc/inittab` by `init`(1M).
`-T` *maxtimeout*	Set the maximum allowable timeout period to *maxtimeout* seconds. The default maximum timeout limit is 7200 second (two hours). You can override the `-T` option through use of the `ftpaccess`(4) file.
`-t` *timeout*	Set the inactivity timeout period to *timeout* seconds. The default timeout period is 900 seconds (15 minutes). You can override the `-t` option through use of the `ftpaccess`(4) file.
`-u` *umask*	Set the default `umask` to *umask*.
`-V`	Display copyright and version information, then terminate.
`-v`	Write debugging information to `syslogd`(1M).
`-W`	Do not record user `login` and `logout` in the `wtmpx`(4) file.
`-w`	Record each user `login` and `logout` in the `wtmpx`(4) file. By default, logins and logouts are recorded.
`-X`	Write the output from the `-i` and `-o` options to the `syslogd`(1M) file instead of `xferlog`(4). This allows the collection of output from several hosts on one central loghost. You can override the `-X` option through use of the `ftpaccess`(4) file.

Requests

The FTP Server currently supports the following FTP requests; case is not distinguished.

`ABOR`	Abort previous command.
`ALLO`	Allocate storage (vacuously).

`APPE`	Append to a file.
`CDUP`	Change to parent of current working directory.
`CWD`	Change working directory.
`DELE`	Delete a file.
`EPRT`	Specify extended address for the transport connection.
`EPSV`	Extended passive command request.
`HELP`	Give help information.
`LIST`	Give list files in a directory (`ls -lA`).
`LPRT`	Specify long address for the transport connection.
`LPSV`	Long passive command request.
`MKD`	Make a directory.
`MDTM`	Show last time file modified.
`MODE`	Specify data transfer *mode*.
`NLST`	Give name list of files in directory (`ls`).
`NOOP`	Do nothing.
`PASS`	Specify password.
`PASV`	Prepare for server-to-server transfer.
`PORT`	Specify data connection port.
`PWD`	Print the current working directory.
`QUIT`	Terminate session.
`REST`	Restart incomplete transfer.
`RETR`	Retrieve a file.
`RMD`	Remove a directory.
`RNFR`	Specify rename-from file name.
`RNTO`	Specify rename-to file name.
`SITE`	Use nonstandard commands.
`SIZE`	Return size of file.
`STAT`	Return status of server.
`STOR`	Store a file.
`STOU`	Store a file with a unique name.
`STRU`	Specify data transfer *structure*.

SYST	Show operating system type of server system.
TYPE	Specify data transfer type.
USER	Specify user name.
XCUP	Change to parent of current working directory. This request is deprecated.
XCWD	Change working directory. This request is deprecated.
XMKD	Make a directory. This request is deprecated.
XPWD	Print the current working directory. This request is deprecated.
XRMD	Remove a directory. This request is deprecated.

The following nonstandard or UNIX specific commands are supported by the SITE request:

ALIAS	List aliases.
CDPATH	List the search path used when changing directories.
CHECKMETHOD	List or set the checksum method.
CHECKSUM	Give the checksum of a file.
CHMOD	Change mode of a file. For example, SITE CHMOD 755 *filename*.
EXEC	Execute a program. For example, SITE EXEC program params
GPASS	Give special group access password. For example, SITE GPASS bar.
GROUP	Request special group access. For example, SITE GROUP foo.
GROUPS	List supplementary group membership.
HELP	Give help information. For example, SITE HELP.
IDLE	Set idle-timer. For example, SITE IDLE 60.
UMASK	Change umask. For example, SITE UMASK 002.

The remaining FTP requests specified in *RFC 959* are recognized, but not implemented.

The FTP server will abort an active file transfer only when the ABOR command is preceded by a Telnet "Interrupt Process" (IP) signal and a Telnet "Synch" signal in the command Telnet stream, as described in *RFC 959*. If a STAT command is received during a data transfer that has been preceded by a Telnet IP and Synch, transfer status will be returned.

`in.ftpd` interprets file names according to the "globbing" conventions used by `csh`(1). This allows users to utilize the metacharacters: `* ? [ ] { } ~`

`in.ftpd` authenticates users according to four rules.

First, the user name must be in the password data base, `/etc/passwd`. The password must not be null. A password must always be provided by the client before any file operations may be performed. The PAM framework is used to verify that the correct password was entered. See `SECURITY` below.

Second, the user name must not appear in either the `/etc/ftpusers` or the `/etc/ftpd/ftpusers` file. Use of the `/etc/ftpusers` files is deprecated, although it is still supported.

Third, the uses must have a standard shell returned by `getusershell`(3C).

Fourth, if the user name is `anonymous` or `ftp`, an anonymous ftp account must be present in the password file for user `ftp`. Use `ftpconfig`(1M) to create the anonymous `ftp` account and home directory tree.

The FTP Server supports virtual hosting, which can be configured by using `ftpaddhost`(1M).

The FTP Server does not support sublogins.

General FTP Extensions

The FTP Server has certain extensions. If the user specifies a filename that does not exist with a `RETR` (retrieve) command, the FTP Server looks for a conversion to change a file or directory that does into the one requested. See `ftpconversions`(4).

By convention, anonymous users supply their email address when prompted for a password. The FTP Server attempts to validate these email addresses. A user whose FTP client hangs on a long reply, for example, a multiline response, should use a dash (-) as the first character of the user's password, as this disables the Server's `lreply()` function.

The FTP Server can also log all file transmission and reception. See `xferlog`(4) for details of the log file format.

The `SITE EXEC` command may be used to execute commands in the `/bin/ftp-exec` directory. Take care that you understand the security implications before copying any command into the `/bin/ftp-exec` directory. For example, do not copy in `/bin/sh`. This would enable the user to execute other commands through the use of `sh -c`. If you have doubts about this feature, do not create the `/bin/ftp-exec` directory.

SECURITY

`in.ftpd` uses `pam`(3PAM) for authentication, account management, and session management. The PAM configuration policy, listed through `/etc/pam.conf`, specifies the module to be used for `in.ftpd`. Here is a partial `pam.conf` file with entries for the `in.ftpd` command using the UNIX authentication, account management, and session management module.

```
ftp  auth        requisite   pam_authtok_get.so.1
ftp  auth        required    pam_dhkeys.so.1
ftp  auth        required    pam_unix_auth.so.1

ftp  account     required    pam_unix_roles.so.1
ftp  account     required    pam_unix_projects.so.1
ftp  account     required    pam_unix_account.so.1

ftp  session     required    pam_unix_session.so.1
```

If there are no entries for the `ftp` service, then the entries for the "other" service will be used. Unlike `login`, `passwd`, and other commands, the `ftp` protocol will only support a single password. Using multiple modules will prevent `in.ftpd` from working properly.

USAGE

The `in.ftpd` command is IPv6–enabled. See `ip6`(7P).

FILES

`/etc/ftpd/ftpaccess`	FTP Server configuration file
`/etc/ftpd/ftpconversions`	FTP Server conversions database
`/etc/ftpd/ftpgroups`	FTP Server enhanced group access file
`/etc/ftpd/ftphosts`	FTP Server individual user host access file
`/etc/ftpd/ftpservers`	FTP Server virtual hosting configuration file.
`/etc/ftpd/ftpusers`	File listing users for whom FTP login privileges are disallowed.
`/etc/ftpusers`	File listing users for whom FTP login privileges are disallowed. This use of this file is deprecated.
`/var/log/xferlog`	FTP Server transfer log file
`/var/run/ftp.pids-`*classname*	
`/var/adm/wtmpx`	Extended database files that contain the history of user access and accounting information for the `wtmpx` database.

ATTRIBUTES

See `attributes` (5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWftpu

SEE ALSO

`csh`(1), `ftp`(1), `ftpcount`(1), `ftpwho`(1), `ls`(1), `ftpaddhost`(1M), `ftpconfig`(1M), `ftprestart`(1M), `ftpshut`(1M), `inetd`(1M), `syslogd`(1M), `chroot`(2), `umask`(2), `getpwent`(3C), `getusershell`(3C), `syslog`(3C), `ftpaccess`(4), `ftpconversions`(4), `ftpgroups`(4), `ftphosts`(4), `ftpservers`(4), `ftpusers`(4),

group(4), passwd(4), services(4), xferlog(4), wtmpx(4), attributes(5), pam_authtok_check(5), pam_authtok_get(5), pam_authtok_store(5), pam_dhkeys(5), pam_passwd_auth(5), pam_unix(5), pam_unix_account(5), pam_unix_auth(5), pam_unix_session(5)ip6(7P)

Allman, M., Ostermann, S., and Metz, C. *RFC 2428, FTP Extensions for IPv6 and NATs*. The Internet Society. September 1998.

Piscitello, D. *RFC 1639, FTP Operation Over Big Address Records (FOOBAR)*. Network Working Group. June 1994.

Postel, Jon, and Joyce Reynolds. *RFC 959, File Transfer Protocol (FTP)*. Network Information Center. October 1985.

St. Johns, Mike. *RFC 931, Authentication Server*. Network Working Group. January 1985.

DIAGNOSTICS

in.ftpd logs various errors to syslogd(1M), with a facility code of daemon.

NOTES

The anonymous FTP account is inherently dangerous and should be avoided when possible.

The FTP Server must perform certain tasks as the superuser, for example, the creation of sockets with privileged port numbers. It maintains an effective user ID of the logged in user, reverting to the superuser only when necessary.

The FTP Server no longer supports the /etc/default/ftpd file. Instead of using UMASK=nnn to set the umask, use the defumask capability in the ftpaccess file. The banner greeting text capability is also now set through the ftpaccess file by using the greeting text capability instead of by using BANNER="...". However, unlike the BANNER string, the greeting text string is not passed to the shell for evaluation. See ftpaccess(4).

The pam_unix(5) module might not be supported in a future release. Similar functionality is provided by pam_authtok_check(5), pam_authtok_get(5), pam_authtok_store(5), pam_dhkeys(5), pam_passwd_auth(5), pam_unix_account(5), pam_unix_auth(5), and pam_unix_session(5).

NAME

in.iked – daemon for the Internet Key Exchange (IKE)

SYNOPSIS

/usr/lib/inet/in.iked [-d] [-f *filename*] [-p *level*]

/usr/lib/inet/in.iked -c [-f *filename*]

DESCRIPTION

in.iked performs automated key management for IPsec using the Internet Key Exchange (IKE) protocol.

in.iked implements the following:

- IKE authentication with either pre-shared keys, DSS signatures, RSA signatures, or RSA encryption.
- Diffie-Hellman key derivation using either 768, 1024, or 1536-bit public key moduli.
- Authentication protection with cipher choices of DES, Blowfish, or 3DES, and hash choices of either HMAC-MD5 or HMAC-SHA-1. Encryption in in.iked is limited to the IKE authentication and key exchange. See ipsecesp(7P) for information regarding IPsec protection choices.

in.iked starts at boot time if the /etc/inet/ike/config file exists. See ike.config(4) for the format of this file.

in.iked listens for incoming IKE requests from the network and for requests for outbound traffic using the PF_KEY socket. See pf_key(7P).

in.iked has two support programs that are used for IKE administration and diagnosis: ikeadm(1M) and ikecert(1M).

IKE only works for IPv4.

The SIGHUP signal causes the IKE daemon to read /etc/inet/ike/config and reload the certificate database. SIGHUP is equivalent to using ikeadm(1M) to read the /etc/inet/ike/config file as a rule, for example:

```
example# ikeadm read rule /etc/inet/ike/config
```

OPTIONS

The following options are supported:

-c	Check the syntax of a configuration file.
-d	Use debug mode. The process stays attached to the controlling terminal and produces large amounts of debugging output.
-f *filename*	Use *filename* instead of /etc/inet/ike/config. See ike.config(4) for the format of this file.
-p *level*	Specify privilege level (*level*). This option sets how much ikeadm(1M) invocations can change or observe about the running in.iked. Valid *levels* are:

0 Base level

1 Access to preshared key info

2 Access to keying material

If `-p` is not specified, *level* defaults to `0`.

SECURITY

This program has sensitive private keying information in its image. Care should be taken with any core dumps or system dumps of a running `in.iked`- as these files contain sensitive keying information. Use the `coreadm`(1M) command to limit any corefiles produced by `in.iked`.

FILES

`/etc/inet/ike/config`

`/etc/inet/secret/ike.privatekeys/*`
Private keys. A private key *must* have a matching public-key certificate with the same filename in `/etc/inet/ike/publickeys/`.

`/etc/inet/ike/publickeys/*`
Public-key certificates. The names are only important with regard to matching private key names.

`/etc/inet/ike/crls/*`
Public key certificate revocation lists.

`/etc/inet/secret/ike.preshared`
IKE pre-shared secrets for Phase I authentication.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`coreadm`(1M), `ikeadm`(1M), `ikecert`(1M), `ike.config`(4), `attributes`(5), `ipsecesp`(7P)

Harkins, Dan and Carrel, Dave. *RFC 2409, Internet Key Exchange (IKE)*. Network Working Group. November 1998.

Maughan, Douglas, Schertler, M., Schneider, M., Turner, J. *RFC 2408, Internet Security Association and Key Management Protocol (ISAKMP)*. Network Working Group. November 1998.

Piper, Derrell, *RFC 2407, The Internet IP Security Domain of Interpretation for ISAKMP*. Netwrok Working Group. November 1998.

NAME

init, telinit – process control initialization

SYNOPSIS

/sbin/init [0123456abcQqSs]

/etc/telinit [0123456abcQqSs]

DESCRIPTION

`init` is a general process spawner. Its primary role is to create processes from information stored in the file `/etc/inittab`.

Run Level Defined

At any given time, the system is in one of eight possible run levels. A run level is a software configuration under which only a selected group of processes exists. Processes spawned by `init` for each of these run levels are defined in `/etc/inittab`. `init` can be in one of eight run levels, `0–6` and `S` or `s` (`S` and `s` are identical). The run level changes when a privileged user runs `/sbin/init`. This sends appropriate signals to the original `init` spawned by the operating system at boot time, saying which run level to invoke.

init and System Booting

When the system is booted, `init` is invoked and the following occurs. First, it reads `/etc/default/init` to set environment variables. This is typically where `TZ` (time zone) and locale-related environments such as `LANG` or `LC_CTYPE` get set. (See the FILES section at the end of this page.) `init` then looks in `/etc/inittab` for the `initdefault` entry (see `inittab`(4)). If the `initdefault` entry:

exists — `init` usually uses the run level specified in that entry as the initial run level to enter.

does not exist — `/etc/inittab`, `init` asks the user to enter a run level from the system console.

- `S` or `s` — `init` goes to the single-user state. In this state, the system console device (`/dev/console`) is opened for reading and writing and the command `/sbin/su`, (see `su`(1M)), is invoked. Use either `init` or `telinit` to change the run level of the system. Note that if the shell is terminated (using an end-of-file), `init` only re-initializes to the single-user state if `/etc/inittab` does not exist.
- `0-6` — `init` enters the corresponding run level. Run levels `0`, `5`, and `6` are reserved states for shutting the system down. Run levels `2`, `3`, and `4` are available as multi-user operating states.

If this is the first time since power up that `init` has entered a run level other than single-user state, `init` first scans `/etc/inittab` for `boot` and `bootwait` entries (see `inittab`(4)). These entries are performed before any other processing of `/etc/inittab` takes place, providing that the run level entered matches that of the

entry. In this way any special initialization of the operating system, such as mounting file systems, can take place before users are allowed onto the system. `init` then scans `/etc/inittab` and executes all other entries that are to be processed for that run level.

To spawn each process in `/etc/inittab`, `init` reads each entry and for each entry that should be respawned, it forks a child process. After it has spawned all of the processes specified by `/etc/inittab`, `init` waits for one of its descendant processes to die, a `powerfail` signal, or a signal from another `init` or `telinit` process to change the system's run level. When one of these conditions occurs, `init` re-examines `/etc/inittab`.

inittab Additions

New entries can be added to `/etc/inittab` at any time; however, `init` still waits for one of the above three conditions to occur before re-examining `/etc/inittab`. To get around this, `init Q` or `init q` command wakes `init` to re-examine `/etc/inittab` immediately.

When `init` comes up at boot time and whenever the system changes from the single-user state to another run state, `init` sets the `ioctl`(2) states of the console to those modes saved in the file `/etc/ioctl.syscon`. `init` writes this file whenever the single-user state is entered.

Run Level Changes

When a run level change request is made, `init` sends the warning signal (`SIGTERM`) to all processes that are undefined in the target run level. `init` waits five seconds before forcibly terminating these processes by sending a kill signal (`SIGKILL`).

When `init` receives a signal telling it that a process it spawned has died, it records the fact and the reason it died in `/var/adm/utmpx` and `/var/adm/wtmpx` if it exists (see `who`(1)). A history of the processes spawned is kept in `/var/adm/wtmpx`.

If `init` receives a `powerfail` signal (`SIGPWR`) it scans `/etc/inittab` for special entries of the type `powerfail` and `powerwait`. These entries are invoked (if the run levels permit) before any further processing takes place. In this way `init` can perform various cleanup and recording functions during the powerdown of the operating system.

Environment Variables in /etc/defaults/init

You can set default values for environment variables, for such items as timezone and character formatting, in `/etc/default/init`. See the FILES section, below, for a list of these variables.

telinit

`telinit`, which is linked to `/sbin/init`, is used to direct the actions of `init`. It takes a one-character argument and signals `init` to take the appropriate action.

SECURITY

`init` uses `pam`(3PAM) for session management. The PAM configuration policy, listed through `/etc/pam.conf`, specifies the session management module to be used for `init`. Here is a partial `pam.conf` file with entries for `init` using the UNIX session management module.

```
init   session   required    pam_unix_session.so.1
```

If there are no entries for the `init` service, then the entries for the "other" service will be used.

OPTIONS

`0`	Go into firmware.
`1`	Put the system in system administrator mode. All local file systems are mounted. Only a small set of essential kernel processes are left running. This mode is for administrative tasks such as installing optional utility packages. All files are accessible and no users are logged in on the system.
`2`	Put the system in multi-user mode. All multi-user environment terminal processes and daemons are spawned. This state is commonly referred to as the multi-user state.
`3`	Extend multi-user mode by making local resources available over the network.
`4`	Is available to be defined as an alternative multi-user environment configuration. It is not necessary for system operation and is usually not used.
`5`	Shut the machine down so that it is safe to remove the power. Have the machine remove power, if possible.
`6`	Stop the operating system and reboot to the state defined by the `initdefault` entry in `/etc/inittab`.
`a, b, c`	process only those `/etc/inittab` entries having the `a`, `b`, or `c` run level set. These are pseudo-states, which may be defined to run certain commands, but which do not cause the current run level to change.
`Q, q`	Re-examine `/etc/inittab`.
`S, s`	Enter single-user mode. This is the only run level that doesn't require the existence of a properly formatted `/etc/inittab` file. If this file does not exist, then by default, the only legal run level that `init` can enter is the single-user mode. When in single-user mode, the filesystems required for basic system operation will be mounted. When the system comes down to single-user mode, these file systems will remain mounted (even if provided by a remote file server), and any other local filesystems will also be left mounted. During the transition down to single-user mode, all processes started by `init` or `init.d` scripts that should only be running in multi-user mode are killed. In addition, any process that has a `utmpx` entry will be killed. This last condition insures that all port monitors started by the SAC are killed and all services started by these port monitors, including `ttymon` login services, are killed.

FILES

`/dev/console`	System console device.

`/etc/default/init` Contains environment variables and their default values. For example, for the timezone variable, `TZ`, you might specify `TZ=US/Pacific`. The variables are:

`TZ`	Either specifies the timezone information (see `ctime`(3C)) or the name of a timezone information file `/usr/share/lib/zoneinfo`.
`CMASK`	The mask (see `umask`(1)) that `init` uses and that every process inherits from the `init` process. If not set, `init` uses the mask it inherits from the kernel. Note that `init` always attempts to apply a `umask` of 022 before creating a file, regardless of the setting of `CMASK`.
`LC_CTYPE`	Character characterization information.
`LC_MESSAGES`	Message translation.
`LC_MONETARY`	Monetary formatting information.
`LC_NUMERIC`	Numeric formatting information.
`LC_TIME`	Time formatting information.
`LC_ALL`	If set, all other `LC_*` environmental variables take-on this value.
`LANG`	If `LC_ALL` is not set, and any particular `LC_*` is also not set, the value of `LANG` is used for that particular environmental variable.

`/etc/initpipe` A named pipe used for internal communication.

`/etc/inittab` Controls process dispatching by `init`.

`/etc/ioctl.syscon` ioctl states of the console, as saved by `init` when single-user state is entered.

`/var/adm/utmpx` User access and administration information.

`/var/adm/wtmpx` History of user access and administration information.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO | login(1), sh(1), stty(1), who(1), shutdown(1M), su(1M), ttymon(1M), ioctl(2), kill(2), ctime(3C), pam(3PAM), inittab(4), pam.conf(4), utmpx(4), attributes(5), pam_authtok_check(5), pam_authtok_get(5), pam_authtok_store(5), pam_dhkeys(5), pam_passwd_auth(5), pam_unix(5), pam_unix_account(5), pam_unix_auth(5), pam_unix_session(5), termio(7I)

DIAGNOSTICS | If init finds that it is respawning an entry from /etc/inittab more than ten times in two minutes, assumes that there is an error in the command string in the entry, and generates an error message on the system console. It will then refuse to respawn this entry until either five minutes has elapsed or it receives a signal from a user-spawned init or telinit. This prevents init from eating up system resources when someone makes a typographical error in the inittab file, or a program is removed that is referenced in /etc/inittab.

NOTES | init and telinit can be run only by a privileged user.

The S or s state must not be used indiscriminately in /etc/inittab. When modifying this file, it is best to avoid adding this state to any line other than initdefault.

If a default state is not specified in the initdefault entry in /etc/inittab, state 6 is entered. Consequently, the system will loop by going to firmware and rebooting continuously.

If the utmpx file cannot be created when booting the system, the system will boot to state "s" regardless of the state specified in the initdefault entry in /etc/inittab. This can occur if the /var file system is not accessible.

When a system transitions down to the S or s state, the /etc/nologin file (see nologin(4)) is created. Upon subsequent transition to run level 2, this file is removed by a script in the /etc/rc2.d directory.

init uses /etc/initpipe, a named pipe, for internal communication.

The pam_unix(5) module might not be supported in a future release. Similar functionality is provided by pam_authtok_check(5), pam_authtok_get(5), pam_authtok_store(5), pam_dhkeys(5), pam_passwd_auth(5), pam_unix_account(5), pam_unix_auth(5), and pam_unix_session(5).

NAME

init.wbem – start and stop the CIM Boot Manager

SYNOPSIS

```
/etc/init.d/init.wbem start | stop | status
```

DESCRIPTION

The `init.wbem` utility is run automatically during installation and each time the system is rebooted. This utility manipulates the CIM Object Manager (CIMOM) and the Solaris Management Console server, both of which run combined in a single process. `init.wbem` can be used to start, stop, or retrieve status from the server.

CIM Object Manager

The CIM Object Manager manages CIM objects on a WBEM-enabled system. A CIM object is a computer representation, or model, of a managed resource, such as a printer, disk drive, or CPU. CIM objects are stored internally as Java classes.

When a WBEM client application accesses information about a CIM object, the CIM Object Manager contacts either the appropriate provider for that object or the CIM Object Manager Repository. Providers are classes that communicate with managed objects to access data.

When a WBEM client application requests data from a managed resource that is not available from the CIM Object Manager Repository, the CIM Object Manager forwards the request to the provider for that managed resource. The provider dynamically retrieves the information.

At startup, the CIM Object Manager performs the following functions:

- Listens for RMI connections on RMI port 5987 and for XML/HTTP connections on HTTP port 5988.
- Sets up a connection to the CIM Object Manager Repository.
- Waits for incoming requests.

During normal operations, the CIM Object Manager performs the following functions:

- Performs security checks to authenticate user login and authorization to access namespaces.
- Performs syntactical and semantic checking of CIM data operations to ensure that they comply with the latest CIM Specification.
- Routes requests to the appropriate provider or to the CIM Object Manager Repository.
- Delivers data from providers and from the CIM Object Manager Repository to WBEM client applications.

A WBEM client application contacts the CIM Object Manager to establish a connection when it needs to perform WBEM operations, such as creating a CIM class or updating a CIM instance. When a WBEM client application connects to a CIM Object Manager, it gets a reference to the CIM Object Manager, which it then uses to request services and operations.

Solaris Management Console Server

The Solaris Management Console server is the back end to the front end console, `smc`(1M). It provides tools for the console to download and performs common services for the console and its tools to use, such as authentication, authorization, logging, messaging, and persistence.

System Booting

The `init.wbem` script is installed in the `/etc/init.d` directory.

OPTIONS

The following options are supported:

`start`	Starts the CIMOM and Solaris Management Console server on the local host.
`stop`	Stops the CIMOM and Solaris Management Console server on the local host.
`status`	Gets the status of the CIMOM and Solaris Management Console server on the local host.

NOTES

When the `init.wbem` script is run, it does not run the CIMOM and Solaris Management Console server directly. The server process is in Java and is too heavyweight to be run immediately at system boot time. Instead, three lightweight processes listen on three different ports that the CIMOM and the Solaris Management Console server normally use. This acts similarly to `inetd`(1M).

Because Java programs cannot inherit file descriptors as other programs can, there is a small time period from when the first connection is made until the server is fully operational where client connections may be dropped. WBEM clients are immune to this, as they will retry until the server comes online. Solaris Management Console clients are not immune, and it may be necessary to manually reconnect, though this should not happen in the common case.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWwbcor

SEE ALSO

`inetd`(1M), `mofcomp`(1M), `smc`(1M), `smcconf`(1M), `wbemadmin`(1M), `wbemlogviewer`(1M), `attributes`(5)

NAME

in.lpd – BSD print protocol adaptor

SYNOPSIS

```
/usr/lib/print/in.lpd
```

DESCRIPTION

`in.lpd` implements the network listening service for the BSD print protocol specified in RFC 1179. The BSD print protocol provides a remote interface for systems to interact with a local spooling system. The protocol defines five standard requests from the client to the server: starting queue processing, transfering print jobs, retrieving terse status, retrieving verbose status, and canceling print jobs.

`in.lpd` is started from `inetd` (see `inetd`(1M)). `inetd` waits for connections on TCP port 515. Upon receipt of a connect request, `in.lpd` is started to service the connection. Once the request has been filled, `in.lpd` closes the connection and exits.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

non-zero An error occurred.

FILES

`/etc/printers.conf`
System printer configuration database.

`printers.conf.byname`
NIS version of `/etc/printers.conf`.

`fns.ctx_dir.`*domain*
NIS+ version of `/etc/printers.conf`.

`/usr/lib/print/bsd-adaptor/bsd_*.so*`
Spooler translation modules.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpcu

SEE ALSO

`inetd`(1M), `printers.conf`(4), `attributes`(5)

NAME

in.mpathd – daemon for network adapter (NIC) failure detection, recovery, automatic failover and failback

SYNOPSIS

/sbin/in.mpathd

DESCRIPTION

The `in.mpathd` daemon performs Network Interface Card (NIC) failure and repair detection. In the event of a NIC failure, it causes IP network access from the failed NIC to failover to a standby NIC, if available, or to any another operational NIC that has been configured as part of the same network multipathing group. Once the failed NIC is repaired, all network access will be restored to the repaired NIC.

`in.mpathd` needs a special test address on each NIC for the purpose of sending and receiving probes on each NIC. Use the `ifconfig` command `-failover` option to configure these test addresses. See `ifconfig`(1M). The test address must belong to a subnet that is known to the hosts and routers on the link.

The `in.mpathd` daemon can detect NIC failure and repair by two methods, by sending and receiving ICMP echo requests and replies on each NIC, and by monitoring the `IFF_RUNNING` flag for each NIC. The link state on some models of NIC is indicated by the `IFF_RUNNING` flag, allowing for faster failure detection when the link goes down. The `in.mpathd` daemon considers a NIC to have failed if either of the above two methods indicates failure. A NIC is considered to be repaired only if both methods indicate the NIC is repaired.

The `in.mpathd` daemon sends the ICMP echo request probes to on-link routers. If no routers are available, it sends the probes to neighboring hosts. Thus, for network failure detection and repair, there must be at least one neighbor on each link that responds to ICMP echo request probes.

`in.mpathd` works on both IPv4 and IPv6. If IPv4 is plumbed on a NIC, an IPv4 test address is configured on theNIC, and the NIC is configured as part of a network multipathing group, then `in.mpathd` will start sending ICMP probes on the NIC using IPv4.

In the case of IPv6, the link-local address must be configured as the test address. The `in.mpathd` daemon will not accept a non-link-local address as a test address. If the NIC is part of a multipathing group, and the test address has been configured, then `in.mpathd` will probe the NIC for failures using IPv6.

Even if both the IPv4 and IPv6 protocol streams are plumbed, it is sufficient to configure only one of the two, that is, either an IPv4 test address or an IPv6 test address on a NIC. If only an IPv4 test address is configured, it probes using only ICMPv4. If only an IPv6 test address is configured, it probes using only ICMPv6. If both type test addresses are configured, it probes using both ICMPv4 and ICMPv6.

The `in.mpathd` daemon accesses three variable values in `/etc/default/mpathd`: *FAILURE_DETECTION_TIME*, *FAILBACK* and *TRACK_INTERFACES_ONLY_WITH_GROUPS*.

The *FAILURE_DETECTION_TIME* variable specifies the NIC failure detection time for the ICMP echo request probe method of detecting NIC failure. The shorter the failure detection time, the greater the volume of probe traffic. The default value of *FAILURE_DETECTION_TIME* is 10 seconds. This means that NIC failure will be detected by in.mpathd within 10 seconds. NIC failures detected by the IFF_RUNNING flag being cleared are acted on as soon as the in.mpathd daemon notices the change in the flag. The NIC repair detection time cannot be configured; however, it is defined as double the value of *FAILURE_DETECTION_TIME*.

By default, in.mpathd does failure detection only on NICs that are configured as part of a multipathing group. You can set *TRACK_INTERFACES_ONLY_WITH_GROUPS* to no to enable failure detection by in.mpathd on all NICs, even if they are not part of a multipathing group. However, in.mpathd cannot do failover from a failed NIC if it is not part of a multipathing group.

The in.mpathd daemon will restore network traffic back to the previously failed NIC, after it has detected a NIC repair. To disable this, set the value of *FAILBACK* to no in /etc/default/mpathd.

FILES

/etc/default/mpathd — Contains default values used by the in.mpathd daemon.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsr

SEE ALSO

ifconfig(1M), attributes(5), icmp(7P), icmp6(7P),

System Administration Guide, Volume 3

DIAGNOSTICS

IFF_NOFAILOVER address *address* is not unique and failure detection is not possible

Every test address that is configured must be unique on the system. Otherwise in.mpathd cannot do failure detection. Since the IPv6 test address is a link-local address, which in turn is derived from the ethernet address, each NIC must have a unique MAC address.

NIC *interface_name* of group *group_name* is not plumbed for IPv[4|6] and may affect failover capability

All NICs in a multipathing group must be homogeneously plumbed. For example, if a NIC is plumbed for IPv4, then all NICs in the group must be plumbed for IPv4. The streams modules pushed on all NICs must be identical.

`Failures cannot be detected on` *interface_name* `as no IFF_NOFAILOVER address is available`
Every NIC that is configured as part of a multipathing group must have a test address, which can be either IPv4 or IPv6, or both. Otherwise `in.mpathd` cannot probe the NIC, and this message is displayed.

`Invalid failure detection` *time* `assuming default 10000`
An invalid value was encountered for *FAILURE_DETECTION_TIME* in the `/etc/default/mpathd` file.

`Too small failure detection time of` *time* `assuming minimum 100`
The minimum value that can be specified for *FAILURE_DETECTION_TIME* is currently 100 milliseconds.

`Invalid value for FAILBACK` *value*
Valid values for the boolean variable *FAILBACK* are `yes` or `no`.

`Invalid value for TRACK_INTERFACES_ONLY_WITH_GROUPS` *value*
Valid values for the boolean variable *TRACK_INTERFACES_ONLY_WITH_GROUPS* are `yes` or `no`.

`Cannot meet requested failure detection time of` *time* `ms on (inet[6]` *interface_name*`) new failure detection is` *time* `ms`
The round trip time for ICMP probes is higher than the specified failure detection time. The network is probably congested or the probe targets are loaded. `in.mpathd` automatically increases the failure detection time to whatever it can achieve under these conditions.

`Improved failure detection time` *time* `ms on (inet[6]` *interface_name*`)`
The round trip time for ICMP probes has now decreased and `in.mpathd` has lowered the failure detection time correspondingly.

`NIC failure detected on` *interface_name*
`in.mpathd` has detected NIC failure on *interface_name*, and has set the `IFF_FAILED` flag on NIC *interface_name*.

`Successfully failed over from NIC` *interface_name1* `to NIC` *interface_name2*
`in.mpathd` has caused the network traffic to failover from NIC *interface_name1* to NIC *interface_name2*, which is part of the multipathing group.

`NIC repair detected on` *interface_name*
`in.mpathd` has detected that NIC *interface_name* is repaired and operational. If the `IFF_FAILED` flag on the NIC was previously set, it will be reset.

`Successfully failed back to NIC` *interface_name*
`in.mpathd` has restored network traffic back to NIC *interface_name*, which is now repaired and operational.

`The link has gone down on` *interface_name*
`in.mpathd` has detected that the `IFF_RUNNING` flag for NIC *interface_name* has been cleared, indicating the link has gone down.

`The link has come up on` *interface_name*
`in.mpathd` has detected that the `IFF_RUNNING` flag for NIC *interface_name* has been set, indicating the link has come up.

NAME

in.named, named – Internet domain name server (DNS)

SYNOPSIS

in.named [-d *debuglevel*] [-p *port#*] [[-b | -c] *config_file*] [-fqrv] [-u *user_name*] [-g *group_name*] [-t *directory*] [-w *directory*] [*config_file*]

DESCRIPTION

in.named is the Internet domain name server. For more information on the Internet name-domain system, see *RFC 1033*, *RFC 1034* and *RFC 1035*.

When run without any arguments, in.named reads the default configuration file /etc/named.conf, reads any initial data, and listens for queries. If you give a *config_file* argument at the end of the command line, it will override any *config_file* that you specified by using the -b or -c options.

The named.conf(4) configuration file controls some of the options and behavior for in.named.

OPTIONS

in.named supports the following options:

-b|c *configfile*
: Use an alternative configuration file. The default value for *configfile* is /etc/named.conf. The *configfile*, if any, specified at the end of the command line, overrides any value specified by this argument.

-d *debuglevel*
: Print debugging information. The value of *debuglevel* determines the level of messages to print. If negative, *debuglevel* is set to 1.

  The new debugging framework is considerably more sophisticated than in previous versions of in.named. The logging statement in the configuration file allows for multiple, distinct levels of debugging for each of a large set of categories of events, for example, for queries and transfers in and out. See named.conf(4).

-f
: Run this process in the foreground. The process will not fork(2). By default, in.named runs as a daemon in the background.

-g *group_name*
: Specify the group the server should run as after it initializes. The value specified may be either a group name or a numeric group id.

-p *port#*
: Use the specified remote port number. in.named will send queries to this port number. By default, the value is the standard port number, that is, the port number returned by getservbyname(3SOCKET) for the service domain.

  The in.named command previously supported the syntax:

  -p *port#[/localport#]*where *port#* was the first port used

when contacting remote servers and *localport#* was the service port bound by the local instance of `in.named`.

The current usage is equivalent to the old usage without *localport#* specified. This functionality can be specified with the `listen-on` clause `options` statement in the configuration file.

`-q` — Trace all incoming queries. This option is deprecated in favor of the `queries` logging category of the logging statement in `named.conf`. See `named.conf`(4).

`-r` — Turn recursion off in the server. Responses come only from local (primary or secondary) zones. You can use this option on root servers. By default, the server uses recursion. This option is deprecated and is overriden by the `recursion` clause of the `options` statement in the configuration file.

`-t` *directory* — Specify the directory that the server is to `chroot`(2) into when it finishes processing command line arguments.

`-u` *user_name* — Specify the user that the server should run as after it initializes. The value of *user_name* may be either a user name or a numeric user id. If the `-g` option is not invoked, then the group id will be the primary group of the user specified. Since `initgroups`(3C) is called, all of the user's group will be available to the server.

`-v` — Report the version and exit.

`-w` *directory* — Set the working directory of the server. The `directory` clause of the `options` statement overrides any value specified on the command line. The default working directory is the current directory (".").

For compatibility with older implementations any additional argument will be interpreted as the name of the configuration file. This argument overrides any *config_file* specified by means of the `-b` or `-c` options. If no further argument is given, then the default configuration file, `/etc/named.conf` is used.

Master File Format

The master file consists of control information and a list of resources for objects in the zone. The file has the following form:

```
$INCLUDE <filename><opt_domain>
$ORIGIN <domain>
$TTL <ttl>
<domain><opt_ttl><opt_class><type><resource_record_data>
```

The fields are defined as follows:

domain
: The value of *domain* can be " . " for root, "@" for the current origin, or a standard domain name. If domain is a standard domain name that does not end with " . ", the current origin is appended to the domain. Domain names ending with " . " are not modified.

opt_domain
: This field is used to define an origin for the data in an included file. It is equivalent to placing an *$ORIGIN* statement before the first line of the included file. This field is optional. Neither the *opt_domain* field nor `$ORIGIN` statements in the included file modify the current origin for this file.

ttl
: An integer number that sets the default time-to-live for future records that do not have an explicit *ttl*.

opt_ttl
: An optional integer number for the time-to-live field. If not set the *ttl* is taken from the last `$TTL` statement. If no `$TTL` statement has occurred then the `SOA` minimum value is used, and a warning is generated.

opt_class
: The object address type. Currently only one type is supported, `IN`, for objects connected to the DARPA Internet.

type
: This field contains one of the following tokens. The data expected in the *resource_record_data* field is in parentheses:

  `A` — A host address (dotted-quad IP address).

  `NS` — An authoritative name server (domain).

  `MX` — A mail exchanger (domain), preceded by a preference value (0..32767), with lower numeric values representing higher logical preferences.

  `CNAME` — The canonical name for an alias (domain).

  `SOA` — Marks the start of a zone of authority (domain of originating host, domain address of maintainer, a serial number and the following parameters in seconds: refresh, retry, expire and minimum *ttl*). See *RFC 883* and *RFC 2308*.

  `NULL` — A `NULL` resource record (no format or data).

  `RP` — A Responsible Person for some domain name (mailbox, TXT-referral).

  `PTR` — A domain name pointer (domain).

	`HINFO` Host information (`cpu_type`, `OS_type`).
resource_record_data	Resource records normally end at the end of a line, but may be continued across lines between opening and closing parentheses. Comments are introduced by semicolons and continue to the end of the line. There are other resource record types not shown here. Some resource record types may have been standardized in more recent RFCs, but they have not yet been implemented in this version of BIND

SOA Record Format

Each master zone file should begin with an SOA record for the zone. An example SOA record is as follows:

```
@       IN      SOA     ucbvax.Berkeley.EDU. rwh.ucbvax.Berkeley.EDU. (
                                    1989020501      ; serial
                                    10800   ; refresh
                                    3600    ; retry
                                    3600000 ; expire
                                    86400 ) ; minimum
```

The SOA specifies a serial number that should be incremented each time the master file is changed. Note that the serial number can be given as a dotted number, but this is a very unwise thing to do since the translation to normal integers is by means of concatenation rather than multiplication and addition. You can spell out the year, month, day of month, and 0..99 version number and still fit inside the unsigned 32-bit size of this field. (It's true that we will have to rethink this strategy in the year 4294, but we're not worried about it.)

Secondary servers check the serial number at intervals specified by the refresh time in seconds; if the serial number changes, a zone transfer will be done to load the new data. If a master server cannot be contacted when a refresh is due, the retry time specifies the interval at which refreshes should be attempted. If a master server cannot be contacted within the interval given by the expire time, all data from the zone is discarded by secondary servers. The minimum value is the cache time-to-live for negative answers. See *RFC 2308*.

FILES

`/etc/named.conf`	default name server configuration file.
`/var/run/named.pid`	the process ID.
srvcwd	in.named's current working directory, usually set by the "directory" options statement in `named.conf`.
srvcwd`/named_dump.db`	dump of the name server database.
srvcwd`/named.run`	debug output.
srvcwd`/named.stats`	name server statistics data.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWinamd
Interface Stability	Standard BIND 8.2.4

SEE ALSO

`kill`(1), `named-xfer`(1M), `chroot`(2), `fork`(2), `getservbyname`(3SOCKET), `resolver`(3RESOLV), `signal`(3C), `syslog`(3C), `resolv.conf`(4), `attributes`(5)

Andrews, M. *RFC 2308, Negative Caching of DNS Queries (DNS NCACHE)*. Network Working Group. March 1998.

Braden, R., editor. *RFC 1123, Requirements for Internet Hosts - Applications and Support* . Network Working Group. October 1989.

Lotter, M. *RFC 1033, Domain Administrators Operations Guide*. Network Working Group. November 1987.

Mockapetris, Paul. *RFC 882, Domain Names - Concepts and Facilities*. Network Working Group. November 1983.

Mockapetris, Paul. *RFC 883, Domain Names - Implementation and Specification*. Network Working Group. November 1983.

Mockapetris, Paul. *RFC 973, Domain System Changes and Observations*. Network Working Group. January 1986.

Mockapetris, Paul. *RFC 1034, Domain Names - Concepts and Facilities*. Network Working Group. November 1987.

Mockapetris, Paul. *RFC 1035, Domain Names - Implementation and Specification*. Network Working Group. November 1987.

Partridge, Craig. *RFC 974, Mail Routing and the Domain System*. Network Working Group. January 1986.

Vixie, Paul, Dunlap, Kevin J., Karels, Michael J., *Name Server Operations Guide for BIND* Internet Software Consortium. 1996.

NOTES

The boot file directives `domain` and `suffixes` are now obsolete. They have been replaced by a resolver-based implementation of suffixing for partially-qualified domain names. The earlier mechanisms failed under a number of situations, for example, when the local name server did not have complete information.

The following signals have the specified effect when sent to the server process using the `kill`(1) command:

`SIGHUP` Causes the server to read `named.conf` and reload the database. `SIGHUP` will also cause the server to check the serial number on all secondary zones.

SIGINT	Dumps the current database and cache to named_dump.db.
SIGILL	Dumps statistics data into named.stats. Statistics data is appended to the file.
SIGTERM	Saves any modified dynamic zones to the file system and shuts down the server.
SIGUSR1	Turns on debugging. Each additional SIGUSR1 signal increments the debug level.
SIGUSR2	Turns off debugging completely.
SIGWINCH	Toggles logging of all incoming queries by means of syslog(3C).

NAME

in.ndpd – daemon for IPv6 autoconfiguration

SYNOPSIS

/usr/sbin/in.ndpd [-adt] [-f *config_file*]

DESCRIPTION

`in.ndpd` provides both the host and router autoconfiguration components of Neighbor Discovery for IPv6 and Stateless Address Autoconfiguration for IPv6. In particular, `in.ndpd` implements

- router discovery;
- prefix discovery;
- parameter discovery; and
- address autoconfiguration.

Other aspects of Neighbor Discovery are implemented by `ip6`(7P), including:

- address resolution;
- neighbor unreachability detection; and
- redirect.

The duplicate address detection function is implemented by `ifconfig`(1M).

If the `/etc/inet/ndpd.conf` file does not exist or does not set the variable `AdvSendAdvertisements` to true for a network interface, then `in.ndpd` will make the node a host for that interface, that is, sending router solicitation messages and then using router advertisement messages it receives to autoconfigure the node. Note that `in.ndpd` only autoconfigures the addresses of global or site-local scope from the prefix advertisement.

If `AdvSendAdvertisements` is set to true for an interface, then `in.ndpd` will perform router functions on that interface, that is, sending router advertisement messages to autoconfigure the attached hosts, but not use any advertisements it receives for autoconfiguration. However, when sending advertisements, `in.ndpd` will use the advertisements it sends itself to autoconfigure its prefixes. For improved robustness `in.ndpd` stores any autoconfigured IPv6 addresses and their expiration times in state files named `ndpd_state.`*interface* that are located in the `/var/inet` directory. Should `in.ndpd` fail to find any routers, it will use the state files as a fallback, autoconfiguring those addresses if the recorded addresses have remaining lifetime. This ensures that a host that reboots faster than the routers, for example after a short power failure, will continue using the addresses that it had before the power failure.

OPTIONS

`-a`	Turn off stateless address auto configuration. When set, the daemon does not autoconfigure any addresses and does not renumber any addresses.
`-d`	Turn on large amounts of debugging output on `stdout`. When set, the program runs in the foreground and stays attached to the controlling terminal.
`-f` *config_file*	Use *config_file* for configuration information instead of the default `/etc/inet/ndpd.conf`.

`-t` Turn on tracing (printing) of all sent and received packets to `stdout`. When set, the program runs in the foreground and stays attached to the controlling terminal.

FILES

`/etc/inet/ndpd.conf` Configuration file. Not needed on a host but required on a router to enable `in.ndpd` to advertise autoconfiguration information to the hosts.

`/var/inet/ndpd_state.`*interface* Contains the addresses for *interface*. The existence of an address in this file does not imply that the address is usable, since the address lifetime may have expired.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWroute

SEE ALSO

`ifconfig`(1M), `ndpd.conf`(4), `attributes`(5), `icmp6`(7p),`ip6`(7p) `attributes`(5)

Narten, T., Nordmark, E., Simpson, W., *RFC 2461, Neighbor Discovery for IP Version 6 (IPv6)*, The Internet Society, December 1998.

Thomson, S., Narten, T., *RFC 2462, IPv6 Stateless Address Autoconfiguration*, The Internet Society, December 1998.

DIAGNOSTICS

Receipt of a `SIGHUP` signal will make `in.ndpd` restart and reread `/etc/inet/ndpd.conf`.

NAME

in.rarpd, rarpd – DARPA Reverse Address Resolution Protocol server

SYNOPSIS

/usr/sbin/in.rarpd [-d] -a

/usr/sbin/in.rarpd [-d] *device unit*

DESCRIPTION

`in.rarpd` starts a daemon that responds to Reverse Address Resolution Protocol (RARP) requests. The daemon forks a copy of itself that runs in background. It must be run as root.

RARP is used by machines at boot time to discover their Internet Protocol (IP) address. The booting machine provides its Ethernet address in a RARP request message. Using the `ethers` and `hosts` databases, `in.rarpd` maps this Ethernet address into the corresponding IP address which it returns to the booting machine in an RARP reply message. The booting machine must be listed in both databases for `in.rarpd` to locate its IP address. `in.rarpd` issues no reply when it fails to locate an IP address.

`in.rarpd` uses the STREAMS-based Data Link Provider Interface (DLPI) message set to communicate directly with the datalink device driver.

OPTIONS

The following options are supported:

`-a` Get the list of available network interfaces from IP using the SIOCGIFADDR ioctl and start a RARP daemon process on each interface returned.

`-d` Print assorted debugging messages while executing.

EXAMPLES

EXAMPLE 1 Starting An `in.rarpd` Daemon For Each Network Interface Name Returned From `/dev/ip:`

The following command starts an `in.rarpd` for each network interface name returned from `/dev/ip:`

```
example# /usr/sbin/in.rarpd -a
```

EXAMPLE 2 Starting An `in.rarpd` Daemon On The Device `/dev/le` With The Device Instance Number `0`

The following command starts one `in.rarpd` on the device `/dev/le` with the device instance number `0`.

```
example# /usr/sbin/in.rarpd le 0
```

FILES

`/etc/ethers` File or other source, as specified by `nsswitch.conf`(4).

`/etc/hosts` File or other source, as specified by `nsswitch.conf`(4).

`/tftpboot`

`/dev/ip`

`/dev/arp`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWbsu

SEE ALSO

`boot`(1M), `ifconfig`(1M), `ethers`(4), `hosts`(4), `netconfig`(4), `nsswitch.conf`(4),`attributes` (5), `dlpi`(7P)

Finlayson, R., Mann, T., Mogul, J., and Theimer, M., *RFC 903, A Reverse Address Resolution Protocol*, Network Information Center, SRI International, June 1984.

Unix International, *Data Link Provider Interface*, Version 2, May 7, 1991, Sun Microsystems, 800-6915-01.

NAME

in.rdisc, rdisc – network router discovery daemon

SYNOPSIS

/usr/sbin/in.rdisc [-a] [-f] [-s] [*send-address*] [*receive-address*]

/usr/sbin/in.rdisc -r [-p *preference*] [-T *interval*] [*send-address*] [*receive-address*]

DESCRIPTION

in.rdisc implements the ICMP router discovery protocol. The first form of the command is used on hosts and the second form is used on routers.

At boot time, in.rdisc is invoked in either the first form (host mode) or second form (router mode). The second form is invoked under the following conditions:

- A machine has multiple interfaces
- The interfaces are not configured by DHCP
- The defaultrouter(4) and /etc/notrouter files are not present

Otherwise, the first form of in.rdisc is invoked.

On a host, in.rdisc populates the network routing tables with default routes. On a router, advertises the router to all the hosts.

Host (First Form)

On a host, in.rdisc listens on the ALL_HOSTS (224.0.0.1) multicast address for ROUTER_ADVERTISE messages from routers. The received messages are handled by first ignoring those listed router addresses with which the host does not share a network. Among the remaining addresses, the ones with the highest preference are selected as default routers and a default route is entered in the kernel routing table for each one of them.

Optionally, in.rdisc can avoid waiting for routers to announce themselves by sending out a few ROUTER_SOLICITATION messages to the ALL_ROUTERS (224.0.0.2) multicast address when it is started.

A timer is associated with each router address. The address will no longer be considered for inclusion in the routing tables if the timer expires before a new *advertise* message is received from the router. The address will also be excluded from consideration if the host receives an *advertise* message with the preference being maximally negative or with a lifetime of zero.

Router (Second Form)

When in.rdisc is started on a router, it uses the SIOCGIFCONF ioctl(2) to find the interfaces configured into the system and it starts listening on the ALL_ROUTERS multicast address on all the interfaces that support multicast. It sends out *advertise* messages to the ALL_HOSTS multicast address advertising all its IP addresses. A few initial *advertise* messages are sent out during the first 30 seconds and after that it will transmit *advertise* messages approximately every 600 seconds.

When in.rdisc receives a *solicitation* message, it sends an *advertise* message to the host that sent the *solicitation* message.

When in.rdisc is terminated by a signal, it sends out an *advertise* message with the preference being maximally negative.

OPTIONS

`-a`	Accept all routers independent of the preference they have in their *advertise* messages. Normally, `in.rdisc` only accepts (and enters in the kernel routing tables) the router or routers with the highest preference.
`-f`	Run `in.rdisc` forever even if no routers are found. Normally, `in.rdisc` gives up if it has not received any *advertise* message after soliciting three times, in which case it exits with a non-zero exit code. If `-f` is not specified in the first form then `-s` must be specified.
`-r`	Act as a router, rather than a host.
`-s`	Send three *solicitation* messages initially to quickly discover the routers when the system is booted. When `-s` is specified, `in.rdisc` exits with a non-zero exit code if it can not find any routers. This can be overridden with the `-f` option.
`-p` *preference*	Set the preference transmitted in the *solicitation* messages. The default is zero.
`-T` *interval*	Set the interval between transmitting the *advertise* messages. The default time is 600 seconds.

FILES

/etc/notrouter	A file that acts as a flag for a boot-time script that determines whether a machine will act as a router or a host. The presence of the `notrouter` file prevents a machine that does not have a default route from acting as an IPv4 router, even if that machine has multiple interfaces.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWroute

SEE ALSO

`in.routed`(1M), `ioctl`(2), `defaultrouter`(4), `attributes`(5), `icmp`(7P), `inet`(7P)

Deering, S.E., editor, *ICMP Router Discovery Messages*, RFC 1256, Network Information Center, SRI International, Menlo Park, California, September 1991.

NAME

in.rexecd, rexecd – remote execution server

SYNOPSIS

```
in.rexecd
```

DESCRIPTION

`in.rexecd` is the server for the `rexec`(3SOCKET) routine. The server provides remote execution facilities with authentication based on user names and passwords. It is invoked automatically as needed by `inetd`(1M), and then executes the following protocol:

1. The server reads characters from the socket up to a null (`\0`) byte. The resultant string is interpreted as an ASCII number, base 10.
2. If the number received in step 1 is non-zero, it is interpreted as the port number of a secondary stream to be used for the `stderr`. A second connection is then created to the specified port on the client's machine.
3. A null terminated user name of at most 16 characters is retrieved on the initial socket.
4. A null terminated password of at most 16 characters is retrieved on the initial socket.
5. A null terminated command to be passed to a shell is retrieved on the initial socket. The length of the command is limited by the upper bound on the size of the system's argument list.
6. `rexecd` then validates the user as is done at login time and, if the authentication was successful, changes to the user's home directory, and establishes the user and group protections of the user. If any of these steps fail the connection is aborted and a diagnostic message is returned.
7. A null byte is returned on the connection associated with the `stderr` and the command line is passed to the normal login shell of the user. The shell inherits the network connections established by `rexecd`.

USAGE

`in.rexecd` and `rexecd` are IPv6–enabled. See `ip6`(7P).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWrcmds

SEE ALSO

`inetd`(1M), `rexec`(3SOCKET), `inetd.conf`(4), `attributes`(5), `ip6`(7P)

DIAGNOSTICS

All diagnostic messages are returned on the connection associated with the `stderr`, after which any network connections are closed. An error is indicated by a leading byte with a value of 1 (0 is returned in step 7 above upon successful completion of all the steps prior to the command execution).

`username too long` The name is longer than 16 characters.

`password too long` The password is longer than 16 characters.

`command too long`	The command line passed exceeds the size of the argument list (as configured into the system).
`Login incorrect.`	No password file entry for the user name existed.
`Password incorrect.`	The wrong password was supplied.
`No remote directory.`	The `chdir` command to the home directory failed.
`Try again.`	A `fork` by the server failed.
`/usr/bin/sh: ...`	The user's login shell could not be started.

NAME

in.ripngd – network routing daemon for IPv6

SYNOPSIS

/usr/sbin/in.ripngd [-s] [-q] [-t] [-p *n*] [-P] [-v] [*logfile*]

DESCRIPTION

in.ripngd is the IPv6 equivalent of in.routed(1M). It is invoked at boot time to manage the network routing tables. The routing daemon uses the Routing Information Protocol for IPv6.

In normal operation, in.ripngd listens on the udp(7P) socket port 521 for routing information packets. If the host is an internetwork router, it periodically supplies copies of its routing tables to any directly connected hosts and networks.

Whenin.ripngd is started, it uses the SIOCGLIFCONF ioctl(2) to find those directly connected IPv6 interfaces configured into the system and marked "up"; the software loopback interface is ignored. If multiple interfaces are present, it is assumed the host will forward packets between networks. in.ripngd then multicasts a request packet on each IPv6 interface and enters a loop, listening for request and response packets from other hosts.

When a request packet is received, in.ripngd formulates a reply based on the information maintained in its internal tables. The response packet contains a list of known routes. With each route is a number specifying the number of bits in the prefix. The prefix is the number of bits in the high order part of an address that indicate the subnet or network that the route describes. Each route reported also has a "*hop count*" metric. A count of 16 or greater is considered "infinity." The metric associated with each route returned provides a metric relative to the sender.

The request packets received by in.ripngd are used to update the routing tables if one of the following conditions is satisfied:

- No routing table entry exists for the destination network or host, and the metric indicates the destination is "reachable, that is, the *hop count* is not infinite.
- The source host of the packet is the same as the router in the existing routing table entry. That is, updated information is being received from the very internetwork router through which packets for the destination are being routed.
- The existing entry in the routing table has not been updated for a period of time, defined to be 90 seconds, and the route is at least as cost-effective as the current route.
- The new route describes a shorter route to the destination than the one currently stored in the routing tables; this is determined by comparing the metric of the new route against the one stored in the table.

When an update is applied, in.ripngd records the change in its internal tables and generates a response packet to all directly connected hosts and networks. To allow possible unstable situations to settle, in.ripngd waits a short period of time (no more than 30 seconds) before modifying the kernel's routing tables.

In addition to processing incoming packets, `in.ripngd` also periodically checks the routing table entries. If an entry has not been updated for 3 minutes, the entry's metric is set to infinity and marked for deletion. Deletions are delayed an additional 60 seconds to insure the invalidation is propagated throughout the internet.

Hosts acting as internetwork routers gratuitously supply their routing tables every 30 seconds to all directly connected hosts and networks.

OPTIONS

`in.ripngd` supports the following options:

`-q` Do not supply routing information.

`-s` Force `in.ripngd` to supply routing information whether it is acting as an internetwork router or not.

`-p` *n* Send and receive the routing packets from other routers using the UDP port number *n*.

`-P` Do not use poison reverse.

`-t` Print all packets sent or received to standard output. `in.ripngd` will not divorce itself from the controlling terminal. Accordingly, interrupts from the keyboard will kill the process.

`-v` Print all changes made to the routing tables to standard output with a timestamp.

Any other argument supplied is interpreted as the name of the file in which the actions of `in.ripngd`, as specified by this option or by `-t`, should be logged versus being sent to standard output.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWroute

SEE ALSO

`in.routed`(1M), `ioctl`(2), `attributes`(5), `udp`(7P)

Malkin, G., Minnear, R., *RFC 2080, RIPng for IPv6*, January 1997.

NOTES

The kernel's routing tables may not correspond to those of `in.ripngd` for short periods of time while processes that utilize existing routes exit; the only remedy for this is to place the routing process in the kernel.

`in.ripngd` currently does not support all of the functionality of `in.routed`(1M). Future releases may support more if appropriate.

`in.ripngd` initially obtains a routing table by examining the interfaces configured on a machine. It then sends a request on all directly connected networks for more routing information. `in.ripngd` does not recognize or use any routing information already

established on the machine prior to startup. With the exception of interface changes, `in.ripngd` does not see any routing table changes that have been done by other programs on the machine, for example, routes added, deleted or flushed by way of the `route`(1M) command. Therefore, these types of changes should not be done while `in.ripngd` is running. Rather, shut down `in.ripngd`, make the changes required, and then restart `in.ripngd`.

NAME

in.rlogind, rlogind – remote login server

SYNOPSIS

/usr/sbin/in.rlogind

DESCRIPTION

`in.rlogind` is the server for the `rlogin`(1) program. The server provides a remote login facility with authentication based on privileged port numbers.

`in.rlogind` is invoked by `inetd`(1M) when a remote login connection is established, and executes the following protocol:

- The server checks the client's source port. If the port is not in the range 512-1023, the server aborts the connection.
- The server checks the client's source address. If an entry for the client exists in both `/etc/hosts` and `/etc/hosts.equiv`, a user logging in from the client is not prompted for a password. If the address is associated with a host for which no corresponding entry exists in `/etc/hosts`, the user is prompted for a password, regardless of whether or not an entry for the client is present in `/etc/hosts.equiv`. See `hosts`(4) and `hosts.equiv`(4).

Once the source port and address have been checked, `in.rlogind` allocates a pseudo-terminal and manipulates file descriptors so that the slave half of the pseudo-terminal becomes the `stdin`, `stdout`, and `stderr` for a login process. The login process is an instance of the `login`(1) program, invoked with the `-r`.

The login process then proceeds with the `pam`(3PAM) authentication process. See `SECURITY` below. If automatic authentication fails, it reprompts the user to login.

The parent of the login process manipulates the master side of the pseudo-terminal, operating as an intermediary between the login process and the client instance of the `rlogin` program. In normal operation, a packet protocol is invoked to provide Ctrl-S and Ctrl-Q type facilities and propagate interrupt signals to the remote programs. The login process propagates the client terminal's baud rate and terminal type, as found in the environment variable, `TERM`; see `environ`(4).

USAGE

`rlogind` and `in.rlogind` are IPv6–enabled. See `ip6`(7P).

SECURITY

`in.rlogind` uses `pam`(3PAM) for authentication, account management, and session management. The PAM configuration policy, listed through `/etc/pam.conf`, specifies the modules to be used for `in.rlogind`. Here is a partial `pam.conf` file with entries for the `rlogin` command using the "rhosts" and UNIX authentication modules, and the UNIX account, session management, and password management modules.

```
rlogin          auth sufficient     pam_rhosts_auth.so.1
rlogin          auth requisite      pam_authtok_get.so.1
rlogin          auth required       pam_dhkeys.so.1
```

```
rlogin          auth required       pam_unix_auth.so.1

rlogin          account required    pam_unix_roles.so.1
rlogin          account required    pam_unix_projects.so.1
rlogin          account required    pam_unix_account.so.1

rlogin          session required    pam_unix_session.so.1
```

With this configuration, the server checks the client's source address. If an entry for the client exists in both `/etc/hosts` and `/etc/hosts.equiv`, a user logging in from the client is not prompted for a password. If the address is associated with a host for which no corresponding entry exists in `/etc/hosts`, the user is prompted for a password, regardless of whether or not an entry for the client is present in `/etc/hosts.equiv`. See `hosts`(4) and `hosts.equiv`(4).

If there are no entries for the `rlogin` service, then the entries for the "other" service will be used. If multiple authentication modules are listed, then the user may be prompted for multiple passwords. Removing the "`pam_rhosts_auth.so.1`" entry will disable the `/etc/hosts.equiv` and `~/.rhosts` authentication protocol and the user would always be forced to type the password. The *sufficient* flag indicates that authentication through the `pam_rhosts_auth.so.1` module is "sufficient" to authenticate the user. Only if this authentication fails is the next authentication module used.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWrcmds

SEE ALSO

`login`(1), `rlogin`(1), `in.rshd`(1M), `inetd`(1M), `pam`(3PAM), `environ`(4), `hosts`(4), `hosts.equiv`(4), `inetd.conf`(4), `pam.conf`(4), `attributes`(5), `pam_authtok_check`(5), `pam_authtok_get`(5), `pam_authtok_store`(5), `pam_dhkeys`(5), `pam_passwd_auth`(5), `pam_unix`(5), `pam_unix_account`(5), `pam_unix_auth`(5), `pam_unix_session`(5)

DIAGNOSTICS

All diagnostic messages are returned on the connection associated with the `stderr`, after which any network connections are closed. An error is indicated by a leading byte with a value of 1.

`Hostname for your address unknown.`
No entry in the host name database existed for the client's machine.

`Try again.`
A *fork* by the server failed.

`/usr/bin/sh: ...`
The user's login shell could not be started.

NOTES

The authentication procedure used here assumes the integrity of each client machine and the connecting medium. This is insecure, but it is useful in an ''open'' environment.

A facility to allow all data exchanges to be encrypted should be present.

The `pam_unix`(5) module might not be supported in a future release. Similar functionality is provided by `pam_authtok_check`(5), `pam_authtok_get`(5), `pam_authtok_store`(5), `pam_dhkeys`(5), `pam_passwd_auth`(5), `pam_unix_account`(5), `pam_unix_auth`(5), and `pam_unix_session`(5).

NAME

in.routed, routed – network routing daemon

SYNOPSIS

/usr/sbin/in.routed [-s] [-q] [-t] [-g] [-S] [-v] [*logfile*]

DESCRIPTION

`in.routed` is invoked at boot time to manage the network routing tables. The routing daemon uses a variant of the Xerox NS Routing Information Protocol in maintaining up-to-date kernel routing table entries.

At boot time, `in.routed` is invoked with either the `-s` or `-q` option. Options are described below. The `-s` option is appropriate for a router. The `in.routed -s` command is invoked under the following conditions:

- A machine has multiple interfaces
- The interfaces are not configured by DHCP
- The `defaultrouter`(4) and `/etc/notrouter` files are not present

Otherwise, the `in.routed -q` command is invoked.

In normal operation, `in.routed` listens on `udp`(7P) socket 520 (decimal) for routing information packets. If the host is an internetwork router, it periodically supplies copies of its routing tables to any directly connected hosts and networks.

When `in.routed` is started, it uses the `SIOCGIFCONF` `ioctl`(2) to find those directly connected interfaces configured into the system and marked "up" (the software loopback interface is ignored). If multiple interfaces are present, it is assumed the host will forward packets between networks. `in.routed` then transmits a *request* packet on each interface (using a broadcast packet if the interface supports it) and enters a loop, listening for *request* and *response* packets from other hosts.

When a *request* packet is received, `in.routed` formulates a reply based on the information maintained in its internal tables. The *response* packet contains a list of known routes, each marked with a "hop count" metric (a count of 16, or greater, is considered "infinite"). The metric associated with each route returned, provides a metric relative to the sender.

request packets received by `in.routed` are used to update the routing tables if one of the following conditions is satisfied:

- No routing table entry exists for the destination network or host, and the metric indicates the destination is "reachable" (that is, the hop count is not infinite).
- The source host of the packet is the same as the router in the existing routing table entry. That is, updated information is being received from the very internetwork router through which packets for the destination are being routed.
- The existing entry in the routing table has not been updated for some time (defined to be 90 seconds) and the route is at least as cost effective as the current route.
- The new route describes a shorter route to the destination than the one currently stored in the routing tables; the metric of the new route is compared against the one stored in the table to decide this.

When an update is applied, `in.routed` records the change in its internal tables and generates a *response* packet to all directly connected hosts and networks. `in.routed` waits a short period of time (no more than 30 seconds) before modifying the kernel's routing tables to allow possible unstable situations to settle.

In addition to processing incoming packets, `in.routed` also periodically checks the routing table entries. If an entry has not been updated for 3 minutes, the entry's metric is set to infinity and marked for deletion. Deletions are delayed an additional 60 seconds to insure the invalidation is propagated throughout the internet.

Hosts acting as internetwork routers gratuitously supply their routing tables every 30 seconds to all directly connected hosts and networks.

In addition to the facilities described above, `in.routed` supports the notion of "distant" `passive` and `active` gateways. When `in.routed` is started up, it reads the file `gateways` to find gateways which may not be identified using the `SIOCGIFCONF`ioctl. Gateways specified in this manner should be marked `passive` if they are not expected to exchange routing information, while gateways marked `active` should be willing to exchange routing information (that is, they should have a `in.routed` process running on the machine). Routes through passive gateways are installed in the kernel's routing tables once upon startup. They may change, depending upon routing information they receive from other gateways. Information regarding their existence is not included in any routing information transmitted. Active gateways are treated equally to network interfaces. Routing information is distributed to the gateway, and if no routing information is received for a period of time, the associated route is deleted.

The `gateways` is comprised of a series of lines, each in the following format:

`< net | host >` *filename1* `gateway` *filename2* `metric` *value* `< passive | active >`

The `net` or `host` keyword indicates if the route is to a network or specific host.

filename1 is the name of the destination network or host. This may be a symbolic name located in `networks` or `hosts`, or an Internet address specified in "dot" notation; see `inet`(3SOCKET).

filename2 is the name or address of the gateway to which messages should be forwarded.

value is a metric indicating the hop count to the destination host or network.

The keyword `passive` or `active` indicates if the gateway should be treated as passive or active (as described above).

OPTIONS

`-g` Is used on internetwork routers to offer a route to the ''default'' destination. This is typically used on a gateway to the Internet, or on a gateway that uses another routing protocol whose routes are not reported to other local routers.

`-q` Is the opposite of the `-s` option.

`-s` Forces `in.routed` to supply routing information whether it is acting as an internetwork router or not.

`-S` If `in.routed` is not acting as an internetwork router it will, instead of entering the whole routing table in the kernel, only enter a default route for each internetwork router. This reduces the the memory requirements without losing any routing reliability.

`-t` All packets sent or received are printed on standard output. In addition, `in.routed` will not divorce itself from the controlling terminal so that interrupts from the keyboard will kill the process. Any other argument supplied is interpreted as the name of the file in which `in.routed`'s actions should be logged. This log contains information about any changes to the routing tables and a history of recent messages sent and received which are related to the changed route.

`-v` Allows a logfile (whose name must be supplied) to be created showing the changes made to the routing tables with a timestamp.

FILES

`/etc/defaultrouter`	list of defaultrouters
`/etc/gateways`	list of distant gateways
`/etc/hosts`	Internet host table
`/etc/networks`	associations of Internet Protocol network numbers with network names
/etc/notrouter	A file that acts as a flag for a boot-time script that determines whether a machine will act as a router or a host. The presence of the `notrouter` file prevents a machine that does not have a default route from acting as an IPv4 router, even if that machine has multiple interfaces.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWroute

SEE ALSO

`route`(1M), `ioctl`(2), `inet`(3SOCKET), `defaultrouter`(4), `attributes`(5), `inet`(7P), `udp`(7P)

NOTES

The kernel's routing tables may not correspond to those of `in.routed` for short periods of time while processes that utilize existing routes exit; the only remedy for this is to place the routing process in the kernel.

`in.routed` should listen to intelligent interfaces, such as an IMP, and to error protocols, such as ICMP, to gather more information.

`in.routed` initially obtains a routing table by examining the interfaces configured on a machine and the `gateways` file. It then sends a request on all directly connected networks for more routing information. `in.routed` does not recognize or use any routing information already established on the machine prior to startup. With the exception of interface changes, `in.routed` does not see any routing table changes that have been done by other programs on the machine, for example, routes added, deleted or flushed by way of the `route`(1M) command. Therefore, these types of changes should not be done while `in.routed` is running. Rather, shut down `in.routed`, make the changes required, and then restart `in.routed`.

NAME

in.rshd, rshd – remote shell server

SYNOPSIS

`in.rshd` *host.port*

DESCRIPTION

`in.rshd` is the server for the `rsh`(1) program. The server provides remote execution facilities with authentication based on privileged port numbers.

`in.rshd` is invoked by `inetd`(1M) each time a shell service is requested, and executes the following protocol:

1. The server checks the client's source port. If the port is not in the range 512-1023, the server aborts the connection. The client's host address (in hex) and port number (in decimal) are the arguments passed to `in.rshd`.
2. The server reads characters from the socket up to a null (`\0`) byte. The resultant string is interpreted as an ASCII number, base 10.
3. If the number received in step 2 is non-zero, it is interpreted as the port number of a secondary stream to be used for the `stderr`. A second connection is then created to the specified port on the client's machine. The source port of this second connection is also in the range 512-1023.
4. A null-terminated user name of at most 16 characters is retrieved on the initial socket. This user name is interpreted as the user identity on the *client*'s machine.
5. A null terminated user name of at most 16 characters is retrieved on the initial socket. This user name is interpreted as a user identity to use on the *server*'s machine.
6. A null terminated command to be passed to a shell is retrieved on the initial socket. The length of the command is limited by the upper bound on the size of the system's argument list.
7. `in.rshd` then validates the user according to the following steps. The remote user name is looked up in the password file and a `chdir` is performed to the user's home directory. If the lookup fails, the connection is terminated. If the `chdir` fails, it does a `chdir` to `/` (root). If the user is not the superuser, (user ID 0), and if the pam_rhosts_auth PAM module is configured for authentication, the file `/etc/hosts.equiv` is consulted for a list of hosts considered "equivalent". If the client's host name is present in this file, the authentication is considered successful. See the SECURITY section below for a discussion of PAM authentication.

   If the lookup fails, or the user is the superuser, then the file `.rhosts` in the home directory of the remote user is checked for the machine name and identity of the user on the client's machine. If this lookup fails, the connection is terminated
8. A null byte is returned on the connection associated with the `stderr` and the command line is passed to the normal login shell of the user. (The `PATH` variable is set to `/usr/bin`.) The shell inherits the network connections established by `in.rshd`.

USAGE

`rshd` and `in.rshd` are IPv6–enabled. See `ip6`(7P).

SECURITY

`in.rshd` uses `pam`(3PAM) for authentication, account management, and session management. The PAM configuration policy, listed through `/etc/pam.conf`, specifies the modules to be used for `in.rshd`. Here is a partial `pam.conf` file with entries for the `rsh` command using rhosts authentication, UNIX account management, and session management module.

```
rsh     auth    required    pam_rhosts_auth.so.1

rsh     account required    pam_unix_roles.so.1
rsh     session required    pam_unix_projects.so.1
rsh     session required    pam_unix_account.so.1

rsh     session required    pam_unix_session.so.1
```

If there are no entries for the `rsh` service, then the entries for the "other" service are used. To maintain the authentication requirement for `in.rshd`, the rsh entry must always be configured with the `pam_rhosts_auth.so.1` module.

FILES

`/etc/hosts.equiv`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWrcmds

SEE ALSO

`rsh`(1), `inetd`(1M), `pam`(3PAM), `hosts`(4), `inetd.conf`(4), `pam.conf`(4), `attributes`(5), `environ`(5), `pam_authtok_check`(5), `pam_authtok_get`(5), `pam_authtok_store`(5), `pam_dhkeys`(5), `pam_passwd_auth`(5), `pam_rhosts_auth`(5), `pam_unix`(5), `pam_unix_account`(5), `pam_unix_auth`(5), `pam_unix_session`(5)`ip6`(7P)

DIAGNOSTICS

The following diagnostic messages are returned on the connection associated with `stderr`, after which any network connections are closed. An error is indicated by a leading byte with a value of 1 in step 8 above (`0` is returned above upon successful completion of all the steps prior to the command execution).

`locuser too long`
The name of the user on the client's machine is longer than 16 characters.

`remuser too long`
The name of the user on the remote machine is longer than 16 characters.

`command too long`
The command line passed exceeds the size of the argument list (as configured into the system).

`Hostname for your address unknown.`
No entry in the host name database existed for the client's machine.

`Login incorrect.`
No password file entry for the user name existed.

`Permission denied.`
The authentication procedure described above failed.

`Can't make pipe.`
The pipe needed for the `stderr` was not created.

`Try again.`
A *fork* by the server failed.

NOTES

The authentication procedure used here assumes the integrity of each client machine and the connecting medium. This is insecure, but it is useful in an "open" environment.

A facility to allow all data exchanges to be encrypted should be present.

The `pam_unix`(5) module might not be supported in a future release. Similar functionality is provided by `pam_authtok_check`(5), `pam_authtok_get`(5), `pam_authtok_store`(5), `pam_dhkeys`(5), `pam_passwd_auth`(5), `pam_unix_account`(5), `pam_unix_auth`(5), and `pam_unix_session`(5).

NAME

in.rwhod, rwhod – system status server

SYNOPSIS

/usr/sbin/in.rwhod [-m [*ttl*]]

DESCRIPTION

`in.rwhod` is the server which maintains the database used by the `rwho`(1) and `ruptime`(1) programs. Its operation is predicated on the ability to broadcast or multicast messages on a network.

`in.rwhod` operates as both a producer and consumer of status information. As a producer of information it periodically queries the state of the system and constructs status messages which are broadcast or multicast on a network. As a consumer of information, it listens for other `in.rwhod` servers' status messages, validating them, then recording them in a collection of files located in the directory `/var/spool/rwho`.

The `rwho` server transmits and receives messages at the port indicated in the `rwho` service specification, see `services`(4). The messages sent and received are defined in `/usr/include/protocols/rwhod.h` and are of the form:

```
struct  outmp {
        char    out_line[8];    /* tty name */
        char    out_name[8];    /* user id */
        long    out_time;       /* time on */
};
struct  whod {
        char    wd_vers;
        char    wd_type;
        char    wd_fill[2];
        int     wd_sendtime;
        int     wd_recvtime;
        char    wd_hostname[32];
        int     wd_loadav[3];
        int     wd_boottime;
        struct  whoent {
                struct  outmp we_utmp;
                int     we_idle;
    } wd_we[1024 / sizeof (struct whoent)];
 };
```

All fields are converted to network byte order prior to transmission. The load averages are as calculated by the `w`(1) program, and represent load averages over the 1, 5, and 15 minute intervals prior to a server's transmission. The host name included is that returned by the `uname`(2) system call. The array at the end of the message contains information about the users who are logged in to the sending machine. This information includes the contents of the `utmpx`(4) entry for each non-idle terminal line and a value indicating the time since a character was last received on the terminal line.

Messages received by the `rwho` server are discarded unless they originated at a `rwho` server's port. In addition, if the host's name, as specified in the message, contains any unprintable ASCII characters, the message is discarded. Valid messages received by `in.rwhod` are placed in files named `whod.hostname` in the directory `/var/spool/rwho`. These files contain only the most recent message, in the format described above.

Status messages are generated approximately once every 3 minutes.

OPTIONS

The following options are supported:

`-m` [*ttl*] Use the rwho IP multicast address (224.0.1.3) when transmitting. Receive announcements both on this multicast address and on the IP broadcast address. If *ttl* is not specified `in.rwhod` multicasts on all interfaces but with the IP TimeToLive set to 1 (that is, packets are not forwarded by multicast routers.) If *ttl* is specified `in.rwhod` only transmits packets on one interface and setting the IP TimeToLive to the specified *ttl*.

FILES

`/var/spool/rwho/whod.*` information about other machines

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWrcmds

SEE ALSO

`ruptime`(1), `rwho`(1), `w`(1), `uname`(2), `services`(4), `utmpx`(4), `attributes`(5)

WARNINGS

This service can cause network performance problems when used by several hosts on the network. It is not run at most sites by default. If used, include the `-m` multicast option.

NOTES

This service takes up progressively more network bandwidth as the number of hosts on the local net increases. For large networks, the cost becomes prohibitive.

`in.rwhod` should relay status information between networks. People often interpret the server dying as a machine going down.

NAME

install – install commands

SYNOPSIS

/usr/sbin/install `-c` *dira* `[-m` *mode*`]` `[-u` *user*`]` `[-g` *group*`]` `[-o]` `[-s]` *file*

/usr/sbin/install `-f` *dirb* `[-m` *mode*`]` `[-u` *user*`]` `[-g` *group*`]` `[-o]` `[-s]` *file*

/usr/sbin/install `-n` *dirc* `[-m` *mode*`]` `[-u` *user*`]` `[-g` *group*`]` `[-o]` `[-s]` *file*

/usr/sbin/install `-d | -i [-m` *mode*`]` `[-u` *user*`]` `[-g` *group*`]` `[-o]` `[-s]` *dirx...*

/usr/sbin/install `[-m` *mode*`]` `[-u` *user*`]` `[-g` *group*`]` `[-o]` `[-s]` *file* `[`*dirx...*`]`

DESCRIPTION

`install` is most commonly used in ''makefiles'' (see `make`(1S)) to install a `file` in specific locations, or to create directories within a file system. Each `file` is installed by copying it into the appropriate directory.

`install` uses no special privileges to copy files from one place to another. The implications of this are:

- You must have permission to read the files to be installed.
- You must have permission to copy into the destination directory.
- You must have permission to change the modes on the final copy of the file if you want to use the `-m` option.
- You must be super-user if you want to specify the ownership of the installed file with the `-u` or `-g` options. If you are not the super-user, the installed file will be owned by you, regardless of who owns the original.

`install` prints messages telling the user exactly what files it is replacing or creating and where they are going.

If no options or directories (*dirx* . . .) are given, `install` searches a set of default directories (`/bin`, `/usr/bin`, `/etc`, `/lib`, and `/usr/lib`, in that order) for a file with the same name as `file`. When the first occurrence is found, `install` issues a message saying that it is overwriting that file with `file`, and proceeds to do so. If the file is not found, the program states this and exits.

If one or more directories (*dirx* . . .) are specified after `file`, those directories are searched before the default directories.

OPTIONS

`-c` *dira* — Install `file` in the directory specified by *dira*, if `file` does not yet exist. If it is found, `install` issues a message saying that the file already exists, and exits without overwriting it.

`-f` *dirb* — Force `file` to be installed in given directory, even if the file already exists. If the file being installed does not already exist, the mode and owner of the new file will be set to `755` and `bin` , respectively. If the file already exists, the mode and owner will be that of the already existing file.

`-n` *dirc*	If `file` is not found in any of the searched directories, it is put in the directory specified in *dirc*. The mode and owner of the new file will be set to `755` and `bin`, respectively.
`-d`	Create a directory. Missing parent directories are created as required as in `mkdir -p`. If the directory already exists, the owner, group and mode will be set to the values given on the command line.
`-i`	Ignore default directory list, searching only through the given directories (*dirx* . . .).
`-m` *mode*	The mode of the new file is set to *mode*. Set to `0755` by default.
`-u` *user*	The owner of the new file is set to *user*. Only available to the super-user. Set to `bin` by default.
`-g` *group*	The group id of the new file is set to *group*. Only available to the super-user. Set to `bin` by default.
`-o`	If `file` is found, save the "found" file by copying it to `OLDfile` in the directory in which it was found. This option is useful when installing a frequently used file such as `/bin/sh` or `/lib/saf/ttymon`, where the existing file cannot be removed.
`-s`	Suppress printing of messages other than error messages.

USAGE

See `largefile`(5) for the description of the behavior of `install` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`chgrp`(1), `chmod`(1), `chown`(1), `cp`(1), `make`(1S), `mkdir`(1), `chown`(1M), `attributes`(5), `largefile`(5)

NAME

installboot – install bootblocks in a disk partition

SPARC

installboot *bootblk raw-disk-device*

IA

installboot *pboot bootblk raw-disk-device*

DESCRIPTION

The boot(1M) program, ufsboot, is loaded from disk by the bootblock program which resides in the boot area of a disk partition.

The ufs boot objects are platform-dependent, and reside in the /usr/platform/*platform-name*/lib/fs/ufs directory. The platform name can be found using the -i option of uname(1).

OPERANDS

bootblk The name of the bootblock code.

raw-disk-device The name of the disk device onto which the bootblock code is to be installed; it must be a character device which is readable and writable. Naming conventions for a SCSI or IPI drive are c?t?d?s? and c?d?s? for an IDE drive.

pboot The name of the partition boot file.

SPARC

To install a ufs bootblock on slice 0 of target 0 on controller 1 of the platform where the command is being run, use:

```
example# installboot /usr/platform/`uname -i`/lib/fs/ufs/bootblk \
    /dev/rdsk/c1t0d0s0
```

IA

To use installboot to install the ufs bootblock and partition boot program on a disk in an IA machine, you must specify slice 2 and that slice must be the entire disk. For example, to install the UFS bootblock on target 0, controller 1 of the platform where the command is being run, use:

```
example# installboot /usr/platform/`uname -i`/lib/fs/ufs/pboot \
        /usr/platform/`uname -i`/lib/fs/ufs/bootblk /dev/rdsk/c1t0d0s2
```

FILES

/usr/platform/*platform-name*/lib/fs/ufs
directory where ufs boot objects reside.

/platform/*platform-name*/ufsboot
second level program to boot from a disk or CD

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

od(1), uname(1), boot(1M), init(1M), kadb(1M), kernel(1M), reboot(1M), rpc.bootparamd(1M), init.d(4), attributes(5)

Solaris 9 Installation Guide

SPARC `monitor`(1M)

IA `fdisk`(1M), `fmthard`(1M)

WARNINGS The `installboot` utility fails if the *bootblk*, *pboot* or *openfirmware* files do not exist or if the raw disk device is not a character device.

NAME

installer – Solaris Web Start installer utility

SYNOPSIS

`installer` `[-locales` *list*`] [-nodisplay] [-noconsole] [-debug]`

DESCRIPTION

The `installer` utility invokes a Web Start install wizard sequence which will lead the user through a sequence of installation panels. This installer utility is found on many CDs that are shipped with Solaris and it will be found among the top level files of these CDs.

When the installer is on a CD being accessed from a desktop file manager, the installer can be double clicked to start the installation sequence. If the user is not currently the system's root user, the root user password will be requested.

The installer utility can also be run from other UNIX scripts. Usually, a script is used in conjunction with the utility's `-nodisplay` option. Add the `-noconsole` option for non-interactive scripts.

OPTIONS

The following options are supported:

`-locales` *list* — Selects product translations for install, corresponding to the specified list of locales if the specified locale translations are present on the installation media. Locales are supplied in a comma-separated list following the `-locales` option. An example *list* would appear as follows:

`installer -locales fr,de,it`This would install products with translations for the French, German, and Italian locales.

`-nodisplay` — Runs the install without a graphical user interface. Use the default product install unless it was modified by the `-locales` options.

`-noconsole` — Run the install without any interactive text console device. Useful when paired with `-nodisplay` for non-interactive UNIX script use.

`-debug` — Outputs extra information about what the install is doing. Mainly for install diagnostic purposes.

FILES

`/var/sadm/install/logs` — location of installation log files

SEE ALSO

`prodreg`(1M)

NAME

installf – add a file to the software installation database

SYNOPSIS

installf [-c *class*] [[-M] -R *root_path*] [-V *fs_file*] *pkginst pathname* [*ftype* [*major minor*] [*mode owner group*]]

installf [-c *class*] [[-M] -R *root_path*] [-V *fs_file*] *pkginst* –

installf -f [-c *class*] [[-M] -R *root_path*] [-V *fs_file*] *pkginst*

DESCRIPTION

`installf` informs the system that a pathname not listed in the `pkgmap`(4) file is being created or modified. It should be invoked before any file modifications have occurred.

When the second synopsis is used, the pathname descriptions will be read from standard input. These descriptions are the same as would be given in the first synopsis but the information is given in the form of a list. The descriptions should be in the form:

pathname [*ftype* [*major minor*] [*mode owner group*]]

After all files have been appropriately created and/or modified, `installf` should be invoked with the `-f` synopsis to indicate that installation is final. Links will be created at this time and, if attribute information for a pathname was not specified during the original invocation of `installf`, or was not already stored on the system, the current attribute values for the pathname will be stored. Otherwise, `installf` verifies that attribute values match those given on the command line, making corrections as necessary. In all cases, the current content information is calculated and stored appropriately.

OPTIONS

`-c` *class*	Class to which installed objects should be associated. Default class is `none`.
`-f`	Indicates that installation is complete. This option is used with the final invocation of `installf` (for all files of a given class).
`-M`	Instruct `installf` not to use the `$`*root_path*`/etc/vfstab` file for determining the client's mount points. This option assumes the mount points are correct on the server and it behaves consistently with Solaris 2.5 and earlier releases.
`-R` *root_path*	Define the full path name of a directory to use as the *root_path*. All files, including package system information files, are relocated to a directory tree starting in the specified *root_path*. The *root_path* may be specified when installing to a client from a server (for example, `/export/root/client1`).
`-V` *fs_file*	Specify an alternative *fs_file* to map the client's file systems. For example, used in situations where the `$`*root_path*`/etc/vfstab` file is non-existent or unreliable.

OPERANDS

pkginst Name of package instance with which the pathname should be associated.

pathname Pathname that is being created or modified.

ftype A one-character field that indicates the file type. Possible file types include:

`b`	block special device
`c`	character special device
`d`	directory
`e`	a file to be edited upon installation or removal
`f`	a standard executable or data file
`l`	linked file
`p`	named pipe
`s`	symbolic link
`v`	volatile file (one whose contents are expected to change)
`x`	an exclusive directory

major The major device number. The field is only specified for block or character special devices.

minor The minor device number. The field is only specified for block or character special devices.

mode The octal mode of the file (for example, 0664). A question mark (`?`) indicates that the mode will be left unchanged, implying that the file already exists on the target machine. This field is not used for linked or symbolically linked files.

owner The owner of the file (for example, `bin` or `root`). The field is limited to 14 characters in length. A question mark (`?`) indicates that the owner will be left unchanged, implying that the file already exists on the target machine. This field is not used for linked or symbolically linked files.

group The group to which the file belongs (for example, `bin` or `sys`). The field is limited to 14 characters in length. A question mark (`?`) indicates that the group will be left unchanged, implying that the file already exists on the target machine. This field is not used for linked or symbolically linked files.

EXAMPLES

EXAMPLE 1 The use of `installf`.

The following example shows the use of `installf`, invoked from an optional pre-install or post-install script:

```
#create /dev/xt directory
#(needs to be done before drvinstall)
installf $PKGINST /dev/xt d 755 root sys ||
    exit 2
majno=`/usr/sbin/drvinstall -m /etc/master.d/xt
     -d $BASEDIR/data/xt.o -v1.0` ||
    exit 2
i=00
while [ $i –lt $limit ]
do
    for j in 0 1 2 3 4 5 6 7
    do
        echo /dev/xt$i$j c $majno `expr $i ? 8 + $j`
             644 root sys |
        echo /dev/xt$i$j=/dev/xt/$i$j
    done
    i=`expr $i + 1`
    [ $i –le 9 ] && i="0$i" #add leading zero
done | installf $PKGINST – || exit 2
# finalized installation, create links
installf -f $PKGINST || exit 2
```

EXIT STATUS

`0` Successful operation.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`pkginfo`(1), `pkgmk`(1), `pkgparam`(1), `pkgproto`(1), `pkgtrans`(1), `pkgadd`(1M), `pkgask`(1M), `pkgchk`(1M), `pkgrm`(1M), `removef`(1M), `pkgmap`(4), `space`(4), `attributes`(5)

Application Packaging Developer's Guide

NOTES

When *ftype* is specified, all applicable fields, as shown below, must be defined:

ftype	Required Fields
`p`, `x`, `d`, `f`, `v`, or `e`	`mode owner group`
`c` or `b`	`major minor mode owner group`

The `installf` command will create directories, named pipes and special devices on the original invocation. Links are created when `installf` is invoked with the `-f` option to indicate installation is complete.

Links should be specified as *path1=path2. path1* indicates the destination and *path2* indicates the source file.

Files installed with `installf` will be placed in the class `none`, unless a class is defined with the command. Subsequently, they will be removed when the associated package is deleted. If this file should not be deleted at the same time as the package, be certain to assign it to a class which is ignored at removal time. If special action is required for the file before removal, a class must be defined with the command and an appropriate class action script delivered with the package.

When classes are used, `installf` must be used in one of the following forms:

```
installf -c class1 . . .
installf -f -c class1 . . .
installf -c class2 . . .
installf -f -c class2 . . .
```

NAME

install_scripts, add_install_client, add_to_install_server, modify_install_server, rm_install_client, setup_install_server, check – scripts used to install the Solaris software

SYNOPSIS

cdrom-mnt-pt`/Solaris_`*XX*`/Tools/add_install_client` [`-i` *IP_address*] [`-e` *Ethernet_address*] [`-s` *server_name* `:` *path*] [`-c` *server_name* `:` *path*] [`-n` [*server*] `:` *name_service* [`(` *netmask*]] [`-p` *server_name* `:` *path*] [`-f` *boot_file_name*] *host_name platform_group*

cdrom-mnt-pt`/Solaris_`*XX*`/Tools/add_install_client` `-d` [`-s` *server_name*`:`*path*] [`-c` *server_name*`:`*path*] [`-p` *server_name*`:`*path*] [`-t` *install_boot_image_path*] [`-f` *boot_file_name*] *platform_name platform_group*

cdrom-mnt-pt`/Solaris_`*XX*`/Tools/add_install_client` `-d` [`-s` *server_name*`:`*path*] [`-c` *server_name*`:`*path*] [`-p` *server_name*`:`*path*] [`-t` *install_boot_image_path*] [`-f` *boot_file_name*] `-e` *Ethernet_address platform_group*

cdrom-mnt-pt`/Solaris_`*XX*`/Tools/add_to_install_server` [`-s`] [`-p` *product_image_path*] *install_server_path*

cdrom-mnt-pt`/Solaris_`*XX*`/Tools/jumpstart_sample/check` [`-p` *install_dir_path*] [`-r` *rulesfile*]

cdrom-mnt-pt`/modify_install_server` [`-p`] *install_dir_path installer_miniroot_pat*`h`

cdrom-mnt-pt`/Solaris_`*XX*`/Tools/rm_install_client` *host_name*

cdrom-mnt-pt`/Solaris_`*XX*`/Tools/setup_install_server` [`-b`] [`-t` *install_boot_image_path*] *install_dir_path*

DESCRIPTION

These commands are located on slice `0` of the Solaris Software and Solaris Installer `CD`s. If the Solaris `CD` has been copied to a local disk, `cdrom_mnt_pt` is the path to the copied Solaris `CD`. They can be used for a variety of installation tasks.

The *XX* in Solaris_*XX* is the version number of the Solaris release being used.

There are three versions of the `add_install_client` command. See `SYNOPSIS`.

Use the following version of the `add_install_client` command to add clients for network installation (these commands update the `bootparams`(4) file). The `add_install_client` command must be run from the install server's Solaris installation image (a mounted Solaris CD or a Solaris CD copied to disk) or the boot server's boot directory (if a boot server is required). The Solaris installation image or the boot directory must be the same Solaris release that you want installed on the client.

cdrom-mnt-pt`/Solaris_`*XX*`/Tools/add_install_client` [`-i` *IP_address*] [`-e` *Ethernet_address*] [`-s` *server_name* `:` *path*] [`-c` *server_name* `:` *path*] [`-n` [*server*] `:` *name_service* [`(` *netmask*]] [`-p` *server_name* `:` *path*] *host_name platform_group*

Use the following version of the `add_install_client` command to add support for instances of a platform within a platform group to the install server. This group will be booted and configured using DHCP. The script will perform the necessary configuration steps on the server, and prints the data that the user needs to add to the DHCP server for the group.

cdrom-mnt-pt`/Solaris_`*XX*`/Tools/add_install_client -d [-s` *server*`:`*path*`] [-c` *server*`:`*path*`] [-p` *server*`:`*path*`] [-t` *install boot image path*`] [-f` *boot file name*`] platform_name platform_group`

Use the following version of the `add_install_client` command to add a single client to the install server. This client will be booted and configured using DHCP. The script will perform the necessary configuration steps on the server, and will print the data that the user needs to add to the DHCP server for the client. The `-f` flag used above needs to be added to the existing usage as well. `-f` allows the user to specify a boot file name to be used for a given client.

cdrom-mnt-pt`/Solaris_`*XX*`/Tools/add_install_client -d [-s` *server_name*`:`*path*`] [-c` *server_name*`:`*path*`] [-p` *server_name*`:`*path*`] [-t` *install_boot_image_path*`] [-f` *boot_file_name*`] -e` *Ethernet_address platform_group*

Note – Always use the `-d` option when registering Intel Architecture (IA) Pre-boot eXecution Environment (PXE) clients. These clients use DHCP for their configuration.

Use `add_to_install_server` to merge other Solaris CDs with an existing image on a Net Install Server. Each CD that can be merged (currently `OS CD 2`, and the `Language CD`) has its own `add_to_install_server` script. Do not use `add_to_install_server` scripts with CDs other than the ones with which they were delivered.

Use `check` to validate the rules in a `rules` file (this is only necessary if a custom JumpStart installation is being set up).

Use `modify_install_server` to replace an existing net install server's miniroot with a Solaris Installation `CD`'s miniroot. This will change the net install server's install time user interface over to the Solaris Installation `CD`'s Web Start user interface.

An existing install image (created using `setup_install_server`) must exist prior to using the `modify_install_server` command.

Use `rm_install_client` to remove clients for network installation (these commands update the `bootparams`(4) file).

Use `setup_install_server` to copy the Solaris CD to a disk (to set up an install server) or to copy just the boot software of the Solaris CD to a disk (to set up a boot server). An install server is required to install clients over the network. A boot server is also required for network installations if the install server and clients to be installed are on different subnets (the boot server must be located on the client's subnet).

OPTIONS

The `add_install_client` supports the following options:

`-c` *server_name*`:`*path*
This option is required only to specify a JumpStart directory for a custom JumpStart installation. *server_name* is the host name of the server with a JumpStart directory. *path* is the absolute path to the JumpStart directory.

`-d`
Specify as a DHCP client.

`-e` *Ethernet_address*
Specify the Ethernet address of the system to be installed.

`-f`
Specify the *boot_file_name* of the client to be installed.

`-i` *IP_address*
Specify the IP address of the client to be installed.

`-n` [*server*]`:` *name_service*[(*netmask*)]
This option specifies which name service should be used during system configuration. This sets the `ns` keyword in the `bootparams`(4) file.

name_service
Valid entries are `nis`, `nisplus`, and `none`.

netmask
A series of four numbers separated by periods, specifying which portion of an IP address is the network part, and which is the host part.

server
The name of the server or IP address of the specified name service. If the server specified is on a different subnet, then the *netmask* may be needed to enable the client to contact the server.

`-p` *server_name*`:` *path*
This option is the location of the user-defined `sysidcfg` file for pre-configuring system or network information. *server_name* is either a valid host name or IP address. *path* is the absolute path to the Jumpstart directory.

`-s` *server_name*`:`*path*
This option is required only when using `add_install_client` from a boot server. Specify the name of the server and the absolute path of the Solaris installation image that will be used for this installation. *path* is either the path to a mounted Solaris `CD` or a path to a directory with a copy of the Solaris `CD`.

The `add_to_install_server` command supports the following options:

`-p`
Specifies the location of the CD (containing the supplemental products) to be copied.

`-s`
Allows users to select from a list only the products needing installation.

The `check` command supports the following options:

`-p` *install_dir_path*
Validates the rules file by using the `check` script from a specified Solaris installation image, instead of the `check` script from the system you are using. `install_dir_path` is the path to a Solaris installation image on a local disk or a mounted Solaris `CD`.

Use this option to run the most recent version of `check` if your system is running a previous version of Solaris.

`-r` *rulesfile*
Specifies a rules file other than the one named `rules`. Using this option, the validity of a rule can be tested before integrating it into the rules file. `check` will report whether or not the rule is valid, but it will not create the `rules.ok` file necessary for a custom JumpStart installation.

The `modify_install_server` command supports the following options:

`-p`
This option preserves the existing images miniroot in *install_dir_path*`/Solaris_`*XX*`/Tools/Boot.orig`.

The `setup_install_server`command supports the following options:

`-b`
This option sets up the server only as a boot server.

`-t`
This option allows an alternate miniroot to be specified.

OPERANDS

The `add_install_client` command supports the following options:

host_name
This is the name of the client to be installed.

platform_group
Vendor-defined grouping of hardware platforms for the purpose of distributing specific software. Examples of valid platform groups are:

System	Platform Group
IA	`i86pc`
SPARCstation 1+	`sun4c`
SPARCstation 5	`sun4m`

Use the `uname`(1) command (with the `-m` option) to determine a system's platform group.

platform_name
Use the `uname`(1) command (with the `-i` option) to determine a system's platform name.

The following example shows the use of the `uname` command to determine the system platform name for an Ultra 1:

```
uname -i
```
The system responds with:

```
SUNW,Ultra-1
```
Therefore, the system's platform name is `SUNW,Ultra1`.

The following command calls `add_install_client` for Ultra 1s:

```
add_install_client -d SUNW,Ultra-1 sun4u
```

For `IA32` platforms, the platform name is always `SUNW.i86pc`.

The following command calls add_install_client for `IA32` platforms:

```
add_install_client -d SUNW.i86pc i86pc
```

The rm_install_client command supports the following operands:

host_name
This is the name of the client to be removed.

The setup_install_server command supports the following options:

install_dir_path
The absolute path of the directory in which the Solaris software is to be copied. The directory must be empty.

EXAMPLES

EXAMPLE 1 Using `add_install_client`

The following `add_install_client` commands add clients for network installation from a mounted Solaris CD on an install server on Solaris 9:

```
example# cd /cdrom/cdrom0/s0/Solaris_9/Tools
example# ./add_install_client system_1 sun4c
example# ./add_install_client system_2 sun4m
```

EXAMPLE 2 Using `add_install_client`

The following `add_install_client` commands add clients for network installation from a mounted Solaris CD on an install server. The `-c` option specifies a server and path to a JumpStart directory that has a rules file and a profile file for performing a custom JumpStart installation. Also, the Solaris CD has been copied to the `/export/install` directory on Solaris 9:

```
example# cd /export/install/Solaris_9/Tools
example# /add_install_client -c install_server:/jumpstart system_1 i86pc
example# ./add_install_client -c install_server:/jumpstart system_2 i86pc
```

EXAMPLE 2 Using `add_install_client` *(Continued)*

EXAMPLE 3 Using `add_install_client`

The following `add_install_client` command adds support for a specific `sun4u` platform machine (`8:0:20:99:88:77`) using the boot file: `sun4u.solaris8`.

```
example# add_install_client -d -f sun4u.solaris8 -e 8:0:20:99:88:77 sun4u
```

EXAMPLE 4 Using `add_install_client`

The following `add_install_client` command adds IA clients that use the PXE standard for network booting:

```
example# add_install_client -d -s svrname:/mnt/export/root SUNW.i86pc i86p
```

EXAMPLE 5 Using `add_to_install_server`

The following `add_to_install_server` command copies the packages in all the CD's products directories to an existing install server on Solaris 9:

```
example# cd /cdrom/cdrom0/s0
example# ./add_to_install_server /export/Solaris_9
```

EXAMPLE 6 Using `check`

The following `check` command validates the syntax of the rules file used for a custom JumpStart installation:

```
example# cd jumpstart_dir_path
example# ./check -p /cdrom/cdrom0/s0
```

EXAMPLE 7 Using `modify_install_server`

The following `modify_install_server` command moves the miniroot created using the above `setup_install_server` to `Boot.orig` and replaces it with the miniroot on the Solaris Installer `CD`.

```
example# cd /cdrom/cdrom0/s0
example# ./modify_install_server -p /export/install /cdrom/cdrom0/s1
```

EXAMPLE 8 Using `modify_install_server`

The following `modify_install_server` command replaces the miniroot created using the above `setup_install_server` with the miniroot on the Solaris Installer `CD`.

```
example# cd /cdrom/cdrom0/s0
example# ./modify_install_server /export/install /cdrom/cdrom0/s1
```

EXAMPLE 9 Using `rm_install_client`

The following `rm_install_client` commands remove clients for network installation on Solaris 9:

EXAMPLE 9 Using `rm_install_client` *(Continued)*

```
example# cd /export/install/Solaris_9/Tools
example# ./rm_install_client holmes
example# ./rm_install_client watson
```

EXAMPLE 10 Using `setup_install_server`

The following `setup_install_server` command copies the mounted Solaris CD to a directory named `/export/install` on the local disk on Solaris 9:

```
example# cd /cdrom/cdrom0/s0/Solaris_9/Tools
example# ./setup_install_server /export/install
```

EXAMPLE 11 Using `setup_install_server`

The following `setup_install_server` command copies the boot software of a mounted Solaris CD to a directory named `/boot_dir` on a system that is going to be a boot server for a subnet on Solaris 9:

```
example# cd /cdrom/cdrom0/s0/Solaris_9/Tools
example# ./setup_install_server -b /boot_dir
```

EXAMPLE 12 Using `setup_install_server`

By default, `setup_install_server` will look for an installation boot directory at the Solaris ../Tools/Boot location of the mount Solaris distribution disc.

If an alternate boot directory is required, such as one saved on a > network boot server by way of an earlier `./setup_install_server -b /boot_dir` command, the `-t` option can be used.

```
example# cd /cdrom/cdrom0/s0/Solairs_9/Tools
example# ./setup_install_server -t /boot_dir /export/install
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` An error has occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	Solaris CD

SEE ALSO

`uname`(1), `bootparams`(4), `attributes`(5)

Solaris Installation Guide

NAME

in.talkd, talkd – server for talk program

SYNOPSIS

```
in.talkd
```

DESCRIPTION

`talkd` is a server used by the `talk`(1) program. It listens at the UDP port indicated in the "talk" service description; see `services`(4). The actual conversation takes place on a TCP connection that is established by negotiation between the two machines involved.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWrcmds

SEE ALSO

`talk`(1), `inetd`(1M), `services`(4), `attributes`(5)

NOTES

The protocol is architecture dependent.

NAME

in.telnetd, telnetd – DARPA TELNET protocol server

SYNOPSIS

/usr/sbin/in.telnetd

DESCRIPTION

`in.telnetd` is a server that supports the DARPA standard TELNET virtual terminal protocol. `in.telnetd` is normally invoked in the internet server (see `inetd`(1M)), for requests to connect to the TELNET port as indicated by the `/etc/services` file (see `services`(4)).

`in.telnetd` operates by allocating a pseudo-terminal device for a client, then creating a login process which has the slave side of the pseudo-terminal as its standard input, output, and error. `in.telnetd` manipulates the master side of the pseudo-terminal, implementing the TELNET protocol and passing characters between the remote client and the login process.

When a TELNET session starts up, `in.telnetd` sends TELNET options to the client side indicating a willingness to do *remote* `echo` of characters, and to *suppress go ahead*. The pseudo-terminal allocated to the client is configured to operate in "cooked" mode, and with XTABS, ICRNL and ONLCR enabled. See `termio`(7I).

`in.telnetd` is willing to do: *echo*, *binary*, *suppress go ahead*, and *timing mark*. `in.telnetd` is willing to have the remote client do: *binary*, *terminal type*, *terminal size*, *logout option*, and *suppress go ahead*.

`in.telnetd` also allows environment variables to be passed, provided that the client negotiates this during the initial option negotiation. The DISPLAY environment variable may be sent this way, either by the TELNET general environment passing methods, or by means of the XDISPLOC TELNET option. DISPLAY can be passed in the environment option during the same negotiation where XDISPLOC is used. Note that if you use both methods, use the same value for both. Otherwise, the results may be unpredictable.

These options are specified in Internet standards RFC 1096, RFC 1408, RFC 1571, and RFC 1572.

The banner printed by `in.telnetd` is configurable. The default is (more or less) equivalent to "`uname -sr`" and will be used if no banner is set in `/etc/default/telnetd`. To set the banner, add a line of the form

`BANNER="..."`to `/etc/default/telnetd`. Nonempty banner strings are fed to shells for evaluation. The default banner may be obtained by

`BANNER="\\r\\n\\r\\n`uname -s` `uname -r`\\r\\n\\r\\n"`and no banner will be printed if `/etc/default/telnetd` contains

```
BANNER=""
```

USAGE

`telnetd` and `in.telnetd` are IPv6–enabled. See `ip6`(7P).

SECURITY

`in.telnetd` uses `pam`(3PAM) for authentication, account management, session management, and password management. The PAM configuration policy, listed through `/etc/pam.conf`, specifies the modules to be used for `in.telnetd`. Here is a partial `pam.conf` file with entries for the `telnet` command using the UNIX authentication, account management, session management, and password management modules.

```
telnet  auth requisite          pam_authtok_get.so.1
telent  auth required           pam_dhkeys.so.1
telent  auth required           pam_unix_auth.so.1

telnet  account requisite       pam_roles.so.1
telnet  account required        pam_projects.so.1
telnet  account required        pam_unix_account.so.1

telnet  session required        pam_unix_session.so.1

telnet  password required       pam_dhkeys.so.1
telent  password requisite      pam_authtok_get.so.1
telnet  password requisite      pam_authtok_check.so.1
telnet  password required       pam_authtok_store.so.1
```

If there are no entries for the `telnet` service, then the entries for the "other" service will be used. If multiple authentication modules are listed, then the user may be prompted for multiple passwords.

FILES

`/etc/default/telnetd`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtnetd

SEE ALSO

`telnet`(1), `inetd`(1M), `pam`(3PAM), `inetd.conf`(4)`pam.conf`(4), `services`(4), `attributes`(5), `pam_authtok_check`(5), `pam_authtok_get`(5), `pam_authtok_store`(5), `pam_dhkeys`(5), `pam_passwd_auth`(5), `pam_unix`(5), `pam_unix_account`(5), `pam_unix_auth`(5), `pam_unix_session`(5), `ip6`(7P), `termio`(7I)

Alexander, S., *TELNET Environment Option*, RFC 1572, Network Information Center, SRI International, Menlo Park, Calif., January 1994.

Borman, Dave, *TELNET Environment Option*, RFC 1408, Network Information Center, SRI International, Menlo Park, Calif., January 1993.

Borman, Dave, *TELNET Environment Option Interoperability Issues*, RFC 1571, Network Information Center, SRI International, Menlo Park, Calif., January 1994.

Crispin, Mark, *TELNET Logout Option*, RFC 727, Network Information Center, SRI International, Menlo Park, Calif., April 1977.

Marcy, G., *TELNET X Display Location Option*. RFC 1096, Network Information Center, SRI International, Menlo Park, Calif., March 1989.

Postel, Jon, and Joyce Reynolds, *TELNET Protocol Specification*, RFC 854, Network Information Center, SRI International, Menlo Park, Calif., May 1983.

Waitzman, D., *TELNET Window Size Option*, RFC 1073, Network Information Center, SRI International, Menlo Park, Calif., October 1988.

NOTES

Some TELNET commands are only partially implemented.

Binary mode has no common interpretation except between similar operating systems.

The terminal type name received from the remote client is converted to lower case.

The *packet* interface to the pseudo-terminal should be used for more intelligent flushing of input and output queues.

`in.telnetd` never sends TELNET *go ahead* commands.

The `pam_unix`(5) module might not be supported in a future release. Similar functionality is provided by `pam_authtok_check`(5), `pam_authtok_get`(5), `pam_authtok_store`(5), `pam_dhkeys`(5), `pam_passwd_auth`(5), `pam_unix_account`(5), `pam_unix_auth`(5), and `pam_unix_session`(5).

NAME

in.tftpd, tftpd – Internet Trivial File Transfer Protocol server

SYNOPSIS

in.tftpd [-s] [*homedir*]

DESCRIPTION

tftpd is a server that supports the Internet Trivial File Transfer Protocol (TFTP). This server is normally started by inetd(1M) and operates at the port indicated in the tftp Internet service description in the /etc/inetd.conf file. By default, the entry for in.tftpd in etc/inetd.conf is commented out. To make in.tftpd operational, the comment character(s) must be deleted from the file. See inetd.conf(4).

Before responding to a request, the server attempts to change its current directory to *homedir*; the default directory is /tftpboot.

The use of tftp does not require an account or password on the remote system. Due to the lack of authentication information, in.tftpd will allow only publicly readable files to be accessed. Files may be written only if they already exist and are publicly writable. Note that this extends the concept of "public" to include all users on all hosts that can be reached through the network. This may not be appropriate on all systems, and its implications should be considered before enabling this service.

in.tftpd runs with the user ID and group ID set to [GU]ID_NOBODY under the assumption that no files exist with that owner or group. However, nothing checks this assumption or enforces this restriction.

OPTIONS

-d Debug. When specified it sets the SO_DEBUG socket option.

-s Secure. When specified, the directory change to *homedir* must succeed. The daemon also changes its root directory to *homedir*.

FILES

/etc/inetd.conf

USAGE

The in.tftpd server is IPv6–enabled. See ip6(7P).

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtftp

SEE ALSO

tftp(1), inetd(1M), inetd.conf(4), netconfig(4), attributes(5), ip6(7P)

Malkin, G. and Harkin, A. *RFC 2347, TFTP Option Extension*. The Internet Society. May 1998

Malkin, G. and Harkin, A. *RFC 2348, TFTP Blocksize Option*. The Internet Society. May 1998

Malkin, G. and Harkin, A. *RFC 2349, TFTP Timeout Interval and Transfer Size Options*. The Internet Society. May 1998

Sollins, K.R. *RFC 1350, The TFTP Protocol (Revision 2)*. Network Working Group. July 1992.

NOTES

The `tftpd` server only acknowledges the transfer size option that is sent with a `read` request when the octet transfer mode is specified.

NAME

in.tnamed, tnamed – DARPA trivial name server

SYNOPSIS

/usr/sbin/in.tnamed [-v]

DESCRIPTION

`in.tnamed` is a server that supports the DARPA Name Server Protocol. The name server operates at the port indicated in the "name" service description (see `services`(4)), and is invoked by `inetd`(1M) when a request is made to the name server.

OPTIONS

`-v` Invoke the daemon in verbose mode.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtnamd

SEE ALSO

`uucp`(1C), `inetd`(1M), `services`(4), `attributes`(5)

Postel, Jon, *Internet Name Server*, IEN 116, SRI International, Menlo Park, California, August 1979.

NOTES

The protocol implemented by this program is obsolete. Its use should be phased out in favor of the Internet Domain Name Service (DNS) protocol.

NAME

in.uucpd, uucpd – UUCP server

SYNOPSIS

/usr/sbin/in.uucpd [-n]

DESCRIPTION

in.uucpd is the server for supporting UUCP connections over networks.

in.uucpd is invoked by inetd(1M) when a UUCP connection is established (that is, a connection to the port indicated in the "uucp" service specification) and executes the following protocol. See services(4):

1. The server prompts with login:. The uucico(1M) process at the other end must supply a username.
2. Unless the username refers to an account without a password, the server then prompts with Password:. The uucico process at the other end must supply the password for that account.

If the username is not valid, or is valid but refers to an account that does not have /usr/lib/uucp/uucico as its login shell, or if the password is not the correct password for that account, the connection is dropped. Otherwise, uucico is run, with the user ID, group ID, group set, and home directory for that account, with the environment variables USER and LOGNAME set to the specified username, and with a -u flag specifying the username. Unless the -n flag is specified, entries are made in /var/adm/utmpx, /var/adm/wtmpx, and /var/adm/lastlog for the username. in.uucpd must be invoked by a user with appropriate privilege (usually root) in order to be able to verify that the password is correct.

SECURITY

in.uucpd uses pam(3PAM) for authentication, account management, and session management. The PAM configuration policy, listed through /etc/pam.conf, specifies the modules to be used for in.uucpd. Here is a partial pam.conf file with entries for uucp using the UNIX authentication, account management, and session management module.

```
uucp    auth requisite          pam_authtok_get.so.1
uucp    auth required           pam_dhkeys.so.1
uucp    auth required           pam_unix_auth.so.1

uucp    account requisite       pam_roles.so.1
uucp    account required        pam_projects.so.1
uucp    account required        pam_unix_account.so.1

uucp      session required      pam_unix_session.so.1
```

If there are no entries for the uucp service, then the entries for the "other" service will be used. If multiple authentication modules are listed, then the peer may be prompted for multiple passwords.

FILES

/var/adm/utmpx	accounting
/var/adm/wtmpx	accounting
/var/adm/lastlog	time of last login

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWbnuu

SEE ALSO

`inetd`(1M), `uucico`(1M), `pam`(3PAM), `pam.conf`(4), `services`(4), `attributes`(5), `pam_authtok_check`(5), `pam_authtok_get`(5), `pam_authtok_store`(5), `pam_dhkeys`(5), `pam_passwd_auth`(5), `pam_unix`(5), `pam_unix_account`(5), `pam_unix_auth`(5), `pam_unix_session`(5)

DIAGNOSTICS

All diagnostic messages are returned on the connection, after which the connection is closed.

`user read`	An error occurred while reading the username.
`passwd read`	An error occurred while reading the password.
`Login incorrect.`	The username is invalid or refers to an account with a login shell other than `/usr/lib/uucp/uucico`, or the password is not the correct password for the account.

NOTES

The `pam_unix`(5) module might not be supported in a future release. Similar functionality is provided by `pam_authtok_check`(5), `pam_authtok_get`(5), `pam_authtok_store`(5), `pam_dhkeys`(5), `pam_passwd_auth`(5), `pam_unix_account`(5), `pam_unix_auth`(5), and `pam_unix_session`(5).

NAME

iostat – report I/O statistics

SYNOPSIS

/usr/bin/iostat [-cCdDeEiImMnpPrstxXz] [-l *n*] [-T u | d] [*disk*...] [*interval* [*count*]]

DESCRIPTION

The `iostat` utility iteratively reports terminal, disk, and tape I/O activity, as well as CPU utilization. The first line of output is for all time since boot; each subsequent line is for the prior interval only.

To compute this information, the kernel maintains a number of counters. For each disk, the kernel counts reads, writes, bytes read, and bytes written. The kernel also takes hi-res time stamps at queue entry and exit points, which allows it to keep track of the residence time and cumulative residence-length product for each queue. Using these values, `iostat` produces highly accurate measures of throughput, utilization, queue lengths, transaction rates and service time. For terminals collectively, the kernel simply counts the number of input and output characters.

During execution of this kernel status command, the "state" of the kernel can change. An example would be CPUs going online or offline. `iostat` reports this as one or more of the following messages:

device_name added
device_name removed
NFS_filesystem mounted
NFS_filesystem unmounted
cpu[s] taken offline: *cpuid*
cpu[s] brought online: *cpuid*

where *device_name*, *NFS_filesystem* and *cpuid* are replaced with the actual name or names of the entities formatted according to other options.

For more general system statistics, use `sar`(1), `sar`(1M), or `vmstat`(1M).

OPTIONS

The `iostat` utility's activity class options default to `tdc` (terminal, disk, and CPU). If any activity class options are specified, the default is completely overridden. Therefore, if only `-d` is specified, neither terminal nor CPU statistics will be reported. The last disk option specified (`-d`, `-D`, or `-x`) is the only one that is used.

The following options are supported:

`-c` Report the percentage of time the system has spent in user mode, in system mode, waiting for I/O, and idling.

`-C` When the `-n` and `-x` options are also selected, report extended disk statistics aggregated by *controller id*.

`-d` For each disk, report the number of kilobytes transferred per second, the number of transfers per second, and the average service time in milliseconds.

`-D`	For each disk, report the reads per second, writes per second, and percentage disk utilization.
`-e`	Display device error summary statistics. The total errors, hard errors, soft errors, and transport errors are displayed.
`-E`	Display all device error statistics.
`-i`	In `-E` output, display the "Device Id" instead of the "Serial No". The "Device Id" is a unique identifier registered by a driver through `ddi_devid_register`(9F).
`-I`	Report the counts in each interval, rather than rates (where applicable).
`-l` *n*	Limit the number of disks included in the report to *n*; the disk limit defaults to 4 for `-d` and `-D`, and unlimited for `-x`. Note: disks explicitly requested (see *disk* below) are not subject to this disk limit.
`-m`	Report file system mount points. This option is most useful if the `-P` or `-p` option is also specified.
`-M`	Display data throughput in MB/sec instead of KB/sec.
`-n`	Display names in descriptive format (for example, `cXtYdZ`, `rmt/N`, `server:/export/path`).
`-p`	For each disk, report per-partition statistics in addition to per-device statistics.
`-P`	For each disk, report per-partition statistics only, no per-device statistics.
`-r`	Display data in a comma-separated format.
`-s`	Suppress messages related to "state changes."
`-t`	Report the number of characters read and written to terminals per second.
`-T u | d`	Display a time stamp. Specify `u` for a printed representation of the internal representation of time. See `time`(2). Specify `d` for standard date format. See `ctime`(3C).
`-x`	For each disk, report extended disk statistics. The output is in tabular form.
`-X`	For disks under `scsi_vhci` control, also report statistics in the form of *target*`.`*controller*.
`-z`	Do not print lines whose underlying data values are all zeros.

OPERANDS

The following operands are supported:

disk	Explicitly specify the disks to be reported; in addition to any explicit disks, any active disks up to the disk limit (see `-l` above) will also be reported.
count	Display only *count* reports.
interval	Report once each *interval* seconds.

EXAMPLES

EXAMPLE 1 Using the `iostat` command

```
example% iostat -xtc 5 2
                  extended device statistics        tty          cpu
device r/s   w/s kr/s   kw/s wait actv svc_t %w  %b  tin tout  us sy wt id
sd0    0.4   0.3 10.4    8.0  0.0  0.0  36.9  0   1    0   10   0  0  1 99
sd1    0.0   0.0  0.3    0.4  0.0  0.0  35.0  0   0
sd6    0.0   0.0  0.0    0.0  0.0  0.0   0.0  0   0
nfs1   0.0   0.0  0.0    0.0  0.0  0.0   0.0  0   0
nfs2   0.0   0.0  0.0    0.1  0.0  0.0  35.6  0   0
             extended device statistics              tty          cpu
device r/s   w/s   kr/s   kw/s wait actv svc_t %w  %b tin tout  us sy wt id
sd0    0.0   0.0   0.0    0.0  0.0  0.0   0.0   0   0   0  155   0  0  0 100
sd1    0.0   0.0   0.0    0.0  0.0  0.0   0.0   0   0
sd6    0.0   0.0   0.0    0.0  0.0  0.0   0.0   0   0
nfs1   0.0   0.0   0.0    0.0  0.0  0.0   0.0   0   0
nfs2   0.0   0.0   0.0    0.0  0.0  0.0   0.0   0   0
```

`device`	name of the disk
`r/s`	reads per second
`w/s`	writes per second
`Kr/s`	kilobytes read per second
`Kw/s`	kilobytes written per second
`wait`	average number of transactions waiting for service (queue length)
`actv`	average number of transactions actively being serviced (removed from the queue but not yet completed)
`svc_t`	average service time, in milliseconds
`%w`	percent of time there are transactions waiting for service (queue non-empty)
`%b`	percent of time the disk is busy (transactions in progress)

EXAMPLE 2 Using the `iostat` command

```
example% iostat -xnp                extended device statistics
r/s  w/s  kr/s kw/s wait actv wsvc_t asvc_t %w %b device
```

EXAMPLE 2 Using the `iostat` command *(Continued)*

```
0.4  0.3  10.4  7.9  0.0  0.0    0.0   36.9  0  1 c0t0d0
0.3  0.3   9.0  7.3  0.0  0.0    0.0   37.2  0  1 c0t0d0s0
0.0  0.0   0.1  0.5  0.0  0.0    0.0   34.0  0  0 c0t0d0s1
0.0  0.0   0.0  0.1  0.0  0.0    0.6   35.0  0  0 expositor:/export/home3/user3
```

The fields have the same meanings as in the previous example, with the following additions:

`wsvc_t`	average service time in wait queue, in milliseconds
`asvc_t`	average service time active transactions, in milliseconds

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`sar`(1), `sar`(1M), `vmstat`(1M), `time`(2), `ctime`(3C), `attributes`(5), `scsi_vhci`(7D)

System Administration Guide: Basic Administration

NOTES

The sum of CPU utilization might vary slightly from 100 because of rounding errors in the production of a percentage figure.

NAME

ipsecconf – configure system wide IPsec policy

SYNOPSIS

/usr/sbin/ipsecconf

/usr/sbin/ipsecconf `-a` *file* `[-q]`

/usr/sbin/ipsecconf `-d` *index*

/usr/sbin/ipsecconf `-f`

/usr/sbin/ipsecconf `-l [-n]`

DESCRIPTION

The `ipsecconf` utility configures the IPsec policy for a host. Once the policy is configured, all outbound and inbound datagrams are subject to policy checks as they exit and enter the host. If no entry is found, no policy checks will be completed, and all the traffic will pass through. Datagrams that are being forwarded will not be subjected to policy checks that are added using this command. See `ifconfig`(1M) and `tun`(7M) for information on how to protect forwarded packets. Depending upon the match of the policy entry, a specific action will be taken.

This command can be run only by superuser.

Each entry can protect traffic in either one direction (requiring a pair of entries) or by a single policy entry which installs the needed symmetric `sadb` rules.

When the command is issued without any arguments, the list of (file policy entries) loaded are shown. To display the (`spd p.e.s`) use the `-l` option. Both will display the index number for the entry.

Note, since one file policy entry (FPE) can generate multiple SPD pol entries (SPEs), the list of FPEs may not show all the actual entries. However, it is still useful in determining what what rules have been added to get the spd into its current state.

You can use the `-d` option with the index to delete a given policy in the system. If the `-d` option removes an FPE entry that produces multiple SPEs, only then SPD with the same policy index as the FPE will be removed. This can produce a situation where there may be SPEs when there are no FPEs

With no options, the entries are displayed in the order that they were added, which is not necessarily the order that the traffic match will take place.

To view the order in which the traffic match will take place, use the `-l` option. The rules are ordered such that all bypass rules are checked first, then ESP rules, then AH rules. After that, they are checked in the order entered.

Policy entries are not preserved across reboot. Thus the policy needs to be added everytime the machine reboots. To configure policies early in the boot, one can setup policies in the `/etc/inet/ipsecinit.conf` file, which are then read from the `inetinit` startup script.

See `SECURITY` for issues in securing this file.

OPTIONS `ipsecconf` supports the following option:

`-a` *file*
: Add the IPsec policy to the system as specified by each entry in the file. An IPsec configuration file contains one or more entries that specify the configuration. Once the policy is added, all outbound and inbound datagrams are subject to policy checks.

  Entries in the files are described in the OPERANDS section below. Examples can be found in the EXAMPLES section below.

  Policy is latched for TCP/UDP sockets on which a `connect`(3SOCKET) or `accept`(3SOCKET) is issued. So, the addition of new policy entries may not affect such endpoints or sockets. However, the policy will be latched for a socket with an existing non-null policy. Thus, make sure that there are no preexisting connections that will be subject to checks by the new policy entries.

  The feature of policy latching explained above may change in the future. It is not advisable to depend upon this feature.

`-d` *index*
: Delete the policy denoted by the index. The index is obtained by invoking ipsecconf without any arguments, or with the `-l` option. See `DESCRIPTION` for more information. Once the entry is deleted, all outbound and inbound datagrams affected by this policy entry will not be subjected to policy checks. Be advised that with connections for which the policy has been latched, packets will continue to go out with the same policy, even if it has been deleted. It is advisable to use the `-l` option to find the correct policy index.

`-f`
: Flush all the policies in the system. Constraints are similar to the `-d` option with respect to latching.

`-l`
: Listing of the internal system policy table. When `ipsecconf` is invoked without any arguments, a complete list of policy entries with indexes added by the user since boot is displayed. The current table can differ from the previous one if, for example, a multi-homed entry was added or policy reordering occurred, or if a single rule entry generates two `spd` rules In the case of a multi-homed entry, all the addresses are listed explicitly. If a mask was not specified earlier but was instead inferred from the address, it will be explicitly listed here. This option is used to view policy entries in the correct order. The outbound and inbound policy entries are listed separately.

`-n`
: Show network addresses, ports, protocols in numbers. The `-n` option may only be used with the `-l` option.

`-q` Quiet mode. Suppresses the warning message generated when adding policies.

OPERANDS Each policy entry contains 3 parts specified as follows:

```
{pattern} action {properties}
```

or

```
{pattern} action {properties} ["or" action {properties}]*
```

Every policy entry begins on a new line and can span multiple lines. *pattern* specifies the traffic pattern that should be matched against the outbound and inbound datagrams. If there is a match, a specific *action* determined by the second argument will be taken, depending upon the *properties* of the policy entry.

If there is an `or` in the rule (multiple action-properties for a given pattern), a transmitter will use the first action-property pair that works, while a receiver will use any that are acceptable.

pattern and *properties* are name-value pairs where name and value are separated by a <space>, <tab> or <newline>. Multiple name-value pairs should be separated by <space>, <tab> or <newline>. The beginning and end of the pattern and properties are marked by { and } respectively.

Files can contain multiple policy entries. An unspecified name-value pair in the *pattern* will be considered as a wildcard. Wildcard entries match any corresponding entry in the datagram.

One thing to remember is that UDP port 500 is always bypassed regardless of any policy entries. This is a requirement for `in.iked`(1M) to work.

File can be commented by using a # as the first character. Comments may be inserted either at the beginning or the end of a line.

The complete syntax of a policy entry is:

```
policy ::= { <pattern1> } <action1> { <properties1> } |
     { <pattern2> } <action2> { <properties2> }
     [ 'or' <action2> { <properties2>} ]*

     pattern1 ::=  <pattern_name_value_pair1>*

     pattern2 ::=  <pattern_name_value_pair2>*

     action1 ::= apply | permit | bypass | pass
     action2 ::=  bypass | pass | drop | ipsec

     properties1 ::=   {<prop_name_value_pair1>}
     properties2 ::=   {<prop_name_value_pair2>}


     pattern_name_value_pair1 ::=
```

```
   saddr <address>/<prefix> |
   src  <address>/<prefix> |
   srcaddr <address>/<prefix> |
   smask <mask> |
   sport <port> |
   daddr <address>/<prefix> |
   dst <address>/<prefix> |
   dstaddr <address>/<prefix> |
   dmask <mask> |
   dport <port> |
   ulp <protocol> |
   proto <protocol>

pattern_name_value_pair2 ::=
   raddr <address>/<prefix> |
   remote  <address>/<prefix> |
   rport <port> |
   laddr <address>/<prefix> |
   local <address>/<prefix> |
   lport <port> |
   ulp <protocol> |
   proto <protocol>  |
   dir <dir_val2>

address ::=  <IPv4 dot notation> | <IPv6 colon notation> |
             <String recognized by gethostbyname>|
             <String recognized by getnetbyname>

prefix ::=  <number>

mask ::= <0xhexdigit[hexdigit]> | <0Xhexdigit[hexdigit]> |
         <IPv4 dot notation>

port ::= <number>| <String recognized by getservbyname>

protocol ::=  <number>| <String recognized by getprotobyname>

prop_name_value_pair1 ::=
     auth_algs <auth_alg> |
     encr_algs <encr_alg> |
     encr_auth_algs <auth_alg> |
     sa <sa_val> |
     dir <dir_val1>

prop_name_value_pair2 ::=
     auth_algs <auth_alg> |
     encr_algs <encr_alg> |
     encr_auth_algs <auth_alg> |
     sa <sa_val>

auth_alg ::=  <auth_algname> ['(' <keylen> ')']
auth_algname ::= any | md5 | hmac-md5 | sha | sha1 | hmac-sha |
                 hmac-sha1 | <number>

encr_alg ::= <encr_algname> ['(' <keylen> ')']
encr_algname ::= any | aes | aes-cbc | des | des-cbc | 3des |
                 3des-cbc | blowfish | blowfish-cbc | <number>
```

```
keylen ::= <number> | <number>'..' | '..'<number> | <number>'..'<number>

sa_val ::= shared | unique

dir_val1 ::= out | in
dir_val2 ::= out | in | both

number ::= < 0 | 1 | 2 ... 9> <number>
```

Policy entries may contain the following (name value) pairs in the *pattern* field. Each (name value) pair may appear only once in given policy entry.

laddr/plen
local/plen
: The value that follows is the local address of the datagram with the prefix length. Only plen leading bits of the source address of the packet will be matched. plen is optional. Local means destination on incoming and source on outgoing packets. The source address value can be a hostname as described in getaddrinfo(3XSOCKET) or a network name as described in `getnetbyname`(3XNET) or a host address or network address in the Internet standard dot notation. See `inet_addr`(3XNET). If a hostname is given and getaddrinfo(3XSOCKET) returns multiple addresses for the host, then policy will be added for each of the addresses with other entries remaining the same.

raddr/plen
remote/plen
: The value that follows is the remote address of the datagram with the prefix length. Only plen leading bits of the remote address of the packet will be matched. plen is optional. Remote means source on incoming packets and destination on outgoing packets. The remote address value can be a hostname as described in `getaddrinfo`(3SOCKET) or a network name as described in `getnetbyname`(3XNET) or a host address or network address in the Internet standard dot notation. See `inet_addr`(3XNET). If a hostname is given and `getaddrinfo`(3SOCKET) returns multiple addresses for the host, then policy will be added for each of the addresses with other entries remaining the same.

src/plen
srcaddr/plen
saddr/plen
: The value that follows is the source address of the datagram with the prefix length. Only *plen* leading bits of the source address of the packet will be matched. *plen* is optional.

  The source address value can be a hostname as described in `getaddrinfo`(3XSOCKET) or a network name as described in `getnetbyname`(3XNET) or a host address or network address in the Internet standard dot notation. See `inet_addr`(3XNET).

If a hostname is given and `getaddrinfo`(3XSOCKET) returns multiple addresses for the host, then policy will be added for each of the addresses with other entries remaining the same.

daddr/plen
dest/plen
dstaddr/plen

The value that follows is the destination address of the datagram with the prefix length. Only *plen* leading bits of the destination address of the packet will be matched. *plen* is optional.

See *saddr* for valid values that can be given. If multiple source and destination addresses are found, then a policy entry that covers each source address-destination address pair will be added to the system.

smask

For IPv4 only. The value that follows is the source mask. If prefix length is given with *saddr*, this should not be given. This can be represented either in hexadecimal number with a leading `0x` or `0X`, for example, `0xffff0000`, `0Xffff0000` or in the Internet decimal dot notation, for example, `255.255.0.0` and `255.255.255.0`. The mask should be contiguous and the behavior is not defined for non-contiguous masks.

smask is considered only when *saddr* is given.

For both IPv4 and IPv6 addresses, the same information can be specified as a *slen* value attached to the *saddr* parameter.

dmask

Analogous to *smask*.

lport

The value that follows is the local port of the datagram. This can be either a port number or a string searched with a NULL proto argument, as described in getservbyname(3XNET)

dport

The value that follows is the destination port of the datagram. This can be either a port number or a string searched with a NULL proto argument, as described in getservbyname(3XNET)

sport

The value that follows is the source port of the datagram. This can be either a port number or a string searched with a `NULL` proto argument, as described in `getservbyname`(3XNET)

dport

The value that follows is the destination port of the datagram. This can be either a port number or a string as described in `getservbyname`(3XNET) searched with `NULL` proto argument.

proto
ulp

The value that follows is the Upper Layer Protocol that this entry should be matched against. It could be a number or a string as described in `getprotobyname`(3XNET)If no smask or plen is specified, a plen of 32 for IPv4 or 128 for IPv6 will be used.

If no smask or plen is specified, a plen of 32 for IPv4 or 128 for IPv6 will be used, meaning a host.

Policy entries may contain the following (name value) pairs in the properties field. Each (name value) pair may appear only once in a given policy entry.

`auth_algs` An acceptable value following this implies that IPsec AH header will be present in the outbound datagram. Values following this describe the authentication algorithms that will be used while applying the IPsec AH on outbound datagrams and verified to be present on inbound datagrams. See *RFC 2402*.

This entry can contain either a string or a decimal number.

`string` This should be either MD5 or HMAC-MD5 denoting the HMAC-MD5 algorithm as described in *RFC 2403*, and SHA1, or HMAC-SHA1 or SHA or HMAC-SHA denoting the HMAC-SHA algorithm described in *RFC 2404*. The string can also be `ANY`, which denotes no-preference for the algorithm. Default algorithms will be chosen based upon the SAs available at this time for manual SAs and the key negotiating daemon for automatic SAs. Strings are not case-sensitive.

`number` A number in the range 1-255. This is useful when new algorithms can be dynamically loaded.

If *auth_algs* is not present, the AH header will not be present in the outbound datagram, and the same will be verified for the inbound datagram.

`encr_algs` An acceptable value following this implies that IPsec ESP header will be present in the outbound datagram. The value following this describes the encryption algorithms that will be used to apply the IPsec ESP protocol to outbound datagrams and verify it to be present on inbound datagrams. See *RFC 2406*.

This entry can contain either a string or a decimal number. Strings are not case-sensitive.

`string` This should be either `MD5` or `HMAC-MD5` denoting the `HMAC-MD5` algorithm as described in RFC 2403, and SHA1, or `HMAC-SHA1` or `SHA` or `HMAC-SHA` denoting the `HMAC-SHA` algorithm described in *RFC 2404*. The string can also be `ANY`, which denotes no-preference for the algorithm. Default

algorithms will be chosen based upon the SAs available at this time for manual SAs and the key negotiating daemon for automatic SAs. Strings are not case-sensitive.

`number` A decimal number in the range 1-255. This is useful when new algorithms can be dynamically loaded.

`encr_auth_algs` An acceptable value following `encr_auth_algs` implies that the IPsec ESP header will be present in the outbound datagram. The values following `encr_auth_algs` describe the authentication algorithms that will be used while applying the IPsec ESP protocol on outbound datagrams and verified to be present on inbound datagrams. See *RFC 2406*. This entry can contain either a string or a number. Strings are case-insensitive.

`string` Valid values are the same as the ones described for `auth_algs` above.

`number` This should be a decimal number in the range 1-255. This is useful when new algorithms can be dynamically loaded.

If `encr_algs` is present and `encr_auth_algs` is not present in a policy entry, the system will use an ESP SA regardless of whether the SA has an authentication algorithm or not.

If `encr_algs` is not present and `encr_auth_algs` is present in a policy entry, null encryption will be provided, which is equivalent to `encr_algs` with `NULL`, for outbound and inbound datagrams.

If both `encr_algs` and `encr_auth_algs` are not present in a policy entry, ESP header will not be present for outbound datagrams and the same will be verified for inbound datagrams.

If both `encr_algs` and `encr_auth_algs` are present in a policy entry, ESP header with integrity checksum will be present on outbound datagrams and the same will be verified for inbound datagrams.

For encr_algs, encr_auth_algs, and auth_algs a key length specification may be present. This is either a single value specifying the only valid key length for the algorithm or a range specifying the valid minimum and/or maximum key lengths. Minimum or maximum lengths may be omitted.

`dir` Values following this decides whether this entry is for outbound or inbound datagram. Valid values are strings that should be one of the following:

`out`	This means that this policy entry should be considered only for outbound datagrams.
`in`	This means that this policy entry should be considered only for inbound datagrams.
both	This means that this policy entry should be considered for both inbound and outbound datagrams

This entry is not needed when the action is "apply", "permit" or "ipsec". But if it is given while the action is "apply" or "permit", it should be "out" or "in" respectively. This is mandatory when the action is "bypass".

`sa` Values following this decide the attribute of the security association. Value indicates whether a unique security association should be used or any existing SA can be used. If there is a policy requirement, SAs are created dynamically on the first outbound datagram using the key management daemon. Static SAs can be created using `ipseckey`(1M). The values used here determine whether a new SA will be used/obtained. Valid values are strings that could be one of the following:

`unique`	Unique Association. A new/unused association will be obtained/used for packets matching this policy entry. If an SA that was previously used by the same 5 tuples, that is, {Source address, Destination address, Source port, Destination Port, Protocol (for example, TCP/UDP)} exists, it will be reused. Thus uniqueness is expressed by the 5 tuples given above. The security association used by the above 5 tuples will not be used by any other socket. For inbound datagrams, uniqueness will not be verified.
`shared`	Shared association. If an SA exists already for this source-destination pair, it will be used. Otherwise a new SA will be obtained. This is the default.

This is mandatory only for outbound policy entries and should not be given for entries whose action is "bypass". If this entry is not given for inbound entries, for example, when "dir" is in or "action" is permit, it will be assumed to be shared.

Action follows the pattern and should be given before properties. It should be one of the following and this field is mandatory.

ipsec	Use IPsec for the datagram as described by the properties, if the pattern matches the datagram. If ipsec is given without a dir spec , the pattern is matched to incoming and outgoing datagrams.
`apply`	Apply IPsec to the datagram as described by the properties, if the pattern matches the datagram. If `apply` is given, the pattern is matched only on the outbound datagram.
`permit`	Permit the datagram if the pattern matches the incoming datagram and satisfies the constraints described by the properties. If it does not satisfy the properties, discard the datagram. If `permit` is given, the pattern is matched only for inbound datagrams.
`bypass` `pass`	Bypass any policy checks if the pattern matches the datagram. `dir` in the properties decides whether the check is done on outbound or inbound datagrams. All the `bypass` entries are checked before checking with any other policy entry in the system. This has the highest precedence over any other entries. `dir` is the only field that should be present when action is `bypass`.
drop	Drop any packets that match the pattern.

If the file contains multiple policy entries, for example, they are assumed to be listed in the order in which they are to be applied. In cases of multiple entries matching the outbound and inbound datagram, the first match will be taken. The system will reorder the policy entry, that is, add the new entry before the old entry, only when:

- The level of protection is "stronger" than the old level of protection. Currently, strength is defined as:

  `AH and ESP > ESP > AH`
  The standard uses of AH and ESP were what drove this ranking of “stronger”. There are flaws with this. ESP can be used either without authentication, which will allow cut-and-paste or replay attacks, or without encryption, which makes it equivalent or slightly weaker than AH. An administrator should take care to use ESP properly. See `ipsecesp`(7P) for more details.

If the new entry has `bypass` as action, `bypass` has the highest precedence. It can be added in any order, and the system will still match all the `bypass` entries before matching any other entries. This is useful for key management daemons which can use this feature to bypass IPsec as it protects its own traffic.

Entries with both AH (`auth_algs` present in the policy entry) and ESP (`encr_auth_algs` or `encr_auth_algs` present in the policy entry) protection are ordered after all the entries with AH and ESP and before any AH–only and ESP–only entries. In all other cases the order specified by the user is not modified, that is, newer entries are added at the end of all the old entries. See EXAMPLES.

A new entry is considered duplicate of the old entry if an old entry matches the same traffic pattern as the new entry. See EXAMPLES for information on duplicates.

SECURITY

If, for example, the policy file comes over the wire from an NFS mounted file system, an adversary can modify the data contained in the file, thus changing the policy configured on the machine to suit his needs. Administrators should be cautious about transmitting a copy of the policy file over a network.

Policy is latched for TCP/UDP sockets on which a `connect`(3SOCKET) or `accept`(3SOCKET) has been issued. Adding new policy entries will not have any effect on them. This feature of latching may change in the future. It is not advisable to depend upon this feature.

Make sure to set up the policies before starting any communications, as existing connections may be affected by the addition of new policy entries. Similarly, do not change policies in the middle of a communication.

Note that certain `ndd` tunables affect how policies configured with this tool are enforced; see `ipsecesp`(7P) for more details.

If your source address is a host that can be looked up over the network, and your naming system itself is compromised, then any names used will no longer be trustworthy.

EXAMPLES

EXAMPLE 1 Protecting Outbound TCP Traffic With ESP and the AES Algorithm

The following example specified that any TCP packet from spiderweb to arachnid should be encrypted with AES, and the SA could be a shared one. It does not verify whether or not the inbound traffic is encrypted.

```
#
# Protect the outbound TCP traffic between hosts spiderweb
# and arachnid with ESP and use AES algorithm.
#
{
     laddr spiderweb
     raddr arachnid
     ulp tcp
     dir out
} ipsec {
         encr_algs AES
}
```

EXAMPLE 2 Verifying Whether or Not Inbound Traffic is Encrypted

Example 1 does not verify whether or not the inbound traffic is encrypted. The entry in this example protects inbound traffic:

```
#
# Protect the TCP traffic on inbound with ESP/DES from arachnid
# to spiderweb
#
{
        laddr spiderweb
        raddr arachnid
```

EXAMPLE 2 Verifying Whether or Not Inbound Traffic is Encrypted *(Continued)*

```
        ulp tcp
        dir in
} ipsec {
        encr_algs AES
}
```

`sa` can be absent for inbound policy entries as it implies that it can be a shared one. Uniqueness is not verified on inbound. Note that in both the above entries, authentication was never specified. This can lead to cut and paste attacks. As mentioned previously, though the authentication is not specified, the system will still use an ESP SA with `encr_auth_alg` specified, if it was found in the SA tables.

EXAMPLE 3 Protecting All Traffic Between Two Hosts

The following example protects both directions at once:

```
{
        laddr spiderweb
        raddr arachnid
        ulp tcp
} ipsec {
        encr_algs AES
}
```

EXAMPLE 4 Authenticating All Inbound Traffic to the Telnet Port

This entry specifies that any inbound datagram to telnet port should come in authenticated with the SHA1 algorithm. Otherwise the datagram should not be permitted. Without this entry, traffic destined to port number 23 can come in clear. `sa` is not specified, which implies that it is shared. This can be done only for inbound entries. You need to have an equivalent entry to protect outbound traffic so that the outbound traffic is authenticated as well, remove the dir.

```
#
# All the inbound traffic to the telnet port should be
# authenticated.
#
{
       lport telnet
       dir in
} ipsec {
       auth_algs sha1
}
```

EXAMPLE 5 Verifying Inbound Traffic is Null-Encrypted

The first entry specifies that any packet with address host-B should not be checked against any policies. The second entry specifies that all inbound traffic from network-B should be encrypted with a `NULL` encryption algorithm and the MD5 authentication

EXAMPLE 5 Verifying Inbound Traffic is Null-Encrypted *(Continued)*

algorithm. `NULL` encryption implies that ESP header will be used without encrypting the datagram. As the first entry is `bypass` it need not be given first in order, as `bypass` entries have the highest precedence. Thus any inbound traffic will be matched against all `bypass` entries before any other policy entries.

```
#
# Make sure that all inbound traffic from network-B is NULL
# encrypted, but bypass for host-B alone from that network.
# Add the bypass first.
{
raddr host-B
    dir in
} bypass {}

# Now add for network-B.
{
    raddr network-B/16
    dir in
} ipsec {
encr_algs NULL
encr_auth_algs md5
}
```

EXAMPLE 6 Entries to Bypass Traffic from IPsec

The first two entries provide that any datagram leaving the machine with source port 53 or coming into port number 53 should not be subjected to IPsec policy checks, irrespective of any other policy entry in the system. Thus the latter two entries will be considered only for ports other than port number 53.

```
#
# Bypass traffic for port no 53
     #
{lport 53} bypass {}
{rport 53} bypass {}
{raddr spiderweb } ipsec {encr_algs any sa unique}
```

EXAMPLE 7 Protecting Outbound Traffic

```
 #
     # Protect the outbound traffic from all interfaces.
     #
{raddr spiderweb dir out} ipsec {auth_algs any sa unique}
```

If the `gethostbyname`(3XNET) call for spiderweb yields multiple addresses, multiple policy entries will be added for all the source address with the same properties.

```
{
    laddr arachnid
    raddr spiderweb
    dir in
```

EXAMPLE 7 Protecting Outbound Traffic *(Continued)*

```
} ipsec {auth_algs any sa unique}
```

If the `gethostbyname`(3XNET) call for spiderweb and the `gethostbyname`(3XNET) call for arachnid yield multiple addresses, multiple policy entries will be added for each (`saddr daddr`) pair with the same properties. Use `ipsecconf -l` to view all the policy entries added.

EXAMPLE 8 Bypassing Unauthenticated Traffic

```
#
# Protect all the outbound traffic with ESP except any traffic
# to network-b which should be authenticated and bypass anything
# to network-c
#
{raddr network-b/16 dir out} ipsec {auth_algs any}
{dir out} ipsec {encr_algs any}
{raddr network-c/16 dir out} bypass {} # NULL properties
```

Note that `bypass` can be given anywhere and it will take precedence over all other entries. `NULL` pattern matches all the traffic.

EXAMPLE 9 Encrypting IPv6 Traffic with 3DES and MD5

The following entry on the host with the link local address `fe80::a00:20ff:fe21:4483` specifies that any outbound traffic between the hosts wtih IPv6 link-local addresses `fe80::a00:20ff:fe21:4483` and `fe80::a00:20ff:felf:e346` must be encrypted with 3DES and MD5.

```
{
    laddr fe80::a00:20ff:fe21:4483
    raddr fe80::a00:20ff:felf:e346
    dir out
} ipsec {
    encr_algs 3DES
    encr_auth_algs MD5
}
```

EXAMPLE 10 Verifying IPv6 Traffic is Authenticated with SHA1

The following two entries require that all IPv6 traffic to and from the IPv6 site-local network `fec0:abcd::0/32` be authenticated with SHA1.

```
{raddr fec0:abcd::0/32} ipsec { auth_algs SHA1 }
```

EXAMPLE 11 Key Lengths

```
# use aes at any key length
{raddr spiderweb} ipsec {encr_algs aes}
```

EXAMPLE 11 Key Lengths *(Continued)*

```
# use aes with a 192 bit key
{raddr spiderweb} ipsec {encr_algs aes(192)}

# use aes with any key length up to 192 bits
# i.e. 192 bits or less
{raddr spiderweb} ipsec {encr_algs aes(..192)}

# use aes with any key length of 192 or more
# i.e. 192 bits or more
{raddr spiderweb} ipsec {encr_algs aes(192..)}

#use aes with any key from 192 to 256 bits
{raddr spiderweb} ipsec {encr_algs aes(192..256)}

#use any algorithm with a key of 192 bits or longer
{raddr spiderweb} ipsec {encr_algs any(192..)}
```

EXAMPLE 12 Using "or"

The following entry allows traffic using the AES or Blowfish algorithms from the remote machine spiderweb:

```
{raddr spiderweb} ipsec {encr_algs aes} or {encr_algs blowfish}
```

FILES

`/var/run/ipsecpolicy.conf`	Cache of IPsec policies currently configured for the system, maintained by `ipsecconf` command. Do not edit this file.
`/etc/inet/ipsecinit.conf`	File containing IPsec policies to be installed at the time the system transitions from run-level 2 or 3. If present, these policies are loaded after `/usr` is mounted but before any non-boot-time routing information is processed and before any Internet services are started, including naming services.
`/etc/inet/ipsecinit.sample`	Sample input file for `ipseconf`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu
Interface Stability	Evolving

SEE ALSO

`in.iked`(1M), `init`(1M), `ifconfig`(1M), `ipseckey`(1M), `accept`(3SOCKET), `connect`(3SOCKET), `gethostbyname`(3XNET), `getnetbyname`(3XNET), `getprotobyname`(3XNET), `getservbyname`(3XNET), `getaddrinfo`(3SOCKET), `socket`(3SOCKET), `attributes`(5), `ipsecah`(7P) , `ipsecesp`(7P), `tun`(7M)

Glenn, R. and Kent, S. *RFC 2410, The NULL Encryption Algorithm and Its Use With IPsec*. The Internet Society. 1998.

Kent, S. and Atkinson, R. *RFC 2402, IP Authentication Header*.The Internet Society. 1998.

Kent, S. and Atkinson, R. *RFC 2406, IP Encapsulating Security Payload (ESP)*. The Internet Society. 1998.

Madsen, C. and Glenn, R. *RFC 2403, The Use of HMAC-MD5-96 within ESP and AH*. The Internet Society. 1998.

Madsen, C. and Glenn, R. *RFC 2404, The Use of HMAC-SHA-1-96 within ESP and AH*. The Internet Society. 1998.

Madsen, C. and Doraswamy, N. *RFC 2405, The ESP DES-CBC Cipher Algorithm With Explicit IV*. The Internet Society. 1998.

Pereira, R. and Adams, R. *RFC 2451, The ESP CBC-Mode Cipher Algorithms*. The Internet Society. 1998.

Frankel, S. and Kelly, R. Glenn, *The AES Cipher Algorithm and Its Use With IPsec*, 2001.

DIAGNOSTICS

Bad "string" on line *N*.
Duplicate "string" on line *N*.
: *string* refers to one of the names in pattern or properties. A Bad string indicates that an argument is malformed; a Duplicate string indicates that there are multiple arguments of a similar type, for example, multiple Source Address arguments..

Error before or at line *N*.
: Indicates parsing error before or at line *N*.

Non-existent index
: Reported when the *index* for delete is not a valid one.

spd_msg return: File exists
: Reported when there is already a policy entry that matches the traffic of this new entry.

NAME

ipseckey – manually manipulate an IPsec Security Association Database (SADB)

SYNOPSIS

ipseckey [-nvp]

ipseckey [-nvp] -f *filename*

ipseckey [-nvp] [delete | get]SA_TYPE {EXTENSION *value*...}

ipseckey [-np] [monitor | passive_monitor | pmonitor]

ipseckey [-nvp] flush {SA_TYPE}

ipseckey [-nvp] dump {SA_TYPE}

ipseckey [-nvp] save SA_TYPE {*filename*}

ipseckey [-nvp] -s *filename*

DESCRIPTION

The ipseckey command is used to manually manipulate the security association databases of the network security services, ipsecah(7P) and ipsecesp(7P). You can use the ipseckey command to set up security associations between communicating parties when automated key management is not available.

While the ipseckey utility has only a limited number of general options, it supports a rich command language. The user may specify requests to be delivered by means of a programmatic interface specific for manual keying. See pf_key(7P). When ipseckey is invoked with no arguments, it will enter an interactive mode which prints a prompt to the standard output and accepts commands from the standard input until the end-of-file is reached. Some commands require an explicit security association ("SA") type, while others permit the SA type to be unspecified and act on all SA types.

ipseckey uses a PF_KEY socket and the message types SADB_ADD, SADB_DELETE, SADB_GET, SADB_UPDATE, SADB_FLUSH, and SADB_X_PROMISC. Thus, you must be a superuser to use this command.

ipseckey handles sensitive cryptographic keying information. Please read the SECURITY CONSIDERATIONS section for details on how to use this command securely.

OPTIONS

-f [*filename*] Read commands from an input file, *filename*. The lines of the input file are identical to the command line language. The load command provides similar functionality. The -s option or the save command can generate files readable by the -f argument.

-n Prevent attempts to print host and network names symbolically when reporting actions. This is useful, for example, when all name servers are down or are otherwise unreachable.

-p Paranoid. Do not print any keying material, even if saving SAs. Instead of an actual hexadecimal digit, print an X when this flag is turned on.

`-s` [*filename*] — The opposite of the `-f` option. If '`-`' is given for a *filename*, then the output goes to the standard output. A snapshot of all current `SA` tables will be output in a form readable by the `-f` option. The output will be a series of `add` commands, but with some names not used. This occurs because a single name may often indicate multiple addresses.

`-v` — Verbose. Print the messages being sent into the `PF_KEY` socket, and print raw seconds values for lifetimes.

COMMANDS

`add` — Add an SA. Because it involves the transfer of keying material, it cannot be invoked from the shell, lest the keys be visible in `ps`(1) output. It can be used either from the interactive `ipseckey>` prompt or in a command file specified by the `-f` command. The `add` command accepts all extension-value pairs described below.

`update` — Update SA lifetime, and in the cases of larval SAs (leftover from aborted automated key management), keying material and other extensions. Like `add`, this command cannot be invoked from the shell because keying material would be seen by the `ps`(1) command. It can be used either from the interactive `ipseckey>` prompt or in a command file specified by the `-f` command. The `update` command accepts all extension-value pairs, but normally is only used for SA lifetime updates.

`delete` — Delete a specific SA from a specific `SADB`. This command requires the `spi` extension, and the `dest` extension for IPsec SAs. Other extension-value pairs are superfluous for a delete message.

`get` — Lookup and display a security association from a specific `SADB`. Like delete, this command only requires `spi` and `dest` for IPsec.

`flush` — Remove all SA for a given `SA_TYPE`, or all SA for all types.

`monitor` — Continuously report on any `PF_KEY` messages. This uses the `SADB_X_PROMISC` message to enable messages that a normal `PF_KEY` socket would not receive to be received. See `pf_key`(7P).

`passive_monitor` — Like monitor, except that it does not use the `SADB_X_PROMISC` message.

`pmonitor` — Synonym for `passive_monitor`.

`dump`	Will display all SAs for a given SA type, or will display all SAs. Because of the large amount of data generated by this command, there is no guarantee that all SA information will be successfully delivered, or that this command will even complete.
`save`	Is the command analog of the `-s` option. It is included as a command to provide a way to snapshot a particular SA type, for example, `esp` or `ah`.
`help`	Prints a brief summary of commands.

SECURITY ASSOCIATION TYPES

`all`	Specifies all known SA types. This type is only used for the `flush` and `dump` commands. This is equivalent to having no SA type for these commands.
`ah`	Specifies the IPsec Authentication Header ("`AH`") SA.
`esp`	Specifies the IPsec Encapsulating Security Payload ("`ESP`") SA.

EXTENSION VALUE TYPES

Commands like `add`, `delete`, `get`, and `update` require that certain extensions and associated values be specified. The extensions will be listed here, followed by the commands that use them, and the commands that require them. Requirements are currently documented based upon the IPsec definitions of an SA. Required extensions may change in the future. `<number>` can be in either hex (`0xnnn`), decimal (`nnn`) or octal (`0nnn`). `<string>` is a text string. `<hexstr>` is a long hexadecimal number with a bit-length. Extensions are usually paired with values; however, some extensions require two values after them.

`spi <number>`
Specifies the security parameters index of the `SA`. This extension is required for the `add`, `delete`, `get` and `update` commands.

`replay <number>`
Specifies the replay window size. If not specified, the replay window size is assumed to be zero. It is not recommended that manually added SAs have a replay window. This extension is used by the `add` and `update` commands.

`state <string>|<number>`
Specifies the SA state, either by numeric value or by the strings "`larval`", "`mature`", "`dying`" or "`dead`". If not specified, the value defaults to `mature`. This extension is used by the `add` and `update` commands.

`auth_alg <string>|<number>`
`authalg <string>|<number>`
Specifies the authentication algorithm for an `SA`, either by numeric value, or by strings indicating an algorithm name. Current authentication algorithms include:

`HMAC-MD5`	`md5`, `hmac-md5`
`HMAC-SH-1`	`sha`, `sha-1`, `hmac-sha1`, `hmac-sha`

Often, algorithm names will have several synonyms. This extension is required by the add command for certain SA types. It is also used by the update command.

encr_alg <string>|<number>
encralg <string>|<number>
Specifies the encryption algorithm for an SA, either by numeric value, or by strings indicating an algorithm name. Current encryption algorithms include DES ("des"), Triple-DES ("3des"), Blowfish ("blowfish"), and AES ("aes"). This extension is required by the add command for certain SA types. It is also used by the update command.

The next six extensions are lifetime extensions. There are two varieties, "hard" and "soft". If a hard lifetime expires, the SA will be deleted automatically by the system. If a soft lifetime expires, an SADB_EXPIRE message will be transmitted by the system, and its state will be downgraded to dying from mature. See pf_key(7P). The monitor command to key allows you to view SADB_EXPIRE messages.

soft_bytes <number>
hard_bytes <number>
Specifies the number of bytes that this SA can protect. If this extension is not present, the default value is zero, which means that the SA will not expire based on the number of bytes protected. This extension is used by the add and update commands.

soft_addtime <number>
hard_addtime <number>
Specifies the number of seconds that this SA can exist after being added or updated from a larval SA. An update of a mature SA does not reset the initial time that it was added. If this extension is not present, the default value is zero, which means the SA will not expire based on how long it has been since it was added. This extension is used by the add and update commands.

soft_usetime <number>
hard_usetime <number>
Specifies the number of seconds this SA can exist after first being used. If this extension is not present, the default value is zero, which means the SA will not expire based on how long it has been since it was added. This extension is used by the add and update commands.

saddr <address|name>
srcaddr <address|name>
saddr6 <IPv6 address>
srcaddr6 <IPv6 address>
src <address|name>
src6 <IPv6 address>
srcaddr <address> and src <address> are synonyms that indicate the source address of the SA. If unspecified, the source address will either remain unset, or it will be set to a wildcard address if a destination address was supplied. To not

specify the source address is valid for IPsec SAs. Future SA types may alter this assumption. This extension is used by the `add`, `update`, `get` and `delete` commands.

`daddr <address|name>`
`dstaddr <address|name>`
`daddr6 <IPv6 address|name>`
`dstaddr6 <IPv6 address|name>`
`dst <addr|name>`
`dst6 <IPv6 address|name>`
`dstaddr <addr>` and `dst <addr>` are synonyms that indicate the destination address of the `SA`. If unspecified, the destination address will remain unset. Because IPsec SAs require a specified destination address and `spi` for identification, this extension, with a specific value, is required for the `add`, `update`, `get` and `delete` commands.

If a name is given, `ipseckey` will attempt to invoke the command on multiple SAs with all of the destination addresses that the name can identify. This is similar to how `ipsecconf` handles addresses.

If `dst6` or `dstaddr6` is specified, only the IPv6 addresses identified by a name are used.

`proxyaddr <address|name>`
`proxy <address|name>`
`proxyaddr <address>` and `proxy <address>` are synonyms that indicate the proxy address for the SA. A proxy address is used for an SA that is protecting an inner protocol header. The proxy address is the source address of the inner protocol's header. This extension is used by the `add` and `update` commands.

`authkey <hexstring>`
Specifies the authentication key for this SA. The key is expressed as a string of hexadecimal digits, with an optional `/` at the end, for example, `123/12`. Bits are counted from the most-significant bits down. For example, to express three '1' bits, the proper syntax is the string `"e/3"`. For multi-key algorithms, the string is the concatenation of the multiple keys. This extension is used by the `add` and `update` commands.

`encrkey <hexstring>`
Specifies the encryption key for this SA. The syntax of the key is the same as `authkey`. A concrete example of a multi-key encryption algorithm is `3des`, which would express itself as a 192-bit key, which is three 64-bit parity-included `DES` keys. This extension is used by the `add` and `update` commands.

Keying material is very sensitive and should be generated as randomly as possible. Some algorithms have known weak keys. IPsec algorithms have built-in weak key checks, so that if a weak key is in a newly added `SA`, the `add` command will fail with an invalid value.

Certificate identities are very useful in the context of automated key management, as they tie the SA to the public key certificates used in most automated key management protocols. They are less useful for manually added SAs. Unlike other extensions, `srcidtype` takes two values, a `type`, and an actual `value`. The type can be one of the following:

`prefix`	An address prefix.
`fqdn`	A fully-qualified domain name.
`domain`	Domain name, synonym for `fqdn`.
`user_fqdn`	User identity of the form `user@fqdn`.
`mailbox`	Synonym for `user_fqdn`.

The value is an arbitrary text string, which should identify the certificate.

`srcidtype <type, value>`
Specifies a source certificate identity for this SA. This extension is used by the `add` and `update` commands.

`dstidtype <type, value>`
Specifies a destination certificate identity for this SA. This extension is used by the `add` and `update` commands

SECURITY CONSIDERATIONS

The `ipseckey` command allows a privileged user to enter cryptographic keying information. If an adversary gains access to such information, the security of IPsec traffic is compromised. The following issues should be taken into account when using the `ipseckey` command.

1. Is the TTY going over a network (interactive mode)?
   - If it is, then the security of the keying material is the security of the network path for this TTY's traffic. Using `ipseckey` over a clear-text telnet or rlogin session is risky.
   - Even local windows may be vulnerable to attacks where a concealed program that reads window events is present.
2. Is the file accessed over the network or readable to the world (`-f` option)?
   - A network-mounted file can be sniffed by an adversary as it is being read. A world-readable file with keying material in it is also risky.

If your source address is a host that can be looked up over the network, and your naming system itself is compromised, then any names used will no longer be trustworthy.

Security weaknesses often lie in misapplication of tools, not the tools themselves. Administrators are urged to be cautious when using `ipseckey`. The safest mode of operation is probably on a console, or other hard-connected TTY.

For further thoughts on this subject, see the afterward by Matt Blaze in Bruce Schneier's *Applied Cryptography: Protocols, Algorithms, and Source Code in C*.

EXAMPLES

EXAMPLE 1 Emptying Out All SAs

To empty out all SA:

```
example# ipseckey flush
```

EXAMPLE 2 Flushing Out IPsec AH SAs Only

To flush out only IPsec `AH` SAs:

```
example# ipseckey flush ah
```

EXAMPLE 3 Saving All SAs To Standard Output

To save all SAs to the standard output:

```
example# ipseckey save all
```

EXAMPLE 4 Saving `ESP` SAs To The File `/tmp/snapshot`

To save `ESP` SAs to the file `/tmp/snapshot`:

```
example# ipseckey save esp /tmp/snapshot
```

EXAMPLE 5 Deleting an IPsec SA

To delete an IPsec SA, only the `SPI` and the destination address are needed:

```
example# ipseckey delete esp spi 0x2112 dst 224.0.0.1
```

EXAMPLE 6 Getting Information on an IPsec SA

Likewise, getting information on a SA only requires the destination address and `SPI`:

```
example# ipseckey get ah spi 0x5150 dst mypeer
```

EXAMPLE 7 Adding or Updating IPsec SAs

Adding or updating SAs requires entering interactive mode:

```
example# ipseckey
ipseckey> add ah spi 0x90125 src me.domain.com dst you.domain.com  \
          authalg md5 authkey 1234567890abcdef1234567890abcdef
ipseckey> update ah spi 0x90125 dst you.domain.com hard_bytes \
          16000000 ipseckey> exit
```

EXAMPLE 8 Adding an SA in the Opposite Direction

In the case of IPsec, SAs are unidirectional. To communicate securely, a second SA needs to be added in the opposite direction. The peer machine also needs to add both SAs.

```
example#  ipseckey
ipseckey> add ah spi 0x2112 src you.domain.com dst me.domain.com \
          authalg md5 authkey bde359723576fdea08e56cbe876e24ad \
```

EXAMPLE 8 Adding an SA in the Opposite Direction *(Continued)*

```
          hard_bytes 16000000
ipseckey> exit
```

EXAMPLE 9 Monitoring PF_KEY Messages

Monitoring for PF_KEY messages is straightforward:

```
example#  ipseckey monitor
```

EXAMPLE 10 Using Commands in a File

Commands can be placed in a file that can be parsed with the -f option. This file may contain comment lines that begin with the "#" symbol. For example:

```
# This is a sample file for flushing out the ESP table and
# adding a pair of SAs.

flush esp

### Watch out!  I have keying material in this file.  See the
### SECURITY CONSIDERATIONS section in this manual page for why this can be
### dangerous .

add esp spi 0x2112 src me.domain.com dst you.domain.com \
    authalg md5 authkey bde359723576fdea08e56cbe876e24ad \
    encralg des encrkey be02938e7def2839 hard_usetime 28800
add esp spi 0x5150 src you.domain.com dst me.domain.com \
    authalg md5 authkey 930987dbe09743ade09d92b4097d9e93 \
    encralg des encrkey 8bd4a52e10127deb hard_usetime 28800

## End of file  -  This is a gratuitous comment
```

EXAMPLE 11 Adding SAs for IPv6 Addresses

The following commands from the interactive-mode create an SA to protect IPv6 traffic between the site-local addresses

```
example # ipseckey
ipseckey> add esp spi 0x6789 src6 fec0:bbbb::4483 dst6 fec0:bbbb::7843
          authalg md5 authkey bde359723576fdea08e56cbe876e24ad
          encralg des encrkey be02938e7def2839 hard_usetime 28800
ipseckey>exit
```

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Evolving

SEE ALSO

`ps`(1), `ipsecconf`(1M), `route`(1M), `attributes`(5), `ipsec`(7P), `ipsecah`(7P), `ipsecesp`(7P), `pf_key`(7P)

Schneier, B., *Applied Cryptography: Protocols, Algorithms, and Source Code in C*. Second ed. New York, New York: John Wiley & Sons, 1996.

DIAGNOSTICS

Parse error on line N.
If an interactive use of `ipseckey` would print usage information, this would print instead. Usually proceeded by another diagnostic.

Unexpected end of command line.
An additional argument was expected on the command line.

Unknown
A value for a specific extension was unknown.

Address type N not supported.
A name-to-address lookup returned an unsupported address family.

is not a bit specifier
bit length N is too big for
string is not a hex string
Keying material was not entered appropriately.

Can only specify single
A duplicate extension was entered.

Don't use extension for <string> for <command>.
An extension not used by a command was used.

One of the entered values is incorrect: Diagnostic code *NN*: <msg>
This is a general invalid parameter error. The diagnostic code and message provides more detail about what precise value was incorrect and why.

NOTES

In spite of its IPsec-specific name, `ipseckey` is analogous to `route`(1M), in that it is a command-line interface to a socket-based administration engine, in this case, `PF_KEY`. `PF_KEY` was originally developed at the United States Naval Research Laboratory.

To have machines communicate securely with manual keying, SAs need to be added by all communicating parties. If two nodes wish to communicate securely, both nodes need the appropriate SAs added.

In the future `ipseckey` may be invoked under additional names as other security protocols become available to `PF_KEY`.

NAME

kadb – a kernel debugger

SPARC

ok boot *device_specifier* kadb [-d] [*boot-flags*]

> b kadb [-d] [*boot-flags*]

IA

select (b)oot or (i)nterpreter: b kadb [-d] [*boot-flags*]

select (b)oot or (i)nterpreter: i kadb [-d] [*boot-flags*]

DESCRIPTION

kadb is an interactive kernel debugger with a user interface similar to adb(1), the assembly language debugger.

kadb must be loaded prior to the standalone program it is to debug. kadb runs with the same environment as the standalone program, so it shares many resources with that program. The debugger is cognizant of and able to control multiple processors, should they be present in a system.

When kadb is started, it requests the default filename from boot(1M), and if loaded non-interactively (without the -d option), it loads the default filename.

On systems which support both 32-bit and 64-bit operating system, boot(1M) chooses an appropriate default filename for that system. If loaded interactively (by specifying the -d option), kadb prompts with the default filename, which can be changed before continuing. The default filename can be specified on the boot(1M) command line. See boot(1M) for details.

Before loading the 64-bit kernel explicitly, review the information in boot(1M) for restrictions on running the 64-bit kernel on certain configurations.

Unlike adb(1), kadb runs in the same supervisor virtual address space as the program being debugged, although it maintains a separate context. The debugger runs as a *co-process* that cannot be killed (no :k command as in adb) or rerun (no :r command as in adb) . There is no signal control (no :i, :t, or $i commands as in adb), although the keyboard facilities (CTRL–C, CTRL–S, and CTRL–Q) are simulated.

In the case of the UNIX system kernel, the keyboard abort sequence suspends kernel operations and breaks into the debugger. This behavior may be disabled by way of the kbd(1) command and may not be the current default on all systems. See kb(7M) for more information.

As the kernel is composed of the core image file and the set of loadable modules already in memory, kadb has the capability of debugging all of these by traversing special data structures. kadb makes use of this feature to allow it to reference any kernel data structure, even if it resides in a loadable module. kadb sets the -d flag by default so the program being debugged can tell it is being watched. If this flag is not given, kadb loads and immediately runs the default kernel .

Most adb(1) commands function in kadb as they do in adb. As with adb -k, $p works when debugging kernels. The verbs ? and / are equivalent in kadb, as there is only one address space in use.

The keyboard abort sequence is L1+A on keyboards with an L1 key, and F1+A on keyboards with no L1 key. On serial lines, the default abort sequence is a BREAK signal, but this can be changed to the sequence "carriage return, tilde, control‐B" using:

```
kbd -a alternate
```

See kbd(1).

Once aborted, kadb responds with the following:

```
kadb[cpu]:
```

where *cpu* is the number of the CPU on which kadb is currently executing.

OPTIONS

The following options are supported:

`-d` Interactive startup. If specified, kadb stops after loading and displays the `kadb:` prompt, followed by the name of the default program to debug.

The user may either press RETURN to debug the default program, or BACK SPACE followed by the name of another program to debug.

OPERANDS

The following operands are supported:

boot-flags Specifies boot flags as arguments to kadb. The specified boot-flags are passed to the program being debugged. See boot(1M) for available boot-flags.

SPARC Only

device-specifier Specifies the device from which to load. See monitor(1M).

Kernel Macros

As with adb(1), kernel macros can be used with kadb, but they cannot be read from a file at runtime. Use the `kadb $M` command to list all of the built-in kadb macros.

Commands

kadb reads commands from the standard input and displays responses on the standard output. kadb supports the majority of the adb(1) commands. kadb does not support the following adb commands: `:k`, `:r`, `:i`, `:t`, or `$i`. See adb(1).

Additionally, kadb supports the following commands:

`[`
Performs the same function as `:e` in adb(1), but requires only one keystroke and no RETURN (ENTER on IA based systems).

`]`
Performs the same function as `:s` in adb(1), but requires only one keystroke and no RETURN (ENTER on IA based systems).

`:a`
Sets a hardware access (read or write) breakpoint using the processor hardware facilities. The syntax and action for this command is the same as the `:b` command in `adb`, with the following exceptions:

- The breakpoint triggers if any bytes from the breakpoint for length bytes are being accessed. See `$l` below for setting the length of a hardware breakpoint.
- Breakpoints should be aligned for the length specified. Any address is valid for length `1`. Addresses divisible by `2` should be used for length `2` (short). Addresses divisible by `4` should be used for length `4` (int).
- Detection of an access breakpoint occurs after completion of the instruction that caused it.
- There are a limited number (`4`) of hardware breakpoint registers, and, when set, this uses one.
- As this breakpoint does not modify memory locations, this command will work on locations that are not in core at the time the breakpoint is set.

@fmt
Used in the same manner as the `adb /` and `?` commands. Specify @ as a physical memory address as opposed to the normal virtual address. Specify `fmt` as any of the formats used with the `adb /` command. This command is useful for displaying memory that may not be mapped, for example, `kernel` page tables or buffers used for DMA by device drivers.

`function:: call` *arg1, arg2, arg3, ...*
Invokes kernel functions with `0` or more arguments. Using this command results in a response such as:

retval = function(arg1,arg2,arg3, . . .);

where *retval* is the return value of the function. This feature can be error prone, as functions may have side effects that cause failures if the kernel is continued.

`:p`
Sets a hardware access (read or write) breakpoint using the processor hardware facilities when an instruction at the specified address is run. The `$l` operation has no effect on this type of breakpoint. This breakpoint occurs before the instruction is executed.

`:P`
Works as `:a`, but this command will only breakpoint when an access is made to the address in IA I/O space. See `:a`.

`:w`
Sets a write hardware access breakpoint using the processor hardware facilities.

[*length*]`$l`
Sets the default data length for an access or write breakpoint. *length* can be set to `1` for byte, `2` for short, and `4` for int word accesses. If *length* is not specified, `1` byte is assumed. Once set, this value affects any newly set access or write breakpoints, but does not affect ones set before this operation.

`$b`
Displays two additional columns that `adb` does not. The first is the `type` column which indicates `soft` for a normal breakpoint, `access` for an access hardware breakpoint, `write` for a write hardware breakpoint, and `inst` for an instruction hardware breakpoint. The second is the `len` column which for access and write breakpoints indicate the length of the operation to break on.

SPARC

`$q` Gives control to the boot prom, from which you may reboot the system.

cpu`:x` Switches the active CPU to *cpu*. Thereafter, commands such as `$r` and `$c` displays the registers and stack of the new CPU, *cpu*.

IA

port`:i` Inputs a byte for display from *port*. *port* is an address-specified I/O port. For example, `330:i` inputs from address port `330`.

port`:i8` Same as the `:i` command. See `:i`.

port`:i16` Inputs two bytes for display from *port*. *port* is an address-specified I/O port.

port`:i32` Inputs four bytes for display from *port*. *port* is an address-specified I/O port.

port,*data*`:o` Outputs a byte to *port*. *port* is an address-specified I/O port. [*address*], [*data*]`:o` outputs the value *data* to address I/O *port*. For example, `330,80:o` outputs `80` to address port `330`.

port,*data*`:o8` Same as the `:o` command. See *port*,*data*`:o`.

port,*data*`:o16` Outputs two bytes to *port*. *port* is an address-specified I/O port.

port,*data*`:o32` Outputs four bytes to *port*. *port* is an address-specified I/O port.

`$q` Prompts the user with:

`Type 'y' if you really want to reboot.` Responding with a `y` or `Y` causes the system to reboot. Responding with anything other than a `y` or `Y` returns control to `kadb`. Use this feature when you cannot press the `reset` switch on your machine. Because using `$q` may result in data loss, this command should only be used when you would press the `reset` switch or power off your system.

Online Help Commands

`::help` Displays the formats of `kadb` commands and extended commands.

`::?` Same as the `::help` command. See `::help`.

`::morehelp` Displays additional information about commonly used commands and provides an explanation of data formats.

Scroll Control Feature

num`::more` A common problem with using `kadb` is that scrolling is sometimes too fast and that CTRL-s and CTRL-q are inexact controls. A conditional scroll control feature similar to `more`(1) has been added to `kadb`. To enable this feature, the user specifies the

number of lines to be displayed, followed by `::more`. For example, the command `14::more` displays `14` (current radix) lines, followed by the `--More--` prompt. At this prompt, press: ENTER or RETURN to display one more line. Press `c`, `C`, or CTRL-c to interrupt the display. Press any other key to display the next *num* number of specified lines (`14` in this example). The command `::more` displays the current setting for the number of lines that `kadb` displays before printing the `--More--` prompt. The initial scroll control value of this feature is `0`, meaning that scrolling is disabled. Once enabled, the `0::more` command disables the scroll control feature.

Deferred Breakpoint Feature

Since the kernel is dynamically loaded, not all modules may be loaded when a breakpoint is set `kadb` can set *deferred* breakpoints which will be dynamically inserted when the corresponding module is loaded. The module and the location must both be specified when referring to a deferred breakpoint, as follows:

```
module_name#location:
```

This syntax is implemented for `kadb` only and uses existing breakpoint commands (for example, `ufs#ufs_open:b` or `ufs#ufs_open+4,5:b`).

If the module has been loaded, `kadb` attempts to find the symbol in the module specified. If `kadb` finds the symbol, it sets a regular breakpoint. If it does not find the symbol, it generates an error message and returns to the command line without setting a breakpoint.

If `kadb` fails to find the module on the list of currently loaded modules, it does not resolve the `location`. Instead, it sends a message to the user and sets a deferred breakpoint.

When the specified module is loaded, `kadb` tries to resolve the location. If the location can be resolved, the deferred breakpoint is converted to a regular breakpoint. If `kadb` cannot resolve the location, a message is sent to the user, and `kadb` halts execution. In this case, `kadb` does not convert the deferred breakpoint to a regular breakpoint; it removes it from the breakpoint table. The user may then re-enter a correct breakpoint. Strict scoping is enforced, so `kadb` does not look at any other module than the one specified with the location.

When specifying a deferred breakpoint in which the module name contains a hyphen, you must escape the hyphen. For example, for a module named `pci-ide`, enter:

```
pci\-ide#_init:b
```

The output from the the `$b` command indicates whether the breakpoint is of type "deferred" (`defr`) or is another type.

FILES

`/platform/`*`platform-name`*`/kadb`
primary debugger path

/platform/*hardware-class-name*/kadb
alternative debugger path for some platforms

/platform/*platform-name*/kernel/unix
primary default 32–bit kernel

/platform/*hardware-class-name*/kernel/unix
alternative default 32–bit kernel for some platforms

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcar

SEE ALSO

adb(1), more(1), uname(1), boot(1M), kernel(1M), attributes(5), kb(7M)

SPARC Only

kbd(1), monitor(1M), obpsym(1M)

DIAGNOSTICS

When there is no current command or format, kadb comments about syntax errors, abnormal termination of commands, and the like.

SPARC Only

On a SPARC based system, kadb cannot reliably single-step over instructions which change the processor status register.

SPARC and IA

If a breakpoint or watchpoint is triggered while the console frame buffer is powered off, the system can crash and be left in a state from which it is difficult to recover. If one of these is triggered while the monitor is powered off, you will not be able to see the debugger output.

NOTES

platform-name can be found using the -i option of uname(1). *hardware-class-name* can be found using the -m option of uname(1).

NAME

kadmin, kadmin.local – Kerberos database administration program

SYNOPSIS

```
/usr/sbin/kadmin [-r realm] [-p principal] [-q query] [-s admin_server
     [:port]] [ [-c credential_cache] | [-k [-t keytab]] | [-w password]]

/usr/sbin/kadmin.local [-r realm] [-p principal] [-q query] [-d dbname]
     [-e "enc:salt..."] [-m] [-D]
```

DESCRIPTION

`kadmin` and `kadmin.local` are interactive command-line interfaces to the Kerberos V5 administration system. They provide for the maintenance of Kerberos principals, policies, and service key tables (keytabs). `kadmin` and `kadmin.local` provide identical functionality; the difference is that `kadmin.local` can run only on the master KDC and does not use Kerberos authentication. Except as explicitly noted otherwise, this man page will use `kadmin` to refer to both versions.

By default, both versions of `kadmin` attempt to determine your user name and perform operations on behalf of your "*username*`/admin`" instance. Operations performed are subject to privileges granted or denied to this user instance by the Kerberos ACL file (see `kadm5.acl`(4)). You may perform administration as another user instance by using the `-p` option.

The remote version, `kadmin`, uses Kerberos authentication and an encrypted RPC to operate securely from anywhere on the network. It normally prompts for a password and authenticates the user to the Kerberos administration server, `kadmind`, whose service principal is *kadmin/admin*. Some options specific to the remote version permit the password prompt to be bypassed. The `-c` option searches the named credentials cache for a valid ticket for the *kadmin/admin* service and uses it to authenticate the user to the Kerberos admin server without a password. The `-k` option searches a keytab for a credential to authenticate to the *kadmin/admin* service, and again no password is collected. If `kadmin` has collected a password, it requests a *kadmin/admin* Kerberos service ticket from the KDC, and uses that service ticket to interact with `kadmind`.

The local version, `kadmin.local`, must be run with an effective UID of root, and normally uses a key from the `/var/krb5/.k5.`*realm* stash file (see `kdb5_util`(1M)) to decrypt information from the database rather than prompting for a password. The `-m` option will bypass the `.k5.`*realm* stash file and prompt for the master password.

OPTIONS

`-r` *realm*
Use *realm* as the default database realm.

`-p` *principal*
Authenticate *principal* to the *kadmin/admin* service. Otherwise, `kadmin` will append `/admin` to the primary principal name of the default credentials cache, the value of the `USER` environment variable, or the username as obtained with `getpwuid`, in that order of preference.

`-k` [`-t` *keytab*]
Use the default keytab (`-k`) or a specific keytab (`-t` *keytab*) to decrypt the KDC response instead of prompting for a password. In this case, the default principal will be `host/hostname`. This is primarily used for keytab maintenance.

`-c` *credentials_cache*
Search *credentials_cache* for a service ticket for the *kadmin/admin* service; it can be acquired with the `kinit`(1) program. If this option is not specified, `kadmin` requests a new service ticket from the KDC, and stores it in its own temporary credentials cache.

`-w` *password*
Use *password* instead of prompting for one. Note that placing the password for a Kerberos principal with administration access into a shell script can be dangerous if unauthorized users gain read access to the script or can read arguments of this command through `ps`(1).

`-q` *query*
Pass *query* directly to `kadmin`, which will perform *query* and then exit. This can be useful for writing scripts.

`-s` *admin_server*[*:port*]
Administer the specified admin server at the specified port number (*port*). This can be useful in administering a realm not known to your client.

`-d` *dbname*
Specify a non-standard database name. [Local only]

`-e` *"enc:salt ..."*
Specify a different encryption type and/or key salt. [Local only]

`-m`
Accept the database master password from the keyboard rather than using the `/var/krb5/.k5.`*realm* stash file. [Local only]

`-D`
Turn on debug mode. [Local only]

COMMANDS

`list_requests`
Lists all the commands available for `kadmin`. Aliased by `lr` and `?`.

`get_privs`
Lists the current Kerberos administration privileges (ACLs) for the principal that is currently running `kadmin`. The privileges are based on the `/etc/krb5/kadm5.acl` file on the master KDC. Aliased by `getprivs`.

`add_principal` [*options*] *newprinc*
Creates a new principal, *newprinc*, prompting twice for a password. If the `-policy` option is not specified and a policy named `default` exists, then the `default` policy is assigned to the principal; note that the assignment of the `default` policy occurs automatically only when a principal is first created, so the `default` policy must already exist for the assignment to occur. The automatic assignment of the `default` policy can be suppressed with the `-clearpolicy` option. This command requires the `add` privilege. Aliased by `addprinc` and `ank`. The options are:

`-expire` *expdate*
Expiration date of the principal. See the `Time Formats` section for the valid absolute time formats that you can specify for *expdate*.

`-pwexpire` *pwexpdate*
Password expiration date. See the `Time Formats` section for the valid absolute time formats that you can specify for *pwexpdate*.

`-maxlife` *maxlife*
Maximum ticket life for the principal. See the `Time Formats` section for the valid time duration formats that you can specify for *maxlife*.

`-maxrenewlife` *maxrenewlife*
Maximum renewable life of tickets for the principal. See the `Time Formats` section for the valid time duration formats that you can specify for *maxrenewlife*.

`-kvno` *kvno*
Explicitly set the key version number.

`-policy` *policy*
Policy used by the principal. If both the `-policy` and `-clearpolicy` options are not specified, the `default` policy is used if it exists; otherwise, the principal will have no policy. Also note that the password and principal name must be different when you add a new principal with a specific policy or the `default` policy.

`-clearpolicy`
`-clearpolicy` prevents the `default` policy from being assigned when `-policy` is not specified. This option has no effect if the `default` policy does not exist.

{`-`|`+`}`allow_postdated`
`-allow_postdated` prohibits the principal from obtaining postdated tickets. (Sets the `KRB5_KDB_DISALLOW_POSTDATED` flag.) `+allow_postdated` clears this flag.

{`-`|`+`}`allow_forwardable`
`-allow_forwardable` prohibits the principal from obtaining forwardable tickets. (Sets the `KRB5_KDB_DISALLOW_FORWARDABLE` flag.) `+allow_forwardable` clears this flag.

{`-`|`+`}`allow_renewable`
`-allow_renewable` prohibits the principal from obtaining renewable tickets. (Sets the `KRB5_KDB_DISALLOW_RENEWABLE` flag.) `+allow_renewable` clears this flag.

{`-`|`+`}`allow_proxiable`
`-allow_proxiable` prohibits the principal from obtaining proxiable tickets. (Sets the `KRB5_KDB_DISALLOW_PROXIABLE` flag.) `+allow_proxiable` clears this flag.

{-|+}allow_dup_skey
-allow_dup_skey disables user-to-user authentication for the principal by prohibiting this principal from obtaining a session key for another user. (Sets the KRB5_KDB_DISALLOW_DUP_SKEY flag.) +allow_dup_skey clears this flag.

{-|+}requires_preauth
+requires_preauth requires the principal to preauthenticate before being allowed to kinit. (Sets the KRB5_KDB_REQUIRES_PRE_AUTH flag.) -requires_preauth clears this flag.

{-|+}requires_hwauth
+requires_hwauth requires the principal to preauthenticate using a hardware device before being allowed to kinit. (Sets the KRB5_KDB_REQUIRES_HW_AUTH flag.) -requires_hwauth clears this flag.

{-|+}allow_svr
-allow_svr prohibits the issuance of service tickets for the principal. (Sets the KRB5_KDB_DISALLOW_SVR flag.) +allow_svr clears this flag.

{-|+}allow_tgs_req
–allow_tgs_req specifies that a Ticket-Granting Service (TGS) request for a service ticket for the principal is not permitted. This option is useless for most things. +allow_tgs_req clears this flag. The default is +allow_tgs_req. In effect, –allow_tgs_req sets the KRB5_KDB_DISALLOW_TGT_BASED flag on the principal in the database.

{-|+}allow_tix
–allow_tix forbids the issuance of any tickets for the principal. +allow_tix clears this flag. The default is +allow_tix. In effect, –allow_tix sets the KRB5_KDB_DISALLOW_ALL_TIX flag on the principal in the database.

{-|+}needchange
+needchange sets a flag in attributes field to force a password change; –needchange clears it. The default is –needchange. In effect, +needchange sets the KRB5_KDB_REQUIRES_PWCHANGE flag on the principal in the database.

{-|+}password_changing_service
+password_changing_service sets a flag in the attributes field marking this as a password change service principal (useless for most things). –password_changing_service clears the flag. This flag intentionally has a long name. The default is –password_changing_service. In effect, +password_changing_service sets the KRB5_KDB_PWCHANGE_SERVICE flag on the principal in the database.

-randkey
Sets the key of the principal to a random value.

-pw *password*
Sets the key of the principal to the specified string and does not prompt for a password. Note that using this option in a shell script can be dangerous if unauthorized users gain read access to the script.

Example:

```
kadmin: addprinc tlyu/admin
WARNING: no policy specified for "tlyu/admin@ACME.COM";
defaulting to no policy.
Enter password for principal tlyu/admin@ACME.COM:
Re-enter password for principal tlyu/admin@ACME.COM:
Principal "tlyu/admin@ACME.COM" created.
kadmin:
```

Errors:

KADM5_AUTH_ADD (requires add privilege)

KADM5_BAD_MASK (should not happen)

KADM5_DUP (principal exists already)

KADM5_UNK_POLICY (policy does not exist)

KADM5_PASS_Q_* (password quality violations)

delete_principal [-force] *principal*

Deletes the specified principal from the database. This command prompts for deletion, unless the -force option is given. This command requires the delete privilege. Aliased by delprinc.

Example:

```
kadmin: delprinc mwm_user
Are you sure you want to delete the principal
"mwm_user@ACME.COM"? (yes/no): yes
Principal "mwm_user@ACME.COM" deleted.
Make sure that you have removed this principal from
all kadmind ACLs before reusing.
kadmin:
```

Errors:

KADM5_AUTH_DELETE (requires delete privilege)

KADM5_UNK_PRINC (principal does not exist)

modify_principal [*options*] *principal*

Modifies the specified principal, changing the fields as specified. The options are as above for add_principal, except that password changing is forbidden by this command. In addition, the option -clearpolicy will clear the current policy of a principal. This command requires the modify privilege. Aliased by modprinc.

Errors:

KADM5_AUTH_MODIFY (requires modify privilege)

KADM5_UNK_PRINC (principal does not exist)

KADM5_UNK_POLICY (policy does not exist)

KADM5_BAD_MASK (should not happen)

`change_password` [*options*] *principal*
Changes the password of *principal*. Prompts for a new password if neither `-randkey` or `-pw` is specified. Requires the `changepw` privilege, or that the principal that is running the program to be the same as the one changed. Aliased by `cpw`. The following options are available:

`-randkey`
Sets the key of the principal to a random value.

`-pw` *password*
Sets the password to the specified string. Not recommended.

Example:

```
kadmin: cpw systest
Enter password for principal systest@ACME.COM:
Re-enter password for principal systest@ACME.COM:
Password for systest@ACME.COM changed.
kadmin:
```

Errors:
`KADM5_AUTH_MODIFY` (requires the `modify` privilege)

`KADM5_UNK_PRINC` (principal does not exist)

`KADM5_PASS_Q_*` (password policy violation errors)

`KADM5_PADD_REUSE` (password is in principal's password history)

`KADM5_PASS_TOOSOON` (current password minimum life not expired)

`get_principal` [`-terse`] *principal*
Gets the attributes of *principal*. Requires the `inquire` privilege, or that the principal that is running the program to be the same as the one being listed. With the `-terse` option, outputs fields as quoted tab-separated strings. Aliased by `getprinc`.

Examples:

```
kadmin: getprinc tlyu/admin
Principal: tlyu/admin@ACME.COM
Expiration date: [never]
Last password change: Mon Aug 12 14:16:47 EDT 1996
Password expiration date: [none]
Maximum ticket life: 0 days 10:00:00
Maximum renewable life: 7 days 00:00:00
Last modified: Mon Aug 12 14:16:47 EDT 1996
(example_user/admin@ACME.COM)
Last successful authentication: [never]
Last failed authentication: [never]
Failed password attempts: 0
Number of keys: 2 Key: vno 1, DES cbc mode with CRC-32,
no salt Key: vno 1, DES cbc mode with CRC-32,
Version 4 Attributes:
Policy: [none]
kadmin: getprinc -terse systest
```

```
systest@ACME.COM 3    86400    604800    1 785926535    753241234
785900000
tlyu/admin@ACME.COM    786100034    0    0
kadmin:
```

Errors:

KADM5_AUTH_GET (requires the get [inquire] privilege)

KADM5_UNK_PRINC (principal does not exist)

`list_principals` [*expression*]

Retrieves all or some principal names. *expression* is a shell-style glob expression that can contain the wild-card characters ?, *, and []'s. All principal names matching the expression are printed. If no expression is provided, all principal names are printed. If the expression does not contain an "@" character, an "@" character followed by the local realm is appended to the expression. Requires the `list` privilege. Aliased by `listprincs`, `get_principals`, and `getprincs`.

Examples:

```
kadmin: listprincs test*
test3@ACME.COM
test2@ACME.COM
test1@ACME.COM
testuser@ACME.COM
kadmin:
```

`add_policy` [*options*] *policy*

Adds the named policy to the policy database. Requires the `add` privilege. Aliased by `addpol`. The following options are available:

`-maxlife` *maxlife*

sets the maximum lifetime of a password. See the `Time Formats` section for the valid time duration formats that you can specify for *maxlife*.

`-minlife` *minlife*

sets the minimum lifetime of a password. See the `Time Formats` section for the valid time duration formats that you can specify for *minlife*.

`-minlength` *length*

sets the minimum length of a password.

`-minclasses` *number*

sets the minimum number of character classes allowed in a password. The valid values are:

`1`	only letters (himom)
`2`	both letters and numbers (hi2mom)
`3`	letters, numbers, and punctuation (hi2mom!)

`-history` *number*

sets the number of past keys kept for a principal.

Errors:
KADM5_AUTH_ADD (requires the add privilege)

KADM5_DUP (policy already exists)

delete_policy *policy*
Deletes the named policy. Prompts for confirmation before deletion. The command will fail if the policy is in use by any principals. Requires the delete privilege. Aliased by delpol.

Example:

```
kadmin: del_policy guests
Are you sure you want to delete the
policy "guests"? (yes/no): yes
Policy "guests" deleted.
kadmin:
```

Errors:
KADM5_AUTH_DELETE (requires the delete privilege)

KADM5_UNK_POLICY (policy does not exist)

KADM5_POLICY_REF (reference count on policy is not zero)

modify_policy [*options*] *policy*
Modifies the named policy. Options are as above for add_policy. Requires the modify privilege. Aliased by modpol.

Errors:
KADM5_AUTH_MODIFY (requires the modify privilege)

KADM5_UNK_POLICY (policy does not exist)

get_policy [-terse] *policy*
Displays the values of the named policy. Requires the inquire privilege. With the -terse flag, outputs the fields as quoted strings separated by tabs. Aliased by getpol.

Examples:

```
kadmin: get_policy admin
Policy: admin
Maximum password life: 180 days 00:00:00
Minimum password life: 00:00:00
Minimum password length: 6
Minimum number of password character classes: 2
Number of old keys kept: 5
Reference count: 17
kadmin: get_policy -terse
admin admin    15552000    0    6    2    5    17
kadmin:
```

Errors:
KADM5_AUTH_GET (requires the get privilege)

KADM5_UNK_POLICY (policy does not exist)

list_policies [*expression*]
Retrieves all or some policy names. *expression* is a shell-style glob expression that can contain the wild-card characters ?, *, and []'s. All policy names matching the expression are printed. If no expression is provided, all existing policy names are printed. Requires the list privilege. Aliased by listpols, get_policies, and getpols.

Examples:

```
kadmin: listpols
test-pol dict-only once-a-min test-pol-nopw
kadmin: listpols t*
test-pol test-pol-nopw kadmin:
```

ktadd [-k *keytab*] [-q] [*principal* | -glob *princ-exp*] [...]
Adds a principal or all principals matching *princ-exp* to a keytab, randomizing each principal's key in the process. Requires the inquire and changepw privileges. An entry for each of the principal's unique encryption types is added, ignoring multiple keys with the same encryption type but different salt types. If the -k argument is not specified, the default keytab file, /etc/krb5/krb5.keytab, is used. If the -q option is specified, less status information is displayed. Aliased by xst. The -glob option requires the list privilege. Also, note that if you use -glob to create a keytab, you need to remove /etc/krb5/kadm5.keytab and create it again if you want to use -p */admin with kadmin.

princ-exp follows the same rules described for the list_principals command.

Example:

```
kadmin: ktadd -k /tmp/new-keytab nfs/chicago
Entry for principal nfs/chicago with kvno 2,
encryption type DES-CBC-CRC added to keytab
WRFILE:/tmp/new-keytab.
kadmin:
```

ktremove [-k *keytab*] [-q] *principal* [*kvno* | all | old]
Removes entries for the specified principal from a keytab. Requires no privileges, since this does not require database access. If all is specified, all entries for that principal are removed; if old is specified, all entries for that principal except those with the highest kvno are removed. Otherwise, the value specified is parsed as an integer, and all entries whose kvno match that integer are removed. If the -k argument is not specified, the default keytab file, /etc/krb5/krb5.keytab, is used. If the -q option is specified, less status information is displayed. Aliased by ktrem.

Example:

```
kadmin: ktremove -k /tmp/new-keytab nfs/chicago
Entry for principal nfs/chicago with kvno 2
removed from keytab
WRFILE:/tmp/new-keytab.
kadmin:
```

`quit`
Quits `kadmin`. Aliased by `exit` and `q`.

Time Formats

Various commands in `kadmin` can take a variety of time formats, specifying time durations or absolute times. The `kadmin` option variables *maxrenewlife*, *maxlife*, and *minlife* are time durations, whereas *expdate* and *pwexpdate* are absolute times.

Examples:

```
kadmin: modprinc -expire "12/31 7pm" jdb
kadmin: modprinc -maxrenewlife "2 fortnight" jdb
kadmin: modprinc -pwexpire "this sunday" jdb
kadmin: modprinc -expire never jdb
kadmin: modprinc -maxlife "7:00:00pm tomorrow" jdb
```

Note that times which do not have the "ago" specifier default to being absolute times, unless they appear in a field where a duration is expected. In that case, the time specifier will be interpreted as relative. Specifying "ago" in a duration can result in unexpected behavior.

The following time formats and units can be combined to specify a time. The time and date format examples are based on the date and time of July 2, 1999, 1:35:30 p.m.

Time Format	Examples
hh[:*mm*][:*ss*][am/pm/a.m./p.m.]	1p.m., 1:35, 1:35:30pm

Variable	Description
hh	hour (12-hour clock, leading zero permitted but not required)
mm	minutes
ss	seconds

Date Format	Examples
mm/dd[/*yy*]	07/02, 07/02/99
yyyy-*mm*-dd	1999-07-02
dd-*month*-*yyyy*	02-July-1999
month [,*yyyy*]	Jul 02, July 02,1999
dd *month*[*yyyy*]	02 JULY, 02 july 1999

Variable Description

`dd`	day
mm	month
yy	year within century (00-38 is 2000 to 2038; 70-99 is 1970 to 1999)
yyyy	year including century
month	locale's full or abbreviated month name

`Time Units`	`Examples`
[+ | - #] year	"-2 year"
[+ | - #] month	"2 months"
[+ | - #] fortnight	
[+ | - #] week	
[+ | - #] day	
[+ | - #] hour	
[+ | - #] minute	
[+ | - #] min	
[+ | - #] second	
[+ | - #] sec	
tomorrow	
yesterday	
today	
now	
this	"this year"
last	"last saturday"
next	"next month"
sunday	
monday	
tuesday	
wednesday	
thursday	
friday	

saturday never	

You can also use the following time modifiers: first, second, third, fourth, fifth, sixth, seventh, eighth, ninth, tenth, eleventh, twelfth, and ago.

FILES

`/var/krb5/principal.db`
Kerberos principal database.

`/var/krb5/principal.kadm5`
Kerberos administrative database. Contains policy information.

`/var/krb5/principal.kadm5.lock`
Lock file for the Kerberos administrative database. This file works backwards from most other lock files (that is, `kadmin` will exit with an error if this file does *not* exist).

`/var/krb5/kadm5.dict`
Dictionary of strings explicitly disallowed as passwords.

`/etc/krb5/kadm5.acl`
List of principals and their `kadmin` administrative privileges.

`/etc/krb5/kadm5.keytab`
Keytab for *kadmin/admin* principal.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWkdcu

SEE ALSO

`kpasswd`(1), `gkadmin`(1M), `kadmind`(1M), `kdb5_util`(1M), `kadm5.acl`(4), `kdc.conf`(4), `krb5.conf`(4), `attributes`(5), `SEAM`(5)

HISTORY

The `kadmin` progam was originally written by Tom Yu at MIT, as an interface to the OpenVision Kerberos administration program.

DIAGNOSTICS

The `kadmin` command is currently incompatible with the MIT `kadmind` daemon interface, so you cannot use this command to administer an MIT-based Kerberos database. However, SEAM-based Kerberos clients can still use a MIT-based KDC.

NAME

kadmind – Kerberos administration daemon

SYNOPSIS

/usr/lib/kadmind [-d] [-m] [-p *port-number*] [-r *realm*]

DESCRIPTION

kadmind runs on the master key distribution center (KDC), which stores the principal and policy databases. kadmind accepts remote requests to administer the information in these databases. Remote requests are sent, for example, by kpasswd(1), gkamdin(1M), and kadmin(1M) commands, all of which are clients of kadmind. When you install a KDC, kadmind is set up in the init scripts to start automatically when the KDC is rebooted.

kadmind requires a number of configuration files to be set up for it to work:

/etc/krb5/kdc.conf
: The KDC configuration file contains configuration information for the KDC and the Kerberos administration system. kadmind understands a number of configuration variables (called relations) in this file, some of which are mandatory and some of which are optional. In particular, kadmind uses the acl_file, dict_file, admin_keytab, and kadmind_port relations in the [*realms*] section. Refer to the kdc.conf(4) man page for information regarding the format of the KDC configuration file.

/etc/krb5/kadm5.keytab
: kadmind requires akeytab (key table) containing correct entries for the kadmin/admin and kadmin/changepw principals for every realm that kadmind answers requests. The keytab can be created with the kadmin.local(1M), kdb5_util(1M) command. The location of the keytab is determined by the admin_keytab relation in the kdc.conf(4) file.

/etc/krb5/kadm5.acl
: kadmind uses an ACL (access control list) to determine which principals are allowed to perform Kerberos administration actions. The path of the ACL file is determined by the acl_file relation in the kdc.conf file. See kdc.conf(4). For information regarding the format of the ACL file, refer to kadm5.acl(4).

After kadmind begins running, it puts itself in the background and disassociates itself from its controlling terminal.

OPTIONS

The following options are supported:

-d
: Specifies that kadmind does not put itself in the background and does not disassociate itself from the terminal. In normal operation, you should use the default behavior, which is to allow the daemon to put itself in the background.

-m
: Specifies that the master database password should be retrieved from the keyboard rather than from the stash file. When using -m, the kadmind daemon receives the password prior to putting itself in the background. If used in combination with the -d option, you must explicitly place the daemon in the background.

`-p` *port-number*	Specifies the port on which the `kadmind` daemon listens for connections. The default is controlled by the `kadmind_port` relation in the `kdc.conf`(4) file.
`-r` *realm*	Specifies the default realm that `kadmind` serves. If *realm* is not specified, the default *realm* of the host is used. `kadmind` answers requests for any realm that exists in the local KDC database and for which the appropriate principals are in its `keytab`.

FILES

`/var/krb5/principal.db`
Kerberos principal database.

`/var/krb5/principal.kadm5`
Kerberos administrative database containing policy information.

`/var/krb5/principal.kadm5.lock`
Kerberos administrative database lock file. This file works backwards from most other lock files (that is, `kadmin` exits with an error if this file does not exist).

`/var/krb5/kadm5.dict`
Dictionary of strings explicitly disallowed as passwords.

`/etc/krb5/kadm5.acl`
List of principals and their `kadmin` administrative privileges.

`/etc/krb5/kadm5.keytab`
Keytab for `kadmin/admin` principal.

`/etc/krb5/kdc.conf`
KDC configuration information.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWkdcu

SEE ALSO

`kpasswd`(1), `gkadmin`(1M), `kadmin`(1M), `kadmin.local`(1M), `kdb5_util`(1M), `kadm5.acl`(4), `kdc.conf`(4), `attributes`(5), `SEAM`(5)

NAME

kdb5_util – Kerberos Database maintenance utility

SYNOPSIS

/usr/sbin/kdb5_util [-d *dbname*] [-f *stashfile_name*] [-k *mkeytype*] [-m] [-M *mkeyname*] [-P *password*] [-r *realm*] *cmd*

DESCRIPTION

kdb5_util enables you to create, dump, load, and destroy the Kerberos V5 database. You can also use kdb5_util to create a stash file containing the Kerberos database master key.

OPTIONS

The following options are supported:

-d *dbname*
Specify the database name. .db is appended to whatever name is specified. You can specify an absolute path. If you do not specify the -d option, the default database name is /var/krb5/principal, which becomes /var/krb5/principal.db.

-f *stashfile_name*
Specify the stash file name. You can specify an absolute path.

-k *mkeytype*
Specify the master key type. Valid values are des-cbc-crc, des-cbc-md5, and des-cbc-raw.

-m
Enter the master key manually.

-M *mkeyname*
Specify the master key name.

-P *password*
Use the specified *password* instead of the stash file.

-r *realm*
Use *realm* as the default database realm.

OPERANDS

The following operands are supported:

cmd Specifies whether to create, destroy, dump, or load the database, or to create a stash file.

You can specify the following commands:

create -s
Creates the database specified by the -d option. You will be prompted for the database master password. If you specify -s, a stash file is created as specified by the -f option. If you did not specify -f, the default stash file name is /var/krb5/.k5.realm. If you use the -f, -k, or -M options when you create a database, then you must use the same options when modifying or destroying the database.

destroy
Destroys the database specified by the -d option.

`stash`
Creates a stash file. If `-f` was not specified, the default stash file name is `/var/krb5/.k5.realm`. You will be prompted for the master database password. This command is useful when you want to generate the stash file from the password.

`dump` [`-verbose`] [*filename*] [*principals*]
Dumps the Kerberos database to a flat file that can be used for loading or propagating to a slave KDC. See `kprop`(1M). Specify file name for a location to dump the Kerberos database. If *filename* is not specified, the principal data is printed to standard error. Specify `-verbose` to print out the principal names to standard error in addition to being dumping into the file. Use *principals* to specify the list of principals that should be dumped.

`load` [`-verbose`] [`-update`] *filename*
Loads the database specified by *dbname* (see `-d` option, above) with data from the file specified by *filename*, which must be a file created by the `dump` command. Use `-update` to specify that the existing database should be updated; otherwise, a new database is created. Specify `-verbose` to print out the principal names to standard error, in addition to being loaded.

EXAMPLES

EXAMPLE 1 Creating File that Contains Information about Two Principals

The following example creates a file named `slavedata` that contains the information about two principals, `jdb@ACME.COM` and `pak@ACME.COM`.

```
# /usr/krb5/bin/kdb5_util dump -verbose slavedata
jdb@ACME.COM pak@ACME.COM
```

FILES

`/var/krb5/principal.db`
Kerberos principal database.

`/var/krb5/principal.kadm5`
Kerberos administrative database. Contains policy information.

`/var/krb5/principal.kadm5.lock`
Lock file for the Kerberos administrative database. This file works backwards from most other lock files (that is, `kadmin` exits with an error if this file does not exist).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWkdcu

SEE ALSO

`kpasswd`(1), `gkadmin`(1M), `kadmin`(1M), `kadmind`(1M), `kadmin.local`(1M), `kadm5.acl`(4), `kdc.conf`(4), `attributes`(5), `SEAM`(5)

NAME

kdmconfig – configure or unconfigure keyboard, display, and mouse options for OpenWindows and internationalization

SYNOPSIS

kdmconfig

kdmconfig [-fv] [-s *hostname*]-c | -t | -u | -d *filename*

DESCRIPTION

The kdmconfig program configures or unconfigures the /etc/openwin/server/etc/OWconfig file with the keyboard, display, and mouse information relevant to a client's machine on IA based systems for Solaris software. kdmconfig can also be used to set up the *display*, *pointer*, and *keyboard* entries in the bootparams(4) database on a server machine or the monitor, *keyboard*, *display*, and *pointer* keywords in a sysidcfg(4) file. kdmconfig can only be run by the super-user. Upon completion of device selection, kdmconfig prompts the user to test the configuration, which is done by running the window system.

The kdmconfig program is normally run during installation and upon reboot, but it can also be run from the command line after the system has been installed. When configuring a client during an initial installation or a reconfigure reboot, the sysidconfig(1M) program will invoke kdmconfig with the -c option, and when the user executes the sys-unconfig(1M) program, kdmconfig will be executed with the -u option. Similarly, when you run kdmconfig from the command line, use the -u option to unconfigure the existing OpenWindows configuration. You can then rerun kdmconfig with the -cf options to create a new OpenWindows configuration. To edit the existing configuration, run kdmconfig from the command line without options. After each reboot, kdmconfig will be invoked by the system with the -t (test mode) option to ensure autoconfiguration capability and identify possible conflicts between the current configuration and the one recorded in the OWconfig file.

OPTIONS

The valid options are:

-c — Run the program in the configuration mode. This mode is used to create or update the OWconfig file. When invoked in this way, kdmconfig first looks for the relevant configuration information in the bootparams(4) databases. It also takes into account the information returned from device probes, unless the -s option is also used. The bootparams(4) databases available to the client are all of the /etc/bootparams files on servers on the same subnet as the client, provided the server machine is running the bootparamd(1M) daemon. kdmconfig is invoked with the -c option when called by sysidconfig(1M)

-d *filename* — Set up a sysidcfg(4) file. This option displays the same screens as the -c option, but the information you specify is saved as sysidcfg(4) keywords (monitor, *keyboard*, *display*, and *pointer*). This enables you to use a sysidcfg (4) file to preconfigure a system's device information and bypass kdmconfig during an installation.

filename is the `sysidcfg`(4) file that is created, and it is created in the directory where `kdmconfig` is being run unless a path is specified. If filename already exists in the specified directory, the keywords are appended to the existing file.

`-f` Force screens mode. When this option is invoked, no network probing will be performed. This is helpful when debugging the client's configuration environment. Note that the `-s` option implies the use of `-f`, bypassing network probing when setting up a server.

`-s` *hostname* Set up the `bootparams`(4) database on this machine for the specified client. This option presents the same screens as it does when run on a client, but instead writes the resulting information to the `/etc/bootparams` file. Also, `-s` implies the use of the `-f` option. That is, the program will always present the screens to the user when invoked this way. This option will reconfigure the `nsswitch.conf`(4) file to look for a `bootparams`(4) database on a local server. This option is only available to the super-user.

`-t` Run the program in test mode. In this mode, `kdmconfig` will use device probe information to determine whether the `OWconfig` file contains complete and up-to-date information about the keyboard, display, and mouse. If the information is accurate, `kdmconfig` will exit silently. Otherwise, `kdmconfig` will prompt for the super-user password and proceed to a normal editing session (as though it had been run without options).

`-u` Unconfigure the system, returning it to an "out-of-the-box" state. In this state, the factory default keyboard, mouse, and display are selected as a result of removing the device configuration entries from the `/etc/openwin/server/etc/OWconfig` file. This may result in an unusable configuration for the display server.

`-v` Enable verbose mode. Normally, `kdmconfig` will not produce any output. This option is helpful for debugging, as it records the different actions taken by `kdmconfig` on `stderr`.

No Options

Run without options, `kdmconfig` is used to edit the current configuration. `kdmconfig` uses the information from the `OWconfig` file in addition to information obtained from the `bootparams`(4) file and from device probes. In other respects, it is similar to using the `-c` option of `kdmconfig`.

FILES

`/etc/openwin/server/etc/OWconfig`
OpenWindows configuration file

`/etc/bootparams`
contains list of clients that diskless clients use for booting

`/etc/nsswitch.conf`
name service configuration file

IA Only

`/dev/openprom` installed devices and properties

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Architecture	IA
Availability	SUNWos86r

SEE ALSO

`bootparamd`(1M), `sys-unconfig`(1M), `sysidconfig`(1M), `bootparams`(4), `nsswitch.conf`(4), `sysidcfg`(4), `attributes`(5)

NAME

kernel – UNIX system executable file containing basic operating system services

SYNOPSIS

kernel-name [-afsrvx]

DESCRIPTION

The operating system image, or kernel, is the collection of software comprising the image files (`unix` and `genunix`) and the modules loaded at any instant in time. The system will not function without a kernel to control it.

The kernel is loaded by the `boot`(1M) command in a machine-specific way. The kernel may be loaded from disk, CD-ROM, or DVD (`diskfull boot`) or over the network (`diskless boot`). In either case, the directories under `/platform` and `/kernel` must be readable and must contain executable code which is able to perform the required kernel service. If the `-a` flag is given, the user is able to supply different pathnames for the default locations of the kernel and modules. See `boot`(1M) for more information on loading a specific kernel.

If the kernel name is not explicitly specified, then on systems capable of supporting the 64-bit kernel, the boot program will attempt to load the 64-bit kernel in preference to the 32-bit kernel by default. See `boot`(1m).

The `moddir` variable contains a colon-separated list of directories that the kernel searches for modules. `moddir` can be set in the `/etc/system` file. The minimal default is `/platform/`*platform-name*`/kernel:/kernel:/usr/kernel`, but this default they be overridden by a specific platform. It is common for many systems to override the default path with:

```
/platform/platform-name/kernel:/platform/hardware-class-name\
     /kernel:/kernel:/usr/kernel
```

where *platform-name* can be found using the `-i` option of `uname`(1), and *hardware-class-name* can be found using the `-m` option of `uname`(1).

The kernel configuration can be controlled using the `/etc/system` file (see `system`(4)).

`genunix` is the platform-independent component of the base kernel.

OPTIONS

The following options are supported:

`-a` Asks the user for configuration information, such as where to find the system file, where to mount `root`, and even override the name of the kernel itself. Default responses will be contained in square brackets ([ ]), and the user may simply enter RETURN to use the default response (note that RETURN is labeled ENTER on some keyboards). To help repair a damaged `/etc/system` file, enter **`/dev/null`** at the prompt that asks for the pathname of the system configuration file. See `system`(4).

`-f` Causes Autoclient systems to `flush` and reinitialize the client system's local cache. This flag is ignored for all non-Autoclient systems.

`-r` Reconfiguration boot. The system will probe all attached hardware devices and assign nodes in the file system to represent only those devices actually

found. It will also configure the logical namespace in `/dev` as well as the physical namespace in `/devices`. See `add_drv`(1M) and `rem_drv`(1M) for additional information about maintaining device drivers.

`-s` Boots only to init level 's'. See `init`(1M).

`-v` Boots with verbose messages enabled. If this flag is not given, the messages are still printed, but the output is directed to the system logfile. See `syslogd`(1M).

`-x` Does not boot in clustered mode. This option only has an effect when a version of Sun Cluster software that supports this option has been installed.

EXAMPLES See `boot`(1M) for examples and instructions on how to boot.

FILES

`/kernel`
Contains kernel components common to all platforms within a particular instruction set that are needed for booting the system. of the core image file.

`/platform/`*platform-name*`/kernel`
The platform-specific kernel components.

`/platform/`*hardware-class-name*`/kernel`
The kernel components specific to this hardware class.

`/usr/kernel`
Contains kernel components common to all platforms within a particular instruction set.

The directories in this section can potentially contain the following subdirectories:

drv Loadable device drivers

exec The modules that execute programs stored in various file formats.

fs File system modules

misc Miscellaneous system-related modules

sched Operating system schedulers

strmod System V STREAMS loadable modules

sys Loadable system calls

SPARC

cpu Processor specific modules

tod Time-Of-Day hardware interface modules

Additionally, some of the subdirectories mentioned above contain `sparcv9` subdirectories that contain 64-bit versions of the same module classes. For example, `/kernel/drv/sparcv9` and `/platform/sun4u/kernel/cpu/sparcv9`.

IA

mach IA hardware support

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcar, SUNWcarx

SEE ALSO

`uname`(1), `isainfo`(1), `add_drv`(1M), `boot`(1M), `init`(1M), `kadb`(1M), `rem_drv`(1M), `savecore`(1M), `syslogd`(1M), `system`(4), `attributes`(5)

SPARC Only

`monitor`(1M)

DIAGNOSTICS

The kernel gives various warnings and error messages. If the kernel detects an unrecoverable fault, it will panic or halt.

BUGS

Bugs in the kernel often result in kernel panics.

Reconfiguration boot does not currently remove filesystem entries for devices that have been physically removed from the system.

NAME

keyserv – server for storing private encryption keys

SYNOPSIS

keyserv [-c] [-d | -e] [-D] [-n] [-s *sizespec*]

DESCRIPTION

keyserv is a daemon that is used for storing the private encryption keys of each user logged into the system. These encryption keys are used for accessing secure network services such as secure NFS and NIS+.

Normally, root's key is read from the file /etc/.rootkey when the daemon is started. This is useful during power-fail reboots when no one is around to type a password.

keyserv will not start up if the system does not have a secure rpc domain configured. Set up the domain name by using the /usr/bin/domainname command. Usually the /etc/init.d/inetinit script reads the domain from /etc/defaultdomain. Invoking the domainname command without arguments tells you if you have a domain set up.

The /etc/default/keyserv file contains the following default parameter settings. See FILES.

ENABLE_NOBODY_KEYS — Specifies whether default keys for nobody are used. ENABLE_NOBODY_KEYS=NO is equivalent to the -d command-line option. The default value for ENABLE_NOBODY_KEYS is YES.

OPTIONS

-c — Do not use disk caches. This option overrides any -s option.

-D — Run in debugging mode and log all requests to keyserv.

-d — Disable the use of default keys for nobody. See FILES.

-e — Enable the use of default keys for nobody. This is the default behavior. See FILES.

-n — Root's secret key is not read from /etc/.rootkey. Instead, keyserv prompts the user for the password to decrypt root's key stored in the publickey database and then stores the decrypted key in /etc/.rootkey for future use. This option is useful if the /etc/.rootkey file ever gets out of date or corrupted.

-s *sizespec* — Specify the size of the extended Diffie-Hellman common key disk caches. The *sizespec* can be one of the following forms:

mechtype=size — size is an integer specifying the maximum number of entries in the cache, or an integer immediately followed by the letter *M*, denoting the maximum size in MB.

size — This form of *sizespec* applies to all caches.

See nisauthconf(1M) for mechanism types. Note that the des mechanism, AUTH_DES, does not use a disk cache.

FILES

`/etc/.rootkey`

`/etc/default/keyserv` Contains default settings. You can use command-line options to override these settings.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`keylogin`(1), `keylogout`(1), `nisauthconf`(1M), `publickey`(4), `attributes`(5)

NOTES

NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

killall – kill all active processes

SYNOPSIS

`/usr/sbin/killall` [*signal*]

DESCRIPTION

`killall` is used by `shutdown`(1M) to kill all active processes not directly related to the shutdown procedure.

`killall` terminates all processes with open files so that the mounted file systems will be unbusied and can be unmounted.

`killall` sends *signal* (see `kill`(1)) to the active processes. If no *signal* is specified, a default of `15` is used.

The `killall` command can be run only by the super-user.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`kill`(1), `ps`(1), `fuser`(1M), `shutdown`(1M), `signal`(3C), `attributes`(5)

NAME

kprop – Kerberos database propagation program

SYNOPSIS

```
/usr/lib/kprop [-d] [-f file] [-p port-number] [-r realm] [-s keytab]
     [host]
```

DESCRIPTION

`kprop` is a command-line utility used for propagating a Kerberos database from a master KDC to a slave KDC. This command must be run on the master KDC. See the *Solaris System Administration Guide, Vol. 6* on how to set up periodic propagation between the master KDC and slave KDCs.

To propagate a Kerberos database, the following conditions must be met:

- The slave KDCs must have an `/etc/krb5/kpropd.acl` file that contains the principals for the master KDC and all the slave KDCs.
- A keytab containing a host principal entry must exist on each slave KDC.
- The database to be propagated must be dumped to a file using `kdb5_util`(1M).

OPTIONS

`-d`	Enable debug mode. Default is debug mode disabled.
`-f` *file*	File to be sent to the slave KDC. Default is the `/var/krb5/slave_datatrans` file.
`-p` *port-number*	Propagate *port-number*. Default is port `754`.
`-r` *realm*	Realm where propagation will occur. Default *realm* is the local realm.
`-s` *keytab*	Location of the keytab. Default location is `/etc/krb5/krb5.keytab`.

OPERANDS

The following operands are supported:

host	Name of the slave KDC.

EXAMPLES

EXAMPLE 1 Propagating the Kerberos Database

The following example propagates the Kerberos database from the `/tmp/slave_data` file to the slave KDC `london`. The machine `london` must have a host principal keytab entry and the `kpropd.acl` file must contain an entry for the all the KDCs.

```
# kprop -f /tmp/slave_data london
```

FILES

`/etc/krb5/kpropd.acl`
List of principals of all the KDCs; resides on each slave KDC.

`/etc/krb5/krb5.keytab`
Keytab for Kerberos clients.

`/var/krb5/slave_datatrans`
Kerberos database propagated to the KDC slaves.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWkdcu

SEE ALSO `kpasswd`(1), `gkadmin`(1M), `kadmind`(1M), `kadmin.local`(1M), `kdb5_util`(1M), `kadm5.acl`(4), `kdc.conf`(4), `attributes`(5), `SEAM`(5)

Solaris System Administration Guide, Security Services

NAME

krb5kdc – KDC daemon

SYNOPSIS

```
/usr/lib/krb5kdc [-d dbpath] [-r realm] [-R replaycachename] [-m]
     [-k masterenctype] [-M masterkeyname] [-p port] [-n]
```

DESCRIPTION

`krb5kdc` is the daemon that runs on the master and slave KDCs to process the Kerberos tickets. For Kerberos to function properly, `krb5kdc` must be running on at least one KDC that the Kerberos clients can access. Prior to running `krb5kdc`, you must initialize the Kerberos database using `kdb5_util`(1M). See the *Solaris System Administration Guide, Security Services* or instructions on setting up KDCs and initialize the Kerberos database.

OPTIONS

The following options are supported:

`-d` *dbpath*	Specify the path to the database; default value is `/var/krb5`.
`-r` *realm*	Specify the realm name; default is the local realm name.
`-R` *replaycachename*	Specify the replay cache name; default value is `krb5kdc_rcache`.
`-m`	Specify that the master key for the database is to be entered manually.
`-k` *masterenctype*	Specify the encryption type for encrypting the database. The default value is `des-cbc-crc`.
`-M` *masterkeyname*	Specify the principal to retrieve the master key for the database.
`-p` *port*	Specify the port that will be used by the KDC to listen for incoming requests.
`-n`	Specify that `krb5kdc` should not detach from the terminal.

FILES

`/var/krb5/principal.db`
Kerberos principal database.

`/var/krb5/principal.kadm5`
Kerberos administrative database. This file contains policy information.

`/var/krb5/principal.kadm5.lock`
Kerberos administrative database lock file. This file works backwards from most other lock files (that is, `kadmin` will exit with an error if this file does *not* exist).

`/etc/krb5/kdc.conf`
KDC configuration file. This file is read at startup.

`/etc/krb5/kpropd.acl`
File that defines the access control list for propagating the Kerberos database using `kprop`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWkdcu

SEE ALSO

`kpasswd`(1), `gkadmin`(1M), `kadmind`(1M), `kadmin.local`(1M), `kdb5_util`(1M), `kadm5.acl`(4), `kdc.conf`(4), `attributes`(5), `SEAM`(5)

Solaris System Administration Guide, Security Services

NAME

kstat – display kernel statistics

SYNOPSIS

```
kstat [-lpq] [-T u | d ] [-c class] [-m module] [-i instance] [-n name]
     [-s statistic] [interval [count]]

kstat [-lpq] [-T u | d ] [-c class] [module:instance:name:statistic...]
     [interval [count]]
```

DESCRIPTION

The `kstat` utility examines the available kernel statistics, or kstats, on the system and reports those statistics which match the criteria specified on the command line. Each matching statistic is printed with its module, instance, and name fields, as well as its actual value.

Kernel statistics may be published by various kernel subsystems, such as drivers or loadable modules; each kstat has a module field that denotes its publisher. Since each module may have countable entities (such as multiple disks associated with the `sd`(7D) driver) for which it wishes to report statistics, the kstat also has an instance field to index the statistics for each entity; kstat instances are numbered starting from zero. Finally, the kstat is given a name unique within its module.

Each kstat may be a special kstat type, an array of name-value pairs, or raw data. In the name-value case, each reported value is given a label, which we refer to as the statistic. Known raw and special kstats are given statistic labels for each of their values by `kstat`; thus, all published values can be referenced as *module*:*instance*:*name*:*statistic*.

When invoked without any module operands or options, kstat will match all defined statistics on the system. Example invocations are provided below. All times are displayed as fractional seconds since system boot.

OPTIONS

The tests specified by the following options are logically ANDed, and all matching kstats will be selected. A regular expression containing shell meta-characters must be protected from the shell by enclosing it with the appropriate quotes.

The argument for the `-c`, `-i`, `-m`, `-n`, and `-s` options may be specified as a shell glob pattern, or a Perl regular expression enclosed in '/' characters.

`-c` *class*	Displays only kstats that match the specified class.
`-i` *instance*	Displays only kstats that match the specified instance.
`-l`	Lists matching kstat names without displaying values.
`-m` *module*	Displays only kstats that match the specified module.
`-n` *name*	Displays only kstats that match the specified name.
`-p`	Displays output in parseable format. All example output in this document is given in this format. If this option is not specified, `kstat` produces output in a human-readable, table format.
`-q`	Displays no output, but return appropriate exit status for matches against given criteria.
`-s` *statistic*	Displays only kstats that match the specified statistic.

`-T d | u`	Displays a time stamp before each statistics block, either in `ctime`(3C) format ('d') or as an alphanumeric representation of the value returned by `time`(2) ('u').

OPERANDS

The following operands are supported:

module:*instance*:*name*:*statistic*	Alternate method of specifying module, instance, name, and statistic as described above. Each of the module, instance, name, or statistic specifiers may be a shell glob pattern or a Perl regular expression enclosed by '/' characters. It is possible to use both specifier types within a single operand. Leaving a specifier empty is equivalent to using the '*' glob pattern for that specifier.
interval	The number of seconds between reports.
count	The number of reports to be printed.

EXAMPLES

In the following examples, all the command lines in a block produce the same output, as shown immediately below. The exact statistics and values will of course vary from machine to machine.

EXAMPLE 1

```
example$ kstat -p -m unix -i 0 -n system_misc -s 'avenrun*'
example$ kstat -p -s 'avenrun*'
example$ kstat -p 'unix:0:system_misc:avenrun*'
example$ kstat -p ':::avenrun*'
example$ kstat -p ':::/^avenrun_/d+min$/'

unix:0:system_misc:avenrun_15min        3
unix:0:system_misc:avenrun_1min 4
unix:0:system_misc:avenrun_5min 2
```

EXAMPLE 2

```
example$ kstat -p -m cpu_stat -s 'intr*'
example$ kstat -p cpu_stat:::/^intr/

cpu_stat:0:cpu_stat0:intr       29682330
cpu_stat:0:cpu_stat0:intrblk    87
cpu_stat:0:cpu_stat0:intrthread 15054222
cpu_stat:1:cpu_stat1:intr       426073
cpu_stat:1:cpu_stat1:intrblk    51
cpu_stat:1:cpu_stat1:intrthread 289668
cpu_stat:2:cpu_stat2:intr       134160
cpu_stat:2:cpu_stat2:intrblk    0
cpu_stat:2:cpu_stat2:intrthread 131
cpu_stat:3:cpu_stat3:intr       196566
cpu_stat:3:cpu_stat3:intrblk    30
cpu_stat:3:cpu_stat3:intrthread 59626
```

EXAMPLE 2 *(Continued)*

EXAMPLE 3

```
example$ kstat -p :::state ':::avenrun*'
example$ kstat -p :::state :::/^avenrun/

cpu_info:0:cpu_info0:state      on-line
cpu_info:1:cpu_info1:state      on-line
cpu_info:2:cpu_info2:state      on-line
cpu_info:3:cpu_info3:state      on-line
unix:0:system_misc:avenrun_15min        4
unix:0:system_misc:avenrun_1min 10
unix:0:system_misc:avenrun_5min 3
```

EXAMPLE 4

```
example$ kstat -p 'unix:0:system_misc:avenrun*' 1 3
unix:0:system_misc:avenrun_15min        15
unix:0:system_misc:avenrun_1min 11
unix:0:system_misc:avenrun_5min 21

unix:0:system_misc:avenrun_15min        15
unix:0:system_misc:avenrun_1min 11
unix:0:system_misc:avenrun_5min 21

unix:0:system_misc:avenrun_15min        15
unix:0:system_misc:avenrun_1min 11
unix:0:system_misc:avenrun_5min 21
```

EXAMPLE 5

```
example$ kstat -p -T d 'unix:0:system_misc:avenrun*' 5 2
Thu Jul 22 19:39:50 1999
unix:0:system_misc:avenrun_15min        12
unix:0:system_misc:avenrun_1min 0
unix:0:system_misc:avenrun_5min 11

Thu Jul 22 19:39:55 1999
unix:0:system_misc:avenrun_15min        12
unix:0:system_misc:avenrun_1min 0
unix:0:system_misc:avenrun_5min 11
```

EXAMPLE 6

```
example$ kstat -p -T u 'unix:0:system_misc:avenrun*'
932668656
unix:0:system_misc:avenrun_15min        14
unix:0:system_misc:avenrun_1min 5
unix:0:system_misc:avenrun_5min 18
```

EXIT STATUS

The following exit values are returned:

0 One or more statistics were matched.

1 No statistics were matched.

2 Invalid command line options were specified.

3 A fatal error occurred.

FILES

`/dev/kstat` kernel statistics driver

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`sh`(1), `time`(2), `ctime`(3C)`gmatch`(3GEN), `kstat`(3KSTAT), `attributes`(5), `kstat`(7D), `sd`(7D), `kstat`(9S)

NOTES

If the pattern argument contains glob or Perl RE meta-characters which are also shell meta-characters, it will be necessary to enclose the pattern with appropriate shell quotes.

NAME

ktkt_warnd – Kerberos warning daemon

SYNOPSIS

/usr/lib/krb5/ktkt_warnd

DESCRIPTION

`ktkt_warnd` is a daemon on Kerberos clients that can warn users when their Kerberos tickets are about to expire. It is invoked by `inetd` when a ticket-granting ticket (TGT) is obtained for the first time, such as after using the `kinit` command. `ktkt_warnd` can be configured through the `/etc/krb5/warn.conf` file on the client.

FILES

`/etc/krb5/warn.conf`	Kerberos warning configuration file

SEE ALSO

`inetd`(1M), `warn.conf`(4), `SEAM`(5)

NAME

labelit – list or provide labels for file systems

SYNOPSIS

`labelit` `[-F` *FSType*`] [-V] [-o` *FSType-specific-options*`]` *special* `[`*operands*`]`

DESCRIPTION

The `labelit` utility is used to write or display labels on unmounted disk file systems.

OPTIONS

The following options are supported:

`-F` *FSType* — Specify the *FSType* on which to operate. The *FSType* should either be specified here or be determinable from `/etc/vfstab` by matching the *special* with an entry in the table. If no matching entry is found, the default file system type specified in `/etc/default/fs` will be used.

`-V` — Echo complete command line. This option may be used to verify and validate the command line. Additional information obtained using a `/etc/vfstab` lookup is included in the output. The command is not executed.

`-o` — Specify *FSType*-specific options. See the manual page for the `labelit` module specific to the file system type.

OPERANDS

The following operands are supported. If no operands are specified, `labelit` will display the value of the labels.

special — The disk partition (for example, `/dev/rdsk/c0t3d0s6`). The device may not be on a remote machine.

operands — *FSType*-specific operands. Consult the manual page of the *FSType*-specific `labelit` command for detailed descriptions.

USAGE

See `largefile`(5) for the description of the behavior of `labelit` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXIT STATUS

The following exit values are returned:

`0` — Write or display of labels was successful.

non-zero — An error occurred.

FILES

`/etc/vfstab` — List of default parameters for each file system.

`/etc/default/fs` — Default local file system type. Default values can be set for the following flags in `/etc/default/fs`. For example:

```
LOCAL=ufs
```

`LOCAL` — The default partition for a command if no *FSType* is specified.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO `labelit_hsfs`(1M), `labelit_udfs`(1M), `labelit_ufs`(1M), `volcopy`(1M), `vfstab`(4), `attributes`(5), `largefile`(5)

NOTES This utility may not be supported for all *FSTypes*.

NAME

labelit_hsfs – provide and print labels for hsfs file systems

SYNOPSIS

/usr/sbin/labelit `-F hsfs` [*generic_options*] `[-o` *specific_options*`]` *special*

DESCRIPTION

`labelit` can be used to provide labels for unmounted CD-ROM images (CD-ROMs may not be labeled, as they are read-only media).

generic_options are options supported by the generic `labelit` command.

If no *specific_options* are specified, `labelit` prints the current value of all label fields.

The *special* name should be the physical disk section (for example, `/dev/dsk/c0d0s6`).

OPTIONS

`-o` Use one or more of the following *name=value* pairs separated by commas (with no intervening spaces) to specify values for specific label fields. According to the ISO 9660 specification, only certain sets of characters may be used to fill in these labels. Thus, ''d-characters'' below refers to the characters 'A' through 'Z', the digits '0' through '9', and the '_' (underscore) character. ''a-characters'' below refers to 'A' through 'Z', '0' through '9', space, and the following characters: !"%&'()*+,-./:;<=>?_.

- `absfile=` Abstract file identifier, d-characters, 37 characters maximum.
- `applid=` Application identifier, d-characters, 128 characters maximum.
- `bibfile=` Bibliographic file identifier, d-characters, 37 characters maximum.
- `copyfile=` Copyright file identifier, d-characters, 128 maximum.
- `prepid=` Data preparer identifier, d-characters, 128 maximum.
- `pubid=` Publisher identifier, d-characters, 128 maximum.
- `sysid=` System identifier, a-characters, 32 maximum.
- `volid=` Volume identifier, d-characters, 32 maximum.
- `volsetid=` Volume set identifier, d-characters, 128 maximum.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`labelit`(1M), `volcopy`(1M), `attributes`(5)

NAME

labelit_udfs – provide and print labels for udf file systems

SYNOPSIS

`labelit` `-F udfs` [*generic_options*] [`-o` *specific_options*] *special* [*fsname volume*]

DESCRIPTION

The `labelit` command writes labels on an unmounted disk that contains a universal disk file (udf) system. These labels can be used to identify volumes.

OPTIONS

The following options are supported:

generic_options
Specify *generic_options* supported by the generic `labelit` command. See `labelit`(1M) for descriptions of supported options.

`-o` *specific_options*
Specify udfs-file-system-specific options in a comma-separated list with no intervening spaces. The following *specific_options* are available:

`lvinfo1=`*string* Specify information to be inserted in the `LVInfo1` field of the Implementation Use Volume Descriptor. Information in `LVInfo1` is generally used to identify the person creating the file system. The maximum length of the string specified is 35 bytes.

`lvinfo2=`*string* Specify information to be inserted into the `LVInfo2` field of the Implementation Use Volume Descriptor. Information in `LVInfo2` is generally used to identify the organization responsible for creating the file system. The maximum length of the string specified is 35 bytes.

`lvinfo3=`*string* Specify information to be inserted into the `LVInfo3` field of the Implementation Use Volume Descriptor. Information in `LVInfo3` is generally used to identify the contact information for the medium. The maximum length of the string specified is 35 bytes.

OPERANDS

The following operands are supported:

special Specify *special* as the physical disk slice, for example, `/dev/rdsk/c0t0d0s6`. The device can not be on a remote machine.

fsname Specify *fsname* as the mount point, (for example, `root`, `u1`, and so forth), of the file system.

volume Specify *volume* as the physical volume name.

If none of the options (*fsname*, *volume*, *specific_options*) is specified, `labelit` prints the current values of *fsname*, *volume*, `LVInfo1`, `LVInfo2` and `LVInfo3`.

EXIT STATUS

The following exit values are returned:

0 Successful completion.

non-zero An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWudf

SEE ALSO

`labelit`(1M), `attributes`(5)

NAME

labelit_ufs – provide and print labels for ufs file systems

SYNOPSIS

`labelit` `-F ufs` [*generic_options*] *special* [*fsname volume*]

DESCRIPTION

`labelit` is used to write labels on unmounted disk file systems. Such labels may be used to uniquely identify volumes and are used by volume-oriented programs such as `volcopy`(1M).

OPTIONS

The following option is supported:

generic_options — options supported by the generic `labelit` command. See `labelit`(1M).

OPERANDS

The following operands are supported:

special — name should be the physical disk section (for example, `/dev/dsk/c0d0s6`). The device may not be on a remote machine.

fsname — represents the mount point (for example, `root`, `u1`, and so on) of the file system.

volume — may be used to represent the physical volume name.

If *fsname* and *volume* are not specified, `labelit` prints the current values of these labels. Both *fsname* and *volume* are limited to six or fewer characters.

EXIT STATUS

The following exit values are returned:

`0` — Write or display of labels was successful.

non-zero — An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`labelit`(1M), `volcopy`(1M), `fs_ufs`(4), `attributes`(5)

NAME

ldapaddent – create LDAP entries from corresponding /etc files

SYNOPSIS

ldapaddent [-cpv] [-a *authenticationMethod*] [-b *baseDN*] -D *bindDN* -w *bind_password* [-f *filename*] *database*

ldapaddent -d [-v] [-a *authenticationMethod*] [-b *baseDN*] [-D *bindDN*] [-w *bind_password*] *database*

DESCRIPTION

`ldapaddent` creates entries in LDAP containers from their corresponding `/etc` files. This operation is customized for each of the standard containers that are used in the administration of Solaris systems. The *database* argument specifies the type of the data being processed. Legal values for this type are one of `aliases`, `auto_*`, `bootparams`, `ethers`, `group`, `hosts` (including IPv6 addresses), `netgroup`, `netmasks`, `networks`, `passwd`, `shadow`, `protocols`, `publickey`, `rpc`, and `services`.

By default, `ldapaddent` reads from the standard input and adds this data to the LDAP container associated with the database specified on the command line. An input file from which data can be read is specified using the `-f` option.

The entries will be stored in the directory based on the client's configuration, thus the client must be configured to use LDAP naming services. The location where entries are to be written can be overridden by using the `-b` option.

If the entry to be added exists in the directory, the command displays an error and exits, unless the `-c` option is used.

Although, there is a `shadow` database type, there is no corresponding `shadow` container. Both the `shadow` and the `passwd` data is stored in the `people` container itself. Similarly, data from `networks` and `netmasks` databases are stored in the `networks` container.

For better performance, the recommended order in which the databases should be loaded is as follows:

You must add entries from the `passwd` database before you attempt to add entries from the `shadow` database. The addition of a `shadow` entry that does not have a corresponding `passwd` entry will fail.

For better performance, the recommended order in which the databases should be loaded is as follows:

- `passwd` database followed by `shadow` database
- `networks` database followed by `netmasks` database
- `bootparams` database followed by `ethers` database

Only the first entry of a given type that is encountered will be added to the LDAP server. The `ldapaddent` command skips any duplicate entries.

OPTIONS

The `ldapaddent` command supports the following options:

`-a` *authenticationMethod*

Specify authentication method. The default value is what has been configured in the profile. The supported authentication methods are:

```
simple
sasl/CRAM-MD5
sasl/DIGEST-MD5
tls:simple
tls:sasl/CRAM-MD5
tls:sasl/DIGEST-MD5
```

Selecting `simple` causes passwords to be sent over the network in clear text. Its use is strongly discouraged. Additionally, if the client is configured with a profile which uses no authentication, that is, either the `credentialLevel` attribute is set to `anonymous` or `authenticationMethod` is set to `none`, the user must use this option to provide an authentication method.

`-b` *baseDN*

Create entries in the *baseDN* directory. *baseDN* is not relative to the client's default search base, but rather. it is the actual location where the entries will be created. If this parameter is not specified, the first search descriptor defined for the service or the default container will be used.

`-c`

Continue adding entries to the directory even after an error. Entries will not be added if the directory server is not responding or if there is an authentication problem.

`-D` *bindDN*

Create an entry which has write permission to the *baseDN*. When used with `-d` option, this entry only needs read permission.

`-d`

Dump the LDAP container to the standard output in the appropriate format for the given database.

`-f` *filename*

Indicates input file to read in an `/etc/` file format.

`-p`

Process the `password` field when loading password information from a file. By default, the `password` field is ignored because it is usually not valid, as the actual password appears in a `shadow` file.

`-w` *bind_password*

Password to be used for authenticating the *bindDN*. If this parameter is missing, the command will prompt for a password. `NULL` passwords are not supported in LDAP.

When you use `-w` *bind_password* to specify the password to be used for authentication, the password is visible to other users of the system by means of the `ps` command, in script files or in shell history.

`-v` Verbose.

OPERANDS

The following operands are supported:

database The name of the database or service name. Supported values are: `aliases`, `auto_*`, `bootparams`, `ethers`, `group`, `hosts` (including IPv6 addresses), `netgroup`, `netmasks`, `networks`, `passwd`, `shadow`, `protocols`, `publickey`, `rpc`, and `services`.

EXAMPLES

EXAMPLE 1 Adding Password Entries to the Directory Server

The following example show how to add password entries to the directory server:

```
example# ldapaddent -D "cn=directory manager" -w secret \
      -f /etc/passwd passwd
```

EXAMPLE 2 Adding Group Entries

The following example shows how to add `group` entries to the directory server using `sasl/CRAM-MD5` as the authentication method:

```
example# ldapaddent -D "cn=directory manager" -w secret \
     -a "sasl/CRAM-MD5" -f /etc/group group
```

EXAMPLE 3 Adding `auto_master` Entries

The following example shows how to add `auto_master` entries to the directory server:

```
example# dapaddent -D "cn=directory manager" -w secret \
     -f /etc/auto_master auto_master
```

EXAMPLE 4 Dumping password Entries from the Directory to File

The following examples shows how to dump `password` entries from the directory to a file foo:

```
example# ldapaddent -d passwd > foo
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

FILES

`/var/ldap/ldap_client_file` `/var/ldap/ldap_client_cred`	Files containing the LDAP configuration of the client. These files are not to be modified manually. Their content is not guaranteed to be human readable. Use `ldapclient`(1M) to update these files.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu
Interface Stability	Evolving

SEE ALSO

`ldap`(1), `ldaplist`(1), `ldapmodify`(1), `ldapmodrdn`(1), `ldapsearch`(1), `idsconfig`(1M), `ldapclient`(1M), `suninstall`(1M), `attributes`(5)

NAME

ldap_cachemgr – LDAP daemon to manage client configuration for LDAP based Network Information Service lookups

SYNOPSIS

/usr/lib/ldap/ldap_cachemgr [-l *log-file*] [-g]

DESCRIPTION

The ldap_cachemgr daemon is a process that provides an up-to-date configuration cache for LDAP naming services. It is started during multi-user boot.

The ldap_cachemgr utility provides caching for all parameters as specified and used by the LDAP naming service clients. The ldap_cachemgr utility uses the cache files which are originally created by executing the ldapclient(1M) utility, as cold start files. Updates to the cache files take place dynamically if profiles are used to configure the client. See the init option to ldapclient(1M).

The ldap_cachemgr utility helps improve the performance of the clients that are using LDAP as the Naming service repository. In order for the LDAP naming services to function properly, the ldap_cachemgr daemon must be running. ldap_cachemgr also improves system security by making the configuration files readable by superuser only.

The cache maintained by this daemon is shared by all the processes that access LDAP Naming information. All processes access this cache through a door call. On startup, ldap_cachemgr initializes the cache from the cache files. See ldapclient(1M). Thus, the cache survives machine reboots.

The ldap_cachemgr daemon also acts as its own administration tool. If an instance of ldap_cachemgr is already running, commands are passed transparently to the running version.

OPTIONS

The following options are supported:

-g	Print current configuration and statistics to standard output. This is the only option executable without superuser privileges.
-l *log-file*	Cause ldap_cachemgr to use a log file other than the default /var/ldap/cachemgr.log.

EXAMPLES

EXAMPLE 1 Stopping and Restarting the ldap_cachemgr Daemon

The following example shows how to stop and to restart the ldap_cachemgr daemon.

```
example# /etc/init.d/ldap.client stop
example# /etc/init.d/ldap.client start
```

EXAMPLE 2 Forcing ldap_cachemgr to Reread the /var/ldap/ldap_client_file and /var/ldap/ldap_client_cred Files

The following example shows how to force ldap_cachemgr to reread the /var/ldap/ldap_client_file and /var/ldap/ldap_client_cred files

EXAMPLE 2 Forcing `ldap_cachemgr` to Reread the `/var/ldap/ldap_client_file` and `/var/ldap/ldap_client_cred` Files *(Continued)*

```
example# pkill -HUP ldap_cachemgr
```

FILES

`/var/ldap/cachemgr.log`	Default log file.
`/var/ldap/ldap_client_file` `/var/ldap/ldap_client_cred`	Files containing the LDAP configuration of the client. These files are not to be modified manually. Their content is not guaranteed to be human readable. Use `ldapclient`(1M) to update these files.

WARNINGS

The `ldap_cachemgr` utility is included in the Solaris 9 release on an uncommitted basis only. It is subject to change or removal in a future minor release.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`ldap`(1), `ldapadd`(1), `ldapdelete`(1), `ldaplist`(1), `ldapmodify`(1), `ldapmodrdn`(1), `ldapsearch`(1),`pkill`(1), `idsconfig`(1M), `ldapaddent`(1M), `ldapclient`(1M), `suninstall`(1M), `signal`(3HEAD), `resolv.conf`(4), `attributes`(5)

NAME

ldapclient – initialize LDAP client machine or output an LDAP client profile in LDIF format

SYNOPSIS

```
/usr/sbin/ldapclient [-v | -q] init [-a proxyName=profile]
     [-adomainName=domain] [-a proxyDN=proxyDN] [-a
     proxyPassword=password] [-a certificatePath=path] LDAP_server_addr
     [:port_number]

/usr/sbin/ldapclient [-v | -q] manual [-a attrName=attrVal]

/usr/sbin/ldapclient [-v | -q] mod [-a attrName=attrVal]

/usr/sbin/ldapclient [-v | -q] list

/usr/sbin/ldapclient [-v | -q] uninit

/usr/sbin/ldapclient [-v | -q] genprofile -a profileName=profileName
     [-a attrName=attrVal]
```

DESCRIPTION

The `ldapclient` utility can be used to:

- initialize LDAP client machines
- restore the network service environment on LDAP clients
- list the contents of the LDAP client cache in human readable format.

The `init` form of the `ldapclient` utility is used to initialize an LDAP client machine, using a profile stored on an LDAP server specified by `LDAP_server_addr`. The LDAP client will use the attributes in the specified profile to determine the configuration of the LDAP client. Using a configuration profile allows for easy installation of LDAP client and propagation of configuration changes to LDAP clients. The `ldap_cachemgr`(1M) utility will update the LDAP client configuration when its cache expires by reading the profile. For more information on the configuration profile refer to IETF document *A Configuration Schema for LDAP Based Directory User Agents*.

The `manual` form of the `ldapclient` utility is used to initialize an LDAP client machine manually. The LDAP client will use the attributes specified on the command line. Any unspecified attributes will be assigned their default values. The `defaultSearchBase` and `defaultServerList` attributes must be specified, and the `domainName` attribute must be specified if the client's `domainName` is not set.

The `mod` form of the `ldapclient` utility is used to modify the configuration of an LDAP client machine that was setup manually. This option modifies only those LDAP client configuration attributes specified on the command line. The `mod` option should only be used on LDAP clients that were initialized using the `manual` option.

Regardless of which method is used for initialization, if a client is to be configured to use a proxy `credentialLevel`, proxy credentials must be provided using `-a proxyDN=`*proxyD* and `-a proxyPassword=`*proxyPassword* options. However, if `-a proxyPassword=`*proxyPassword* is not specified, `ldapclient` will prompt for it. Note that `NULL` passwords are not allowed in LDAP.

If any file is modified during installation, it will be backed up to `/var/ldap/restore`. The files that are typically modified during initialization are:

- `/etc/nsswitch.conf`
- `/etc/defaultdomain` (if it exists)
- `/var/yp/binding/'domainname'` (for a NIS(YP) client)
- `/var/nis/NIS_COLD_START` (for a NIS+ client)
- `/var/ldap/ldap_client_file` (for an existing LDAP client)
- `/var/ldap/ldap_client_cred` (for an existing LDAP client)

`ldapclient` does not set up a client to resolve hostnames using DNS. It simply copies `/etc/nsswitch.ldap` to `/etc/nsswitch.conf`. If you prefer to use DNS for host resolution, please refer to the DNS documentation for information on setting up DNS. See `resolv.conf`(4).

The `list` form of the `ldapclient` utility is used to list the LDAP client configuration. The output will be human readable. LDAP configuration files are not guaranteed to be human readable.

The `uninit` form of the `ldapclient` utility is used to uninitialize the network service environment, restoring it to the state it was in prior to the last execution of `ldapclient` using `init` or `manual`. The restoration will succeed only if the machine was initialized with the `init` or `manual` form of `ldapclient`, as it uses the backup files created by these options.

The `genprofile` option is used to write an LDIF formatted configuration profile based on the attributes specified on the command line to standard output. This profile can then be loaded into an LDAP server to be used as the client profile, which can be downloaded by means of the `ldapclient init` command. Loading the LDIF formatted profile to the directory server can be done through `ldapadd`(1), or through any server specific import tool. Note that the attributes `proxyDN`, `proxyPassword`, `certificatePath`, and `domainName` are not part of the configuration profile and thus are not permitted.

You must have superuser privileges to run the `ldapclient` command, except with the `genprofile` option.

To access the information stored in the directory, clients can either authenticate to the directory, or use an unauthenticated connection. The LDAP client is configured to have a credential level of either `anonymous` or `proxy`. In the first case, the client does not authenticate to the directory. In the second case, client authenticates to the directory using a proxy identity.

If a client is configured to use an identity, you can configure which authentication method the client will use. The LDAP client supports the following authentication methods:

`none`
`simple`

```
sasl/CRAM-MD5
sasl/DIGEST-MD5
tls:simple
tls:sasl/CRAM-MD5
tls:sasl/DIGEST-MD5
```

Note that some directory servers may not support all of these authentication methods. For `simple`, be aware that the bind password will be sent in the clear to the LDAP server. For those authentication methods using TLS (transport layer security), the entire session is encrypted. You will need to install the appropriate certificate databases to use TLS.

Commands

The following commands are supported:

`init`	Initialize client from a profile on a server.
`manual`	Manually initialize client with the specified attribute values.
`mod`	Modify attribute values in the configuration file after a manual initialization of the client.
`list`	Write the contents of the LDAP client cache to standard output in human readable form.
`uninit`	Uninitialize an LDAP client, assuming that `ldapclient` was used to initialize the client.
`genprofile`	Generate a configuration profile in LDIF format that can then be stored in the directory for clients to use, with the `init` form of this command.

Attributes

The following attributes are supported:

`attributeMap`

Specify a mapping from an attribute defined by a service to an attribute in an alternative schema. This can be used to change the default schema used for a given service. The syntax of `attributeMap` is defined in the profile IETF draft. This option can be specified multiple times. The default value for all services is `NULL`. In the example,

```
attributeMap: passwd:uid=employeeNumber
```

the LDAP client would use the LDAP attribute `employeeNumber` rather than `uid` for the `passwd` service. This is a multivalued attribute.

`authenticationMethod`

Specify the default authentication method used by all services unless overridden by the `serviceAuthenticationMethod` attribute. Multiple values can be specified, using a comma-separated list.. The default value is `none`. For those services that use `credentialLevel` and `credentialLevel` is `anonymous`, this attribute is ignored. Services such as `pam_ldap` will use this attribute, even if `credentialLevel` is anonymous. The supported authentication methods are described above.

`bindTimeLimit`
The maximum time in seconds that a client should spend performing a bind operation. Set this to a positive integer. The default value is 30.

`certificatePath`
The certificate path for the location of the certificate database. The value is the path where security database files reside. This is used for TLS support, which is specified in the `authenticationMethod` and `serviceAuthenticationMethod` attributes. The default is `/var/ldap`.

`credentialLevel`
Specify the credential level the client should use to contact the directory. The credential levels supported are either `anonymous` or `proxy`. If a `proxy` credential level is specified, then the `authenticationMethod` attribute must be specified to determine the authentication mechanism. Further, if the credential level is `proxy` and at least one of the authentication methods require a bind DN, the `proxyDN` and `proxyPassword` attribute values must be set.

`defaultSearchBase`
Specify the default search base DN. There is no default. The `serviceSearchDescriptor` attribute can be used to override the `defaultSearchBase` for given services.

`defaultSearchScope=one | sub`
Specify the default search scope for the client's search operations. This default can be overridden for a given service by specifying a `serviceSearchDescriptor`. The default is one level search.

`domainName`
Specify the DNS domain name. This becomes the default domain for the machine. The default is the current domain name. This attribute is only used in client initialization.

`followReferrals=true | false`
Specify the referral setting. A setting of true implies that referrals will be automatically followed and false would result in referrals not being followed. The default is true.

`objectclassMap`
Specify a mapping from an `objectclass` defined by a service to an `objectclass` in an alternative schema. This can be used to change the default schema used for a given service. The syntax of `objectclassMap` is defined in the profile IETF draft. This option can be specified multiple times. The default value for all services is `NULL`. In the example,

```
objectclassMap=passwd:posixAccount=unixAccount
```

the LDAP client would use the LDAP `objectclass` of `unixAccount` rather than the `posixAccount` for the `passwd` service. This is a multivalued attribute.

`preferredServerList`
Specify the space separated list of preferred server IP addresses to be contacted before servers specified by the `defaultServerList` attribute. The port number is optional. If not specified, the default LDAP server port number 389 is used, except when TLS is specified in the authentication method. In this case, the default LDAP server port number is 636.

Fully qualified hostnames may also be used. If fully qualified hostnames are used, you must configure `nsswitch.conf` to use `files` or `dns`, not `ldap`, to resolve hosts lookup. If you fail to configure `nsswitch.conf` properly, then your system or certain processes can hang if you use a hostname value.

`profileName`
Specify the profile name. For `ldapclient init`, this attribute is the name of an existing profile which may be downloaded periodically depending on the value of the `profileTTL` attribute. For `ldapclient genprofile`, this is the name of the profile to be generated. The default value is `default`.

`profileTTL`
Specify the TTL value in seconds for the client information. This is only relevant if the machine was initialized with a client profile. If you do not want `ldap_cachemgr`(1M) to attempt to refresh the LDAP client configuration from the LDAP server, set `profileTTL` to 0 (zero) . Valid values are either zero 0 (for no expiration) or a positive integer in seconds. The default value is 12 hours.

`proxyDN`
Specify the Bind Distinguished Name for the proxy identity. This option is required if the credential level is `proxy`, and at least one of the authentication methods requires a bind DN. There is no default value.

`proxyPassword`
Specify client proxy password. This option is required if the credential level is `proxy`, and at least one of the authentication methods requires a bind DN. There is no default.

`searchTimeLimit`
Specify maximum number of seconds allowed for an LDAP search operation. The default is 30 seconds. The server may have its own search time limit.

`serviceAuthenticationMethod`
Specify authentication methods to be used by a service. Multiple values can be specified by a comma-separated list. The default value is no service authentication methods, in which case, each service would default to the `authenticationMethod` value. The supported authentications are described above.

Three services support this feature: `passwd-cmd`, `keyserv`, and `pam_ldap`. The `passwd-cmd` service is used to define the authentication method to be used by `passwd`(1) to change the user's password and other attributes. The `keyserv` service is used to identify the authentication method to be used by the `chkey`(1) and `newkey`(1M) utilities. The `pam_ldap` service defines the authentication

method to be used for authenticating users when `pam_ldap`(5) is configured. If this attribute is not set for any of these services, the `authenticationMethod` attribute is used to define the authentication method. This is a multivalued attribute.

`serviceCredentialLevel`
Specify credential level to be used by a service. Multiple values can be specified in a space-separated list. The default value for all services is `NULL`. The supported credential levels are: `anonymous` or `proxy`. At present, no service uses this attribute. This is a multivalued attribute.

`serviceSearchDescriptor`
Override the default base DN for LDAP searches for a given service. The format of the descriptors also allow overriding the default search scope and search filter for each service. The syntax of `serviceSearchDescriptor` is defined in the profile IETF draft. The default value for all services is `NULL`. This is a multivalued attribute. In the example,

```
serviceSearchDescriptor=passwd:ou=people,dc=a1,dc=acme,dc=com?one
```

the LDAP client would do a one level search in `ou=people,dc=a1,dc=acme,dc=com` rather than `ou=people,`*defaultSearchBase* for the `passwd` service.

OPTIONS

The following options are supported:

`-a`	Specify `attrName` and its value.
`-q`	Quiet mode. No output is generated.
`-v`	Verbose output.

OPERANDS

The following operand is supported:

defaultServerList
A space separated list of server IP addresses. The port number is optional. If not specified, the default LDAP server port number 389 is used except when TLS is specified in the authentication method, in which case the default LDAP server port number is 636.

Fully qualified hostnames may be used. If you use fully qualified hostnames, you must configure `nsswitch.conf` to use `files` or `dns`, not `ldap`, to resolve hosts lookup. If you fail to configure `nsswitch.conf` properly, then your system or certain processes can hang if you use a hostname value.

EXAMPLES

EXAMPLE 1 Setting Up a Client By Using the Default Profile Stored on a Specified LDAP Server

The following example shows how to set up a client using the default profile stored on the specified LDAP server. This command will only be successful if either the credential level in the profile is set to `anonymous` or the authentication method is set to `none`.

```
example# ldapclient init 129.100.100.1
```

EXAMPLE 2 Setting Up a Client Using Only One Server

The following example shows how to set up a client using only one server. The authentication method is set to `none`, and the search base is `dc=mycompany,dc=com`.

```
example# ldapclient manual -a authenticationMethod=none \
-a defaultSearchBase=dc=mycompany,dc=com \
-a defaultServerList=129.100.100.1
```

EXAMPLE 3 Setting Up a Client Using Only One Server That Does Not Follow Referrals

The following example shows how to set up a client using only one server. The credential level is set to `proxy`. The authentication method of is `sasl/CRAM_MD5`, with the option not to follow referrals. The domain name is `xyz.mycompany.com`, and the LDAP server is running on port number 386 at IP address `129.100.100.1`.

```
example# ldapclient manual \
-a credentialLevel=proxy \
-a authenticationMethod=sasl/CRAM_MD5 \
-a proxyPassword=secret \
-a proxyDN=cn=proxyagent,ou=profile,dc=xyz,dc=mycompany,dc=com \
-a defaultSearchBase=dc=xyz,dc=mycompany,dc=com \
-a domainName=xyz.mycompany.com \
-a followReferrals=false \
-a defaultServerList=129.100.100.1:386
```

EXAMPLE 4 Using `genprofile` to Set Only the `defaultSearchBase` and the Server Addresses

The following example shows how to use the `genprofile` command to set the `defaultSearchBase` and the server addresses.

```
example# ldapclient genprofile -a profileName=myprofile \
-a defaultSearchBase=dc=eng,dc=sun,dc=com \
-a "defaultServerList=129.100.100.1 129.100.234.15:386" \
> myprofile.ldif
```

EXAMPLE 5 Creating a Profile That Overrides Every Default Value

The following example shows a profile that overrides every default value.

EXAMPLE 5 Creating a Profile That Overrides Every Default Value *(Continued)*

```
example# ldapclient genprofile -a profileName=eng \
-a credentialLevel=proxy -a authenticationMethod=sasl/DIGEST-MD5 \
-a bindTimeLimit=20 \
-a defaultSearchBase=dc=eng,dc=acme,dc=com \
-a "serviceSearchDescriptor=passwd:ou=people,dc=a1,dc=acme,dc=com?one" \
-a serviceAuthenticationMethod=pam_ldap:tls:simple \
-a defaultSearchScope=sub \
-a attributeMap=passwd:uid=employeeNumber \
-a objectclassMap=passwd:posixAccount=unixAccount \
-a followReferrals=false -a profileTTL=6000 \
-a preferredServerList=129.100.100.30 -a searchTimeLimit=30 \
-a "defaultServerList=29.100.200.1 129.100.100.1 204.34.5.6" > eng.ldif
```

EXIT STATUS

The following exit values are returned:

0 The command successfully executed.

1 An error occurred. An error message is output.

2 `proxyDN` and `proxyPassword` attributes are required, but they are not provided.

FILES

`/var/ldap/ldap_client_cred`
`/var/ldap/ldap_client_file` — Contain the LDAP configuration of the client. These files are not to be modified manually. Their content is not guaranteed to be human readable. Use `ldapclient` to update them.

`/etc/defaultdomain` — System default domain name, matching the domain name of the data in the LDAP servers.

`/etc/nsswitch.conf` — Configuration file for the name-service switch.

`/etc/nsswitch.ldap` — Sample configuration file for the name-service switch configured with ldap and files.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu
Interface Stability	Evolving

SEE ALSO | `chkey`(1), `ldap`(1), `ldapadd`(1), `ldapdelete`(1), `ldaplist`(1), `ldapmodify`(1), `ldapmodrdn`(1), `ldapsearch`(1), `idsconfig`(1M), `ldapaddent`(1M), `ldap_cachemgr`(1M), `suninstall`(1M), `resolv.conf`(4), `attributes`(5)

NAME

link, unlink – link and unlink files and directories

SYNOPSIS

/usr/sbin/link *existing-file new-file*

/usr/xpg4/bin/link *existing-file new-file*

/usr/sbin/unlink *file*

DESCRIPTION

The `link` and `unlink` commands link and unlink files and directories. Only super-users can use these commands.

Use `link` to create a new file that points to an existing file. The *existing-file* and *new-file* operands specify the existing file and newly-created files. See `OPERANDS`.

`link` and `unlink` directly invoke the `link`(2) and `unlink`(2) system calls, performing exactly what they are told to do and abandoning all error checking. This differs from the `ln`(1) command. See `ln`(1).

While linked files and directories can be removed using `unlink`, it is safer to use `rm`(1) and `rmdir`(1) instead. See `rm`(1) and `rmdir`(1).

/usr/xpg4/bin/link

If the existing file being hard linked is itself a symbolic link, then the newly created file (*new-file*) will be a hard link to the file referenced by the symbolic link, not to the symbolic link object itself (*existing-file*).

OPERANDS

existing-file Specifies the name of the existing file to be linked.

file Specifies the name of the file to be unlinked.

new-file Specifies the name of newly created (linked) file.

ENVIRONMENT VARIABLES

See `environ`(5) for descriptions of the following environment variables that affect the execution of `link`: `LANG`, `LC_ALL`, `LC_CTYPE`, `LC_MESSAGES`, and `NLSPATH`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

/usr/xpg4/bin/link

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWxcu4

SEE ALSO

`ln`(1), `rm`(1), `link`(2), `unlink`(2), `attributes`(5), `environ`(5)

NAME

listdgrp – lists members of a device group

SYNOPSIS

/usr/bin/listdgrp *dgroup...*

DESCRIPTION

`listdgrp` displays the members of the device groups specified by the *dgroup* list.

EXAMPLES

EXAMPLE 1 An example of `listdgrp`.

The following example lists the devices that belong to group `partitions`:

```
example% listdgrp partitions
     root
     swap
     usr
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` Command was syntax incorrect, an invalid option used, or an internal error occurred.

`2` A device group table could not be opened for reading.

`3` A device group *dgroup* could not be found in the device group table.

FILES

`/etc/dgroup.tab`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`putdgrp`(1M), `attributes`(5)

NAME

listen – network listener daemon

SYNOPSIS

/usr/lib/saf/listen [-m *devstem*] *net_spec*

DESCRIPTION

The `listen` process ''listens'' to a network for service requests, accepts requests when they arrive, and invokes servers in response to those service requests. The network listener process may be used with any connection-oriented network (more precisely, with any connection-oriented transport provider) that conforms to the Transport Layer Interface (TLI) Specification.

The listener internally generates a pathname for the minor device for each connection; it is this pathname that is used in the `utmpx` entry for a service, if one is created. By default, this pathname is the concatenation of the prefix `/dev/`*netspec* with the decimal representation of the minor device number. In either case, the representation of the minor device number will be at least two digits (for example, 05 or 27), or longer when it is necessary to accommodate minor device numbers larger than 99.

SERVER INVOCATION

When a connection indication is received, the listener creates a new transport endpoint and accepts the connection on that endpoint. Before giving the file descriptor for this new connection to the server, any designated STREAMS modules are pushed and the configuration script is executed, (if one exists). This file descriptor is appropriate for use with either TLI (see `t_sync`(3NSL)) or the sockets interface library.

By default, a new instance of the server is invoked for each connection. When the server is invoked, file descriptor 0 refers to the transport endpoint, and is open for reading and writing. File descriptors 1 and 2 are copies of file descriptor 0; no other file descriptors are open. The service is invoked with the user and group IDs of the user name under which the service was registered with the listener, and with the current directory set to the HOME directory of that user.

Alternatively, a service may be registered so that the listener will pass connections to a standing server process through a FIFO or a named STREAM, instead of invoking the server anew for each connection. In this case, the connection is passed in the form of a file descriptor that refers to the new transport endpoint. Before the file descriptor is sent to the server, the listener interprets any configuration script registered for that service using `doconfig`(3NSL), although `doconfig` is invoked with both the `NORUN` and `NOASSIGN` flags. The server receives the file descriptor for the connection in a `strrecvfd` structure using an `I_RECVFD` `ioctl`(2).

For more details about the listener and its administration, see `nlsadmin`(1M).

OPTIONS

-m*devstem* The listener will use *devstem* as the prefix for the pathname.

FILES

`/etc/saf/`*pmtag*`/*`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`nlsadmin`(1M), `pmadm`(1M), `sac`(1M), `sacadm`(1M), `ioctl`(2), `doconfig`(3NSL), `nlsgetcall`(3NSL), `nlsprovider`(3NSL), `t_sync`(3NSL), `attributes`(5), `streamio`(7I)

System Administration Guide: Basic Administration

NOTES

When passing a connection to a standing server, the user and group IDs contained in the `strrecvfd` structure will be those for the listener (that is, they will both be 0); the user name under which the service was registered with the listener is not reflected in these IDs.

When operating multiple instances of the listener on a single transport provider, there is a potential race condition in the binding of addresses during initialization of the listeners, if any of their services have dynamically assigned addresses. This condition would appear as an inability of the listener to bind a static-address service to its otherwise valid address, and would result from a dynamic-address service having been bound to that address by a different instance of the listener.

NAME

llc2_loop – loopback diagnostics to test the driver, adapter and network.

SYNOPSIS

/usr/lib/llc2/llc2_loop2 [-v] *ppa*

/usr/lib/llc2/llc2_loop3 *ppa sap frames*

/usr/lib/llc2/llc2_loop3 *ppa type frames*

/usr/lib/llc2/llc2_loop4 [-v] *ppa*

Loop 2

The `loop2` test sends a NULL XID frame to the broadcast (all 1's) destination MAC address. The source SAP (Service Access Point) value used is 0x04 (SNA's SAP). Therefore, if SNA is running on the system, the `loop2` test will fail. The destination SAP value is the NULL SAP (0x00). This test finds out who is listening and can receive frames sent out from a node. The verbose (`-v`) option displays the MAC address of responding nodes. All possible responders may not be displayed, since the `loop2` test only waits for responses for 2 seconds, but during this time 50-200 nodes may be displayed. The most likely error is:

```
    Unexpected DLPI primitive x, expected y.
```

where $x = 5$ and $y = 6$. From `/usr/include/sys/dlpi.h`, the expected return value from one of the DLPI primitives is 6 (`DL_OK_ACK`), but instead a 5 (`DL_ERROR_ACK`) was received. This can occur for two reasons:

- The `loop2` command was issued to a non-existent PPA (Physical Point of Attachment).
- The SAP (0x04) is already in use (for example, the SNA subsystem is up).

Loop 3

The `loop3` test sends 1,495 byte Unnumbered Information (UI) frames to the NULL (all 0's) destination MAC address. This should be used along with data capture either on the local node or another node on the same LAN to verify the transmission of data. The *ppa* argument specifies the adapter on which to run the test. The *ppa* is the relative physical position of the adapter and may be ascertained by viewing the adapter configuration (see `llc2_config`(1)). For Token Ring or Ethernet, specify an even *sap* value from 2 through 254, or, for Ethernet only, any *type* value from 1519 (0x05ef) through 65535 (0xffff). It is advised to pick a value that is easily recognized when the data capture output is viewed. *frames* is the decimal number of 1,495 bytes packets to transmit. The test will only display a message if a failure occurs.

Loop 4

The `loop4` test sends a TEST frame (no information field) to the broadcast (all 1's) destination MAC address. The source SAP value used is 0x04 (SNA's SAP). Therefore, if SNA is running on the system, the `loop4` test will fail. The destination SAP value is the NULL SAP (0x00). This test finds out who is listening and can receive frames sent out from a node. The verbose (`-v`) option displays the MAC address of responding nodes. All possible responders may not be displayed since the `loop4` test only waits for responses for 2 seconds, but during this time 50-200 nodes may be displayed. The `loop4` test displays information similar to the following example if other nodes are listening and respond (verbose mode):

```
-Attaching
    -Binding
    -Sending TEST
```

```
    -Responders
       1-0000c0c12449
       2-08000e142990
       3-08000e142a51
       4-0000c0450044
       5-0000c0199e46
    -Unbinding
-Detaching
5 nodes responding
```

The errors displayed are the same as for `loop2`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWllc

SEE ALSO

`llc2_config`(1), `llc2`(4), `attributes`(5), `llc2`(7D)

NOTES

For information about how to start the service, see `llc2`(7D)

NAME

locator – location indicator control

SYNOPSIS

```
/usr/sbin/locator [-f | -n]
```

DESCRIPTION

The locator command sets or queries the state of the system locator if such a device exists.

Without options, the locator command reports the current state of the system.

The privileges required to use this command are hardware dependent. Typically, only the super user can get or set a locator.

OPTIONS

The following options are supported:

`-f` Turns the locator off.

`-n` Turns the locator on.

OPERANDS

The following operands are supported:

EXAMPLES

EXAMPLE 1 Using the `locator` Command on a Platform Which Has a System Locator LED

When issued on a platform which has a system locator LED, the following command turns the locator on:

```
# locator -n
# locator
The 'system' locator is on
```

EXAMPLE 2 Using the `locator` Command on a Platform Which Does Not Have a System Locator LED

When issued on a platform which does not have a system locator LED, the following command attempts to turn the locator on. The command returns an error message.

```
# locator -n
'system' locator not found
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` Invalid command line input.

`2` The requested operation failed.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO | `attributes`(5)

NAME

lockd – network lock daemon

SYNOPSIS

/usr/lib/nfs/lockd [-g *graceperiod*] [-t *timeout*] [*nthreads*]

DESCRIPTION

The `lockd` utility is part of the NFS lock manager, which supports record locking operations on NFS files. See `fcntl`(2) and `lockf`(3C). The lock manager provides two functions:

- it forwards `fcntl`(2) locking requests for NFS mounted file systems to the lock manager on the NFS server
- it generates local file locking operations in response to requests forwarded from lock managers running on NFS client machines.

State information kept by the lock manager about these locking requests can be lost if the `lockd` is killed or the operating system is rebooted. Some of this information can be recovered as follows. When the server lock manager restarts, it waits for a grace period for all client-site lock managers to submit reclaim requests. Client-site lock managers, on the other hand, are notified by the status monitor daemon, `statd`(1M), of the restart and promptly resubmit previously granted lock requests. If the lock daemon fails to secure a previously granted lock at the server site, then it sends `SIGLOST` to a process.

Administrators wanting to change startup parameters for `lockd` should, as root, make changes in the `/etc/default/nfs` file rather than editing the `/etc/init.d/nfs.client` file. See `nfs`(4).

OPTIONS

`-g` *graceperiod*	Specify the number of seconds that clients have to reclaim locks after the server reboots. The default is 45 seconds. Equivalent of the `LOCKD_GRACE_PERIOD` parameter in the `nfs` file.
`-t` *timeout*	Specify the number of seconds to wait before retransmitting a lock request to the remote server. The default value is 15 seconds. Equivalent of the `LOCKD_RETRANSMIT_TIMEOUT` parameter in the `nfs` file.
nthreads	Specify the maximum number of concurrent threads that the server can handle. This concurrency is achieved by up to *nthreads* threads created as needed in the kernel. *nthreads* should be based on the load expected on this server. If *nthreads* is not specified, the maximum number of concurrent threads will default to 20. Equivalent of the `LOCKD_SERVERS` parameter in the `nfs` file.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnfscu

SEE ALSO

`statd`(1M), `fcntl`(2), `lockf`(3C), `nfs`(4), `attributes`(5)

NAME

lockfs – change or report file system locks

SYNOPSIS

/usr/sbin/lockfs [-adefhnuw] [-c *string*] [*file-system*...]

DESCRIPTION

`lockfs` is used to change and report the status of file system locks. `lockfs` reports the lock status and unlocks the file systems that were improperly left locked.

Using `lockfs` to lock a file system is discouraged because this requires extensive knowledge of SunOS internals to be used effectively and correctly.

When invoked with no arguments, `lockfs` lists the UFS file systems that are locked. If *file-system* is not specified, and `-a` is specified, `lockfs` is run on all mounted, UFS type file systems.

The following options (described below) are mutually exclusive: `wndheuf`. If you do specify more than one of these options on a `lockfs` command line, the utility does not protest and invokes only the last option specified. In particular, you cannot specify a flush (`-f`) and a lock (for example, `-w`) on the same command line. However, all locking operations implicitly perform a flush, so the `-f` is superfluous when specifying a lock.

OPTIONS

The following options are supported. You must be super-user to use any of the following options, with the exception of `-a`.

`-a` Apply command to all mounted, UFS type file systems. *file-system* is ignored when `-a` is specified.

`-c` *string* Accept a string that is passed as the comment field. The `-c` only takes affect when the lock is being set using the `-d`, `-h`, `-n`, `-u`, or `-w` options.

`-d` delete-lock (dlock) the specified *file-system*. dlock suspends access that could remove directory entries.

`-e` error-lock (elock) the specified *file-system*. elock blocks all local access to the locked file system and returns `EWOULDBLOCK` on all remote access. File systems are elocked by UFS on detection of internal inconsistency. They may only be unlocked after successful repair by `fsck`, which is usually done automatically (see `mount_ufs`(1M)). elocked file systems can be unmounted.

`-f` Flush all transactions out of the log and write the transactions to the master file system. This option is valid only if logging has been enabled on the file system.

`-h` Hard-lock (hlock) the specified *file-system*. hlock returns an error on every access to the locked file system, and cannot be unlocked. hlocked file systems can be unmounted.

`-n` Name-lock (nlock) the specified *file-system*. nlock suspends accesses that could change or remove existing directories entries.

`-u`	Unlock (ulock) the specified *file-system*. ulock awakens suspended accesses.
`-w`	Write-lock (wlock) the specified *file-system*. wlock suspends writes that would modify the file system. Access times are not kept while a file system is write-locked.

OPERANDS

The following operands are supported.

file-system	A list of path names separated by white spaces.

USAGE

See `largefile`(5) for the description of the behavior of `lockfs` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXAMPLES

EXAMPLE 1 `lockfs` with the `-a` option.

In the following examples, *filesystem* is the pathname of the mounted-on directory (mount point). `Locktype` is one of "`write`," "*name*," "*delete*," "*hard*," or "*unlock*". When enclosed in parenthesis, the lock is being set. `Comment` is a string set by the process that last issued a lock command.

The following example shows the `lockfs` output when only the `-a` option is specified.

```
example#  /usr/sbin/lockfs -a
```

Filesystem	Locktype	Comment
/	unlock	
/var	unlock	

example#

EXAMPLE 2 `lockfs` with the `-w` option.

The following example shows the `lockfs` output when the `-w` option is used to write lock the `/var` file system and the comment string is set using the `-c` option. The `-a` option is then specified on a separate command line.

```
example#  /usr/sbin/lockfs -w -c "lockfs: write lock example" /var
example#  /usr/sbin/lockfs -a
```

Filesystem	Locktype	Comment
/	unlock	
/var	write	lockfs: write lock example

example#

EXAMPLE 3 `lockfs` with the `-u` option.

The following example shows the `lockfs` output when the `-u` option is used to unlock the `/var` file system and the comment string is set using the `-c` option.

```
example#  /usr/sbin/lockfs -uc "lockfs: unlock example" /var
example#  /usr/sbin/lockfs /var
```

Filesystem	Locktype	Comment
/var	unlock	lockfs: unlock example

example#

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`kill`(1), `mount_ufs`(1M), `fs_ufs`(4), `attributes`(5), `largefile`(5)

System Administration Guide: Basic Administration

DIAGNOSTICS

file system`: Not owner`
You must be root to use this command.

file system `:Deadlock condition detected/avoided`
A file is enabled for accounting or swapping, on *file system*.

file system`: Device busy`
Another process is setting the lock on *file system*.

NAME

lockstat – report kernel lock and profiling statistics

SYNOPSIS

```
lockstat [-ACEHI] [-e event_list] [-i rate] [-b | -t | -h | -s depth]
     [-n nrecords] [-l lock [, size]] [-d duration] [-f function [, size]]
     [-T] [-ckgwWRpP] [-D count] [-o filename] command [args]
```

DESCRIPTION

The `lockstat` utility gathers and displays kernel locking and profiling statistics. `lockstat` allows you to specify which events to watch (for example, "spin on adaptive mutex," "block on read access to rwlock due to waiting writers," and so forth) how much data to gather for each event, and how to display the data. By default, `lockstat` monitors all lock contention events, gathers frequency and timing data about those events, and displays the data in decreasing frequency order, so that the most common events appear first.

`lockstat` gathers data until the specified command completes. For example, to gather statistics for a fixed-time interval, use `sleep`(1) as the command, as follows:

```
example# lockstat sleep 5
```

When the `-I` option is specified, `lockstat` establishes a per-processor high-level periodic interrupt source to gather profiling data. The interrupt handler simply generates a `lockstat` event whose "caller" is the interrupted PC (program counter). The profiling event is just like any other `lockstat` event, so all of the normal `lockstat` options are applicable.

`lockstat` relies on the `lockstat`(7D) driver, an exclusive-access device that modifies the running kernel's text to intercept events of interest. This imposes a small but measurable overhead on all system activity, so access to the `lockstat`(7D) driver is restricted to super-user by default. The system administrator may relax this restriction by changing the permissions on `/dev/lockstat`.

OPTIONS

The following options are supported:

Event Selection

If no event selection options are specified, the default is `-CE`.

`-A`	Watch all lock events. `-A` is equivalent to `-CEH`.
`-C`	Watch contention events.
`-E`	Watch error events.
`-e` *event_list*	Only watch the specified events. *event list* is a comma-separated list of events or ranges of events such as 1,4-7,35. Run `lockstat` with no arguments to get a brief description of all events.
`-H`	Watch hold events.
`-I`	Watch profiling interrupt events.
`-i` *rate*	Interrupt rate (per second) for `-I`. The default is 97 Hz, so that profiling doesn't run in lockstep with the clock interrupt (which runs at 100 Hz).

Data Gathering (Mutually Exclusive)

`-b`	Basic statistics: lock, caller, number of events.
`-h`	Histogram: Timing plus time-distribution histograms.
`-s` *depth*	Stack trace: Histogram plus stack traces up to *depth* frames deep.
`-t`	Timing: Basic plus timing for all events [default].

Data Filtering

`-d` *duration*	Only watch events longer than *duration*.
`-f` *func[,size]*	Only watch events generated by *func*, which can be specified as a symbolic name or hex address. *size* defaults to the ELF symbol size if available, or `1` if not.
`-l` *lock[,size]*	Only watch *lock*, which can be specified as a symbolic name or hex address. `size` defaults to the ELF symbol size or `1` if the symbol size is not available.
`-n` *nrecords*	Maximum number of data records.
`-T`	Trace (rather than sample) events [off by default].

Data Reporting

`-c`	Coalesce lock data for lock arrays (for example, `pse_mutex[]`).
`-D` *count*	Only display the top *count* events of each type.
`-g`	Show total events generated by function. For example, if `foo()` calls `bar()` in a loop, the work done by `bar()` counts as work generated by `foo()` (along with any work done by `foo()` itself). The `-g` option works by counting the total number of stack frames in which each function appears. This implies two things: (1) the data reported by `-g` can be misleading if the stack traces are not deep enough, and (2) functions that are called recursively might show greater than 100% activity. In light of issue (1), the default data gathering mode when using `-g` is `-s 50`.
`-k`	Coalesce PCs within functions.
`-o` *filename*	Direct output to *filename*.
`-P`	Sort data by (*count* * *time*) product.
`-p`	Parsable output format.
`-R`	Display rates (events per second) rather than counts.
`-W`	Whichever: distinguish events only by caller, not by lock.
`-w`	Wherever: distinguish events only by lock, not by caller.

DISPLAY FORMATS

The following headers appear over various columns of data.

`Count` or `ops/s`	Number of times this event occurred, or the rate (times per second) if `-R` was specified.
`indv`	Percentage of all events represented by this individual event.

`genr`	Percentage of all events generated by this function.
`cuml`	Cumulative percentage; a running total of the individuals.
`rcnt`	Average reference count. This will always be `1` for exclusive locks (mutexes, spin locks, rwlocks held as writer) but may be greater than `1` for shared locks (rwlocks held as reader).
`spin` or `nsec`	Average number of times caller spun trying to get the lock, or average duration of the events in nanoseconds, as appropriate for the event. For the profiling event, "duration" means interrupt latency.
`Lock`	Address of the lock; displayed symbolically if possible.
`CPU+PIL`	CPU plus processor interrupt level (PIL). For example, if CPU 4 is interrupted while at PIL 6, this will be reported as `cpu[4]+6`.
`Caller`	Address of the caller; displayed symbolically if possible.

EXAMPLES

EXAMPLE 1 Measuring Kernel Lock Contention

```
example# lockstat sleep 5
Adaptive mutex spin: 2210 events in 5.055 seconds (437 events/sec)

Count indv cuml rcnt     spin Lock                Caller
-------------------------------------------------------------------------
  269  12%  12% 1.00       10 service_queue       background+0xdc
  249  11%  23% 1.00        8 service_queue       qenable_locked+0x64
  228  10%  34% 1.00       13 service_queue       background+0x15c
   68   3%  37% 1.00        7 0x30000024070       untimeout+0x1c
   59   3%  40% 1.00       38 0x300066fa8e0       background+0xb0
   43   2%  41% 1.00        3 rqcred_lock         svc_getreq+0x3c
   42   2%  43% 1.00       34 0x30006834eb8       background+0xb0
   41   2%  45% 1.00       13 0x30000021058       untimeout+0x1c
   40   2%  47% 1.00        3 rqcred_lock         svc_getreq+0x260
   37   2%  49% 1.00      237 0x300068e83d0       hmestart+0x1c4
   36   2%  50% 1.00        7 0x30000021058       timeout_common+0x4
   36   2%  52% 1.00       35 0x300066fa120       background+0xb0
   32   1%  53% 1.00        9 0x30000024070       timeout_common+0x4
   31   1%  55% 1.00      292 0x300069883d0       hmestart+0x1c4
   29   1%  56% 1.00       36 0x300066fb290       background+0xb0
   28   1%  57% 1.00       11 0x3000001e040       untimeout+0x1c
   25   1%  59% 1.00        9 0x3000001e040       timeout_common+0x4
   22   1%  60% 1.00        2 0x30005161110       sync_stream_buf+0xdc
   21   1%  60% 1.00       29 0x30006834eb8       putq+0xa4
   19   1%  61% 1.00        4 0x3000515dcb0       mdf_alloc+0xc
   18   1%  62% 1.00       45 0x30006834eb8       qenable+0x8
   18   1%  63% 1.00        6 service_queue       queuerun+0x168
   17   1%  64% 1.00       26 0x30005418ee8       vmem_free+0x3c
[...]

R/W reader blocked by writer: 76 events in 5.055 seconds (15 events/sec)

Count indv cuml rcnt     nsec Lock                Caller
-------------------------------------------------------------------------
   23  30%  30% 1.00 22590137 0x300098ba358       ufs_dirlook+0xd0
```

EXAMPLE 1 Measuring Kernel Lock Contention *(Continued)*

```
   17  22%  53% 1.00  5820995 0x3000ad815e8       find_bp+0x10
   13  17%  70% 1.00  2639918 0x300098ba360       ufs_iget+0x198
    4   5%  75% 1.00  3193015 0x300098ba360       ufs_getattr+0x54
    3   4%  79% 1.00  7953418 0x3000ad817c0       find_bp+0x10
    3   4%  83% 1.00   935211 0x3000ad815e8       find_read_lof+0x14
    2   3%  86% 1.00 16357310 0x300073a4720       find_bp+0x10
    2   3%  88% 1.00  2072433 0x300073a4720       find_read_lof+0x14
    2   3%  91% 1.00  1606153 0x300073a4370       find_bp+0x10
    1   1%  92% 1.00  2656909 0x300107e7400       ufs_iget+0x198
[...]
```

EXAMPLE 2 Measuring Hold Times

```
example# lockstat -H -D 10 sleep 1
Adaptive mutex spin: 513 events

Count indv cumul rcnt     nsec Lock                 Caller
-------------------------------------------------------------------------------
  480   5%   5% 1.00     1136 0x300007718e8       putnext+0x40
  286   3%   9% 1.00      666 0x3000077b430       getf+0xd8
  271   3%  12% 1.00      537 0x3000077b430       msgio32+0x2fc
  270   3%  15% 1.00     3670 0x300007718e8       strgetmsg+0x3d4
  270   3%  18% 1.00     1016 0x300007c38b0       getq_noenab+0x200
  264   3%  20% 1.00     1649 0x300007718e8       strgetmsg+0xa70
  216   2%  23% 1.00     6251 tcp_mi_lock         tcp_snmp_get+0xfc
  206   2%  25% 1.00      602 thread_free_lock    clock+0x250
  138   2%  27% 1.00      485 0x300007c3998       putnext+0xb8
  138   2%  28% 1.00     3706 0x300007718e8       strrput+0x5b8
-------------------------------------------------------------------------------
[...]
```

EXAMPLE 3 Measuring Hold Times for Stack Traces Containing a Specific Function

```
example# lockstat -H -f tcp_rput_data -s 50 -D 10 sleep 1
Adaptive mutex spin: 11 events in 1.023 seconds (11
events/sec)

-------------------------------------------------------------------------------
Count indv cumul rcnt     nsec Lock                   Caller
    9  82%  82% 1.00     2540 0x30000031380          tcp_rput_data+0x2b90

      nsec ------ Time Distribution ------ count     Stack
       256 |@@@@@@@@@@@@@@@@@              5         tcp_rput_data+0x2b90
       512 |@@@@@@                         2         putnext+0x78
      1024 |@@@                            1         ip_rput+0xec4
      2048 |                               0         _c_putnext+0x148
      4096 |                               0         hmeread+0x31c
      8192 |                               0         hmeintr+0x36c
     16384 |@@@                            1
sbus_intr_wrapper+0x30
-------------------------------------------------------------------------------
Count indv cumul rcnt     nsec Lock                   Caller
    1   9%  91% 1.00     1036 0x30000055380          freemsg+0x44
```

EXAMPLE 3 Measuring Hold Times for Stack Traces Containing a Specific Function *(Continued)*

```
      nsec ------ Time Distribution ------ count     Stack
      1024 |@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ 1         freemsg+0x44
                                                     tcp_rput_data+0x2fd0
                                                     putnext+0x78
                                                     ip_rput+0xec4
                                                     _c_putnext+0x148
                                                     hmeread+0x31c
                                                     hmeintr+0x36c

sbus_intr_wrapper+0x30
-------------------------------------------------------------------------------
[...]
```

EXAMPLE 4 Basic Kernel Profiling

For basic profiling, we don't care whether the profiling interrupt sampled `foo()+0x4c` or `foo()+0x78`; we care only that it sampled somewhere in `foo()`, so we use `-k`. The CPU and PIL aren't relevant to basic profiling because we are measuring the system as a whole, not a particular CPU or interrupt level, so we use `-W`.

```
example# lockstat -kIW -D 20 ./polltest
Profiling interrupt: 82 events in 0.424 seconds (194
events/sec)

Count indv cumul rcnt     nsec Hottest CPU+PIL     Caller
-------------------------------------------------------------------------
    8  10%  10% 1.00      698 cpu[1]              utl0
    6   7%  17% 1.00      299 cpu[0]              read
    5   6%  23% 1.00      124 cpu[1]              getf
    4   5%  28% 1.00      327 cpu[0]              fifo_read
    4   5%  33% 1.00      112 cpu[1]              poll
    4   5%  38% 1.00      212 cpu[1]              uiomove
    4   5%  43% 1.00      361 cpu[1]              mutex_tryenter
    3   4%  46% 1.00      682 cpu[0]              write
    3   4%  50% 1.00       89 cpu[0]              pcache_poll
    3   4%  54% 1.00      118 cpu[1]              set_active_fd
    3   4%  57% 1.00      105 cpu[0]              syscall_trap32
    3   4%  61% 1.00      640 cpu[1]              (usermode)
    2   2%  63% 1.00      127 cpu[1]              fifo_poll
    2   2%  66% 1.00      300 cpu[1]              fifo_write
    2   2%  68% 1.00      669 cpu[0]              releasef
    2   2%  71% 1.00      112 cpu[1]              bt_getlowbit
    2   2%  73% 1.00      247 cpu[1]              splx
    2   2%  76% 1.00      503 cpu[0]              mutex_enter
    2   2%  78% 1.00      467 cpu[0]+10           disp_lock_enter
    2   2%  80% 1.00      139 cpu[1]              default_copyin
-------------------------------------------------------------------------
```

EXAMPLE 5 Generated-load Profiling

In the example above, 5% of the samples were in `poll()`. This tells us how much time was spent inside `poll()` itself, but tells us nothing about how much work was *generated* by `poll()`; that is, how much time we spent in functions called by `poll()`. To determine that, we use the `-g` option. The example below shows that although `polltest` spends only 5% of its time in `poll()` itself, `poll()`-induced work accounts for 34% of the load.

Note that the functions that generate the profiling interrupt (`lockstat_intr()`, `cyclic_fire()`, and so forth) appear in every stack trace, and therefore are considered to have generated 100% of the load. This illustrates an important point: the generated load percentages do *not* add up to 100% because they are not independent. If 72% of all stack traces contain both `foo()` and `bar()`, then both `foo()` and `bar()` are 72% load generators.

```
example# lockstat -kgIW -D 20 ./polltest
Profiling interrupt: 80 events in 0.412 seconds (194 events/sec)

Count genr cuml rcnt     nsec Hottest CPU+PIL     Caller
-------------------------------------------------------------------------
   80 100% ---- 1.00      310 cpu[1]             lockstat_intr
   80 100% ---- 1.00      310 cpu[1]             cyclic_fire
   80 100% ---- 1.00      310 cpu[1]             cbe_level14
   80 100% ---- 1.00      310 cpu[1]             current_thread
   27  34% ---- 1.00      176 cpu[1]             poll
   20  25% ---- 1.00      221 cpu[0]             write
   19  24% ---- 1.00      249 cpu[1]             read
   17  21% ---- 1.00      232 cpu[0]             write32
   17  21% ---- 1.00      207 cpu[1]             pcache_poll
   14  18% ---- 1.00      319 cpu[0]             fifo_write
   13  16% ---- 1.00      214 cpu[1]             read32
   10  12% ---- 1.00      208 cpu[1]             fifo_read
   10  12% ---- 1.00      787 cpu[1]             utl0
    9  11% ---- 1.00      178 cpu[0]             pcacheset_resolve
    9  11% ---- 1.00      262 cpu[0]             uiomove
    7   9% ---- 1.00      506 cpu[1]             (usermode)
    5   6% ---- 1.00      195 cpu[1]             fifo_poll
    5   6% ---- 1.00      136 cpu[1]             syscall_trap32
    4   5% ---- 1.00      139 cpu[0]             releasef
    3   4% ---- 1.00      277 cpu[1]             polllock
-------------------------------------------------------------------------

  .  .  .
```

EXAMPLE 6 Gathering Lock Contention and Profiling Data for a Specific Module

In this example we use the `-f` option not to specify a single function, but rather to specify the entire text space of the `sbus` module. We gather both lock contention and profiling statistics so that contention can be correlated with overall load on the module.

```
example# modinfo | grep sbus
 24 102a8b6f   b8b4  59   1  sbus (SBus (sysio) nexus driver)
```

EXAMPLE 6 Gathering Lock Contention and Profiling Data for a Specific Module *(Continued)*

```
example# lockstat -kICE -f 0x102a8b6f,0xb8b4 sleep 10
Adaptive mutex spin: 39 events in 10.042 seconds (4 events/sec)

Count indv cuml rcnt     spin Lock               Caller
-------------------------------------------------------------------------
   15  38%  38% 1.00        2 0x30005160528      sync_stream_buf
    7  18%  56% 1.00        1 0x30005160d18      sync_stream_buf
    6  15%  72% 1.00        2 0x300060c3118      sync_stream_buf
    5  13%  85% 1.00        2 0x300060c3510      sync_stream_buf
    2   5%  90% 1.00        2 0x300060c2d20      sync_stream_buf
    2   5%  95% 1.00        2 0x30005161cf8      sync_stream_buf
    1   3%  97% 1.00        2 0x30005161110      sync_stream_buf
    1   3% 100% 1.00        2 0x30005160130      sync_stream_buf
-------------------------------------------------------------------------

Adaptive mutex block: 9 events in 10.042 seconds (1 events/sec)

Count indv cuml rcnt     nsec Lock               Caller
-------------------------------------------------------------------------
    4  44%  44% 1.00   156539 0x30005160528      sync_stream_buf
    2  22%  67% 1.00   763516 0x30005160d18      sync_stream_buf
    1  11%  78% 1.00   462130 0x300060c3510      sync_stream_buf
    1  11%  89% 1.00   288749 0x30005161110      sync_stream_buf
    1  11% 100% 1.00  1015374 0x30005160130      sync_stream_buf
-------------------------------------------------------------------------

Profiling interrupt: 229 events in 10.042 seconds (23 events/sec)

Count indv cuml rcnt     nsec Hottest CPU+PIL    Caller

-------------------------------------------------------------------------
   89  39%  39% 1.00      426 cpu[0]+6           sync_stream_buf
   64  28%  67% 1.00      398 cpu[0]+6           sbus_intr_wrapper
   23  10%  77% 1.00      324 cpu[0]+6           iommu_dvma_kaddr_load
   21   9%  86% 1.00      512 cpu[0]+6           iommu_tlb_flush
   14   6%  92% 1.00      342 cpu[0]+6           iommu_dvma_unload
   13   6%  98% 1.00      306 cpu[1]             iommu_dvma_sync
    5   2% 100% 1.00      389 cpu[1]             iommu_dma_bindhdl
-------------------------------------------------------------------------
```

EXAMPLE 7 Determining the Average PIL (processor interrupt level) for a CPU

```
example# lockstat -Iw -l cpu[3] ./testprog

Profiling interrupt: 14791 events in 152.463 seconds (97 events/sec)

Count indv cuml rcnt     nsec CPU+PIL             Hottest Caller

-------------------------------------------------------------------------
13641  92%  92% 1.00      253 cpu[3]              (usermode)
  579   4%  96% 1.00      325 cpu[3]+6            ip_ocsum+0xe8
  375   3%  99% 1.00      411 cpu[3]+10           splx
  154   1% 100% 1.00      527 cpu[3]+4            fas_intr_svc+0x80
```

EXAMPLE 7 Determining the Average PIL (processor interrupt level) for a CPU *(Continued)*

```
   41   0% 100% 1.00      293 cpu[3]+13           send_mondo+0x18
    1   0% 100% 1.00      266 cpu[3]+12           zsa_rxint+0x400
-------------------------------------------------------------------------
```

FILES

`/dev/lockstat` Lockstat driver

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu (32-bit)
	SUNWcsxu (64-bit)

SEE ALSO

`attributes`(5), `lockstat`(7D), `mutex`(9F), `rwlock`(9F)

NOTES

The profiling support provided by `lockstat -I` replaces the old (and undocumented) `/usr/bin/kgmon` and `/dev/profile`.

Tail-call elimination may affect call sites. For example, if `foo()+0x50` calls `bar()` and the last thing `bar()` does is call `mutex_exit()`, the compiler may arrange for `bar()` to branch to `mutex_exit()` with a return address of `foo()+0x58`. Thus, the `mutex_exit()` in `bar()` will appear as though it occurred at `foo()+0x58`.

The PC in the stack frame in which an interrupt occurs may be bogus because, between function calls, the compiler is free to use the return address register for local storage.

When using the `-I` and `-s` options together, the interrupted PC will usually not appear anywhere in the stack since the interrupt handler is entered asynchronously, not by a function call from that PC.

The `lockstat` technology is provided on an as-is basis. The format and content of `lockstat` output reflect the current Solaris kernel implementation and are therefore subject to change in future releases.

NAME

lofiadm – administer files available as block devices through lofi

SYNOPSIS

/usr/sbin/lofiadm `-a` *file* [*device*]

/usr/sbin/lofiadm `-d` *file* | *device*

/usr/sbin/lofiadm [*file* | *device*]

DESCRIPTION

`lofiadm` administers `lofi`(7D), the loopback file driver. `lofi`(7D) allows a file to be associated with a block device. That file can then be accessed through the block device. This is useful when the file contains an image of some filesystem (such as a floppy or CD-ROM image), because the block device can then be used with the normal system utilities for mounting, checking or repairing filesystems. See `fsck`(1M) and `mount`(1M).

Use `lofiadm` to add a file as a loopback device, remove such an association, or print information about the current associations.

OPTIONS

The following options are supported:

`-a` *file* [*device*] Add *file* as a block device.

If *device* is not specified, an available device is picked.

If *device* is specified, `lofiadm` attempts to assign it to *file*. *device* must be available or `lofiadm` will fail. The ability to specify a device is provided for use in scripts that wish to re-establish a particular set of associations.

`-d` *file* | *device* Remove an association by *file* or *device* name, if the associated block device is not busy, and deallocates the block device.

OPERANDS

The following operands are supported:

file Print the block device associated with *file*.

device Print the file name associated with the block device *device*.

Without arguments, print a list of the current associations. Filenames must be valid absolute pathnames.

When a file is added, it is opened for reading or writing by root. Any restrictions apply (such as restricted root access over NFS). The file is held open until the association is removed. It is not actually accessed until the block device is used, so it will never be written to if the block device is only opened read-only.

EXAMPLES

EXAMPLE 1 Mounting an Existing CD-ROM Image

You should ensure that Solaris understands the image before creating the CD. `lofi` allows you to mount the image and see if it works.

EXAMPLE 1 Mounting an Existing CD-ROM Image *(Continued)*

This example mounts an existing CD-ROM image (`sparc.iso`), of the `Red Hat 6.0 CD` which was downloaded from the Internet. It was created with the `mkisofs` utility from the Internet.

Use `lofiadm` to attach a block device to it:

```
# lofiadm -a /home/mike_s/RH6.0/sparc.iso
/dev/lofi/1
```

`lofiadm` picks the device and prints the device name to the standard output. You can run `lofiadm` again by issuing the following command:

```
# lofiadm
Block Device     File
/dev/lofi/1      /home/mike_s/RH6.0/sparc.iso
```

Or, you can give it one name and ask for the other, by issuing the following command:

```
# lofiadm /dev/lofi/1
/home/mike_s/RH6.0/sparc.iso
```

Use the `mount` command to mount the image:

```
# mount -F hsfs -o ro /dev/lofi/1 /mnt
```

Check to ensure that Solaris understands the image:

```
# df -k /mnt
Filesystem            kbytes    used   avail capacity  Mounted on
/dev/lofi/1           512418  512418       0   100%    /mnt
# ls /mnt
./            RedHat/       doc/          ls-lR         rr_moved/
../           TRANS.TBL     dosutils/     ls-lR.gz      sbin@
.buildlog     bin@          etc@          misc/         tmp/
COPYING       boot/         images/       mnt/          usr@
README        boot.cat*     kernels/      modules/
RPM-PGP-KEY   dev@          lib@          proc/
```

Solaris can mount the CD-ROM image, and understand the filenames. The image was created properly, and you can now create the CD-ROM with confidence.

As a final step, unmount and detach the images:

```
# umount /mnt
# lofiadm -d /dev/lofi/1
# lofiadm
Block Device             File
```

EXAMPLE 2 Mounting a Floppy Image

This is similar to Example 1.

EXAMPLE 2 Mounting a Floppy Image *(Continued)*

Using `lofi` to help you mount files that contain floppy images is helpful if a floppy disk contains a file that you need, but the machine which you're on doesn't have a floppy drive. It is also helpful if you don't want to take the time to use the `dd` command to copy the image to a floppy.

This is an example of getting to MDB floppy for Solaris x86:

```
# lofiadm -a /export/s28/MDB_s28x_wos/latest/boot.3
/dev/lofi/1
# mount -F pcfs /dev/lofi/1 /mnt
# ls /mnt
./            COMMENT.BAT*  RC.D/         SOLARIS.MAP*
../           IDENT*        REPLACE.BAT*  X/
APPEND.BAT*   MAKEDIR.BAT*  SOLARIS/
# umount /mnt
# lofiadm -d /export/s28/MDB_s28x_wos/latest/boot.3
```

EXAMPLE 3 Making a UFS Filesystem on a File

Making a UFS filesystm on a file can be useful, particularly if a test suite requires a scratch filesystem. It can be painful (or annoying) to have to re-partition a disk just for the test suite, but you don't have to. You can `newfs` a file with `lofi`

Create the file:

```
# mkfile 35m /export/home/test
```

Attach it to a block device. You also get the character device that `newfs` requires, so `newfs` that:

```
# lofiadm -a /export/home/test
/dev/lofi/1
# newfs /dev/rlofi/1
newfs: construct a new file system /dev/rlofi/1: (y/n)? y
/dev/rlofi/1:   71638 sectors in 119 cylinders of 1 tracks, 602 sectors
        35.0MB in 8 cyl groups (16 c/g, 4.70MB/g, 2240 i/g)
super-block backups (for fsck -F ufs -o b=#) at:
 32, 9664, 19296, 28928, 38560, 48192, 57824, 67456,
```

Note that `ufs` might not be able to use the entire file. Mount and use the filesystem:

```
# mount /dev/lofi/1 /mnt
# df -k /mnt
Filesystem            kbytes    used   avail capacity  Mounted on
/dev/lofi/1            33455       9   30101     1%    /mnt
# ls /mnt
./           ../          lost+found/
# umount /mnt
# lofiadm -d /dev/lofi/1
```

EXAMPLE 4 Creating a PC (FAT) File System on a Unix File

The following series of commands creates a FAT file system on a Unix file. The file is associated with a block device created by `lofiadm`.

```
# mkfile 10M /export/test/testfs
# lofiadm -a /export/test testfs
/dev/lofi/1
Note use of rlofi, not lofi, in following command.
# mkfs -F pcfs -o nofdisk,size=20480 /dev/rlofi/1
Construct a new FAT file system on /dev/rlofi/1: (y/n)? y
# mount -F pcfs /dev/lofi/1 /mnt
# cd /mnt
# df -k .
Filesystem            kbytes    used   avail capacity  Mounted on
/dev/lofi/1            10142       0   10142     0%    /mnt
```

ENVIRONMENT VARIABLES

See `environ`(5) for descriptions of the following environment variables that affect the execution of `lofiadm`: `LC_CTYPE`, `LC_MESSAGES` and `NLSPATH`.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`fsck`(1M), `mount`(1M), `mount_ufs`(1M), `attributes`(5), `lofi`(7D), `lofs`(7FS)

NOTES

Just as you would not directly access a disk device that has mounted file systems, you should not access a file associated with a block device except through the `lofi` file driver. It might also be appropriate to ensure that the file has appropriate permissions to prevent such access.

Associations are not persistant across reboots. A script can be used to re-establish them if required.

The abilities of `lofiadm`, and who can use them, are controlled by the permissions of `/dev/lofictl`. Read-access allows query operations, such as listing all the associations. Write-access is required to do any state-changing operations, like adding an association. As shipped, `/dev/lofictl` is owned by `root`, in group `sys`, and mode `0644`, so all users can do query operations but only root can change anything. The administrator can give users write-access, allowing them to add or delete associations, but that is very likely a security hole and should probably only be given to a trusted group.

When mounting a filesystem image, take care to use appropriate mount options. In particular, the `nosuid` mount option might be appropriate for UFS images whose origin is unknown. Also, some options might not be useful or appropriate, like `logging` or `forcedirectio` for UFS. For compatibility purposes, a raw device is also exported along with the block device. For example, `newfs`(1M) requires one.

The output of `lofiadm` (without arguments) might change in future releases.

NAME

logadm – manage endlessly growing log files

SYNOPSIS

logadm

logadm [-*options*] *logname*...

DESCRIPTION

`logadm` is a general log rotation tool that is suitable for running from `cron`(1M).

Without arguments, `logadm` reads the `/etc/logadm.conf` file, and for every entry found in that file checks the corresponding log file to see if it should be rotated. Typically this check is done each morning by an entry in the root's `crontab`.

If the *logname* argument is specified, `logadm` renames the corresponding log file by adding a suffix so that the most recent log file ends with `.0` (that is, *logfile*`.0`), the next most recent ends with `.1` (that is, *logfile*`.1`), and so forth. By default, ten versions of old log files are kept (that is, *logfile*`.0` through *logfile*`.9`) and `logadm` automatically deletes the oldest version when appropriate to keep the count of files at ten.

`logadm` takes a number of *options*. You can specify these options on the command line or in the `/etc/logadm.conf`file . The `logadm` command searches `/etc/logadm.conf` for lines of the form *logname options*

logname Identifies the name of the entry in `/etc/logadm.conf`, but if no log file name is given in that entry it is assumed that the logname is the same as the actual log file name.

options Identifies command line options exactly as they would be entered on the command line. This allows commonly used log rotation policies to be stored in the `/etc/logadm.conf` file. See `EXAMPLES`.

If *options* are specified both in `/etc/logadm.conf` and on the command line, those in the `/etc/logadm.conf` file are applied first. Therefore, the command line options override those in `/etc/logadm.conf`.

Log file names specified in `/etc/logadm.conf` may contain filename substitution characters such as `*` and `?`, that are supported by `csh`(1).

Two options are available which control when a log file is rotated. They are: `-s` size `-p` period.

When using more than one of these options at a time, there is an implied *and* between them. This means that all conditions must be met before the log is rotated.

If neither of these two options are specified, the default conditions for rotating a log file are: `-s 1b -p 1w`, which means the log file is only rotated if the size is non-zero and if at least 1 week has passed since the last time it was rotated.

By specifying `-p never` as a rotation condition, any other rotation conditions are ignored and `logadm` moves on to the expiration of old log files. By specifying `-p now` as a rotation condition, a log rotation is forced.

Unless specified by the `-o`, `-g`, or `-m` options, `logadm` replaces the log file (after renaming it) by creating an empty file whose owner, group ID, and permissions match the original file.

Three options control when old log files are expired: `-A` age `-C` count `-S` size. These options expire the oldest log files until a particular condition or conditions are met. For example, the combination `-C 5` and the `-S 10m` options expires old log files until there are no more than 5 of the *and* their combined disk usage is no more than 10 megabytes. If none of these options are specified, the default expiration is `-C 10` which keeps ten old log files. If no files are to be expired, use `-C 0` to prevent expiration by default.

OPTIONS

The following options are supported:

`-a` *post_command*
: Execute the *post_command* after renaming the log file. *post_command* is passed to `sh -c`.

  Specify *post_command* as a valid shell command. Use quotes to protect spaces or shell metacharacters in *post_command*.

  This option can be used to restart a daemon that is writing to the file. When rotating multiple logs with one `logadm` command, *post_command* is executed only once after all the logs are rotated, not once per rotated log.

`-A` *age*
: Delete any versions that have not been modified for the amount of time specified by *age*.

  Specify *age* as a number followed by an `h` (hours), `d` (days), `w`(weeks), `m` (months), or `y` (years).

`-b` *pre_command*
: Execute *pre_command* before renaming the log file. *pre_command* is passed to `sh -c`.

  Specify *pre_command* as a valid shell command. Use quotes to protect spaces or shell metacharacters in the *pre_command*.

  This option can be used to stop a daemon that is writing to the file. When rotating multiple logs with one `logadm` command, *pre_command* is executed only once before all the logs are rotated, not once per rotated log.

`-c`
: Rotate the log file by copying it and truncating the original logfile to zero length, rather than renaming the file.

`-C` *count*
: Delete the oldest versions until there are not more than *count* files left.

  If no expire options (`-A`, `-C`, or `-S`) are specified, `-C 10` is the default. To prevent the default expire rule from being added automatically, specify `-C 0` .

`-e` *mail_addr*
: Send error messages by email to *mail_addr*.

  As `logadm` is typically run from `cron`(1M), error messages are captured by `cron` and mailed to the owner of the `crontab`.

  This option is useful you want the mail regarding error messages to go to another address instead. If no errors are encountered, no mail message is generated.

`-E` *cmd*
: Execute *cmd* to expire the file, rather than deleting the old log file to expire it.

  cmd is passed it to `sh -c`. The file is considered expired after *cmd* completes. If the old log file is not removed or renamed by the *cmd*, `logadm` considers it for expiration the next time that it runs on the specified log file. If present, the keyword `$file` is expanded in the specified *cmd* to the name of the file being expired.

  This option is useful for tasks such as mailing old log files to administrators, or copying old log files to long term storage.

`-f` *conf_file*
: Use *conf_file* instead of `/etc/logadm.conf`.

  This option allows non-root users to keep their own `logadm` configuration files.

`-g` *group*
: Create a new empty file with the ID specified by *group*, instead of preserving the group ID of the log file.

  Specify *group* by name or by numeric group ID, as accepted by `chgrp`(1).

  This option requires the ability to change file group ownership using the `chgrp`(1) command.

`-h`
: Print a help message that describes `logadm`'s options.

`-m` *mode*
: Create a new empty file with the mode specified by *mode*, instead of preserving the mode of the log file.

  Specify *mode* in any form that is accepted by the `chmod`(1) command.

`-n`
: Print the actions that the `logadm` command will perform without actually performing them.

  This option is useful for checking arguments before making any changes to the system.

It is important to remember, however, that since log rotating actions are only printed with this option, `logadm` might not find files that need expiring, but if run without the `-n` `logadm` might create a file that needs expiring by performing the log rotating actions. Therefore, if you see no files being expired with the `-n` option, files still might be expired without it.

`-N` Prevent an error message if the specified logfile does not exist. Normally, `logadm` produces an error message if the log file is not found. With `-N`, if the log file doesn't exist `logadm` moves on to the expire rules (if any) and then to the next log file (if any), without creating the empty replacement log file.

`-o` *owner* Create the new empty file with *owner*, instead of preserving the owner of the log file.

Specify *owner* in any form that is accepted by the `chown`(1) command.

`-p` *period* Rotate a log file after the specified time period (*period*) .

Specify *period* as a number followed by `d` for days, `w` for weeks, `m` for months (really 30 days) or `y` for years. There are also two special values for period: `now` and `never`. `-p now` forces log rotation. `-p never` forces no log rotation.

`-P` *timestamp* Used by `logadm` to record the last time the log was rotated in `/etc/logadm.conf`.

This option uses *timestamp* to determine if the log rotation period has passed. The format of *timestamp* matches the format generated by `ctime`(3C), with quotes around it to protect embedded spaces.

`-r` Remove any entries corresponding to the specified *logname* from the `/etc/logadm.conf`.

`-R` *cmd* Run the *cmd* when an old log file is created by a log rotation. If the keyword `$file` is embedded in the specified command, it is expanded to the name of the old log file just created by log rotation.

This option is useful for processing log file contents after rotating the log. *cmd* is executed by passing it to `sh -c`. When rotating multiple logs with one logadm command, the command supplied with `-R` is executed once every time a log is rotated. This is useful for post-processing a log file (that is, sorting it, removing uninteresting lines, etc.). The `-a` option is a better choice for restarting daemons after log rotation.

`-s` *size* Rotate the log file only if its size is greater than or equal to *size*.

Specify *size* as a number followed by the letter `b` for bytes, `k` for kilobytes, `m` for megabytes, or `g` for gigabytes.

`-S` *size*

Delete the oldest versions until the total disk space used by the old log files is less than the specified size.

Specify *size* as a number followed by the letter `b` for bytes, `k` for kilobytes, `m` for megabytes, or `g` for gigabytes.

`-t` *template*

Specify the template to use when renaming log files.

template can be a simple name, such as `/var/adm/oldfile`, or it can contain special keywords which are expanded by `logadm` and are in the form `$`*word*. Allowed sequences are:

$*file*	The full path name of the file to be rotated
$*dirname*	The directory of the file to be rotated
$*basename*	The log file name, without the directory name
$*n*	The version number, `0` is most recent, `1` is next most recent, and so forth
$*N*	The same as $*n*, but starts at `1` instead of zero
$*secs*	The number of seconds since `00:00:00 UTC, January 1,1970`
$*nodename*	Expands to the output of `uname -n`
$*platform*	Expands to the output of `uname -i`
$*isa*	Expands to the output of `uname -p`
$*release*	Expands to the output of `uname -r`
$*machine*	Expands to the output of `uname -m`
$*domain*	Expands to the output of `domainname`

To actually have the dollar sign character in the file name, use `$$`. Any percent sequences allowed by `strftime`(3C) are also allowed, for example, `%d` expands to the day of the month. To actually have a percent sign character in the file name, use `%%`. Both dollar-sign keywords and percent sequences can appear anywhere in the template. If the template results in a pathname with non-existent directories, they are created as necessary when rotating the log file.

If no `-t` option is specified, the default template is `$file.$n`. Actual *rotation* of log files, where each version is shifted up until it expires is done using the `$n` keyword. If the template does not contain the `$n` keyword, the log file is simply renamed to the new name and then the expire rules, if any, are applied.

`-T` *pattern*

Normally `logadm` looks for a list of old log files by turning the template (specified with the `-t` option) into a pattern and finding existing files whose names match that pattern. The `-T` option causes the given pattern to be used instead.

This option is useful if another program fiddles with the old log file names, like a `cron` job to compress them over time. The pattern is in the form of a pathname with special characters such as `*` and `?` as supported by `csh`(1) filename substitution.

`-v`

Print information about the actions being executed in verbose mode.

`-V`

Validate the configuration file.

This option validates that an entry for the specified *logname* exists in the `/etc/logadm.conf` file and is syntactically correct. If *logname* is not specified, all entries in the configuration file are validated. If a `logname` argument is specified, the command validates the syntax of that entry. If the entry is found, it is printed and the exit value of the command is true. Otherwise the exit value is false.

`-w` *entryname*

Write an entry into the config file (that is, `/etc/logadm.conf`) which corresponds to the current command line arguments. If an entry already existed for the specified entryname, it is removed first. This is the preferred method for updating `/etc/logadm.conf` since using it prevents syntax errors in that file. The entryname is the name of the entry in `/etc/logadm.conf`, and that name can be used as the "*logname*" argument to future calls to logadm to take advantage of that entry. The entryname can be chosen to be something that is easy to specify, or it can be the actual log file name. If no log file name is provided on the command line, the entry name is assumed to be the same as the log file name. For example, the following two lines achieve the same thing, keeping two copies of rotated log files, but the first example names the entry something easier to enter on the command line:

```
% logadm -C2 -w mylog /my/really/long/log/file/name
% logadm -C2 -w /my/really/long/log/file/name
```

`-z` *count*
: Compress old log files as they are created. *count* of the most recent log files are left uncompressed, therefore making the *count* most recent files easier to peruse. Use *count* of zero to compress all old logs.

  The compression is done with `gzip`(1) and the resulting log file has the suffix of `.gz`.

OPERANDS

The following operands are supported:

logname
: Identifies the name of the entry in `/etc/logadm.conf`. If the log file name is specified in the *logname* field, it is assumed that *logname* is the same as the actual log file name.

EXAMPLES

EXAMPLE 1 Rotating a File and Keeping Previous Versios

The following example rotates the `/var/adm/exacct/proc` file, keeping ten previous versions in `/var/adm/exacct/proc.0` through `/var/adm/exacct/proc.9`.

Tell `logadm` to copy the file and truncate it.

```
% logadm -c /var/adm/exacct/proc
```

EXAMPLE 2 Rotating syslog

The following example rotates `syslog` and keeps eight log files. Old log files are put in the directory `/var/oldlogs` instead of `/var/log`:

```
% logadm -C8 -t'/var/oldlogs/syslog.$n' /var/log/syslog
```

EXAMPLE 3 Rotating /var/adm/sulog and Expiring Based on Age

The following entry in the `/etc/logadm.conf` file rotates the `/var/adm/sulog` file and expires any copies older than 30 days.

```
/var/adm/sulog -A 30d
```

EXAMPLE 4 Rotating Files and Expiring Based on Disk Usage

The following entry in the `/etc/logadm.conf` file rotates the `/var/adm/sulog` file and expires old log files when more than 100 megabytes are used by the sum of all the rotated log files.

```
/var/adm/sulog -S 100m
```

EXAMPLE 5 Creating an Entry that Stores the Logfile Name

This example creates an entry storing the log file name and the fact that we want to keep 20 copies in `/etc/logadm.conf`, but the `-p never` means the entry is ignored by the normal `logadm` run from root's crontab every morning.

```
% logadm -w locallog /usr/local/logfile -C20 -p never
```

Use the following entry on the command line to override the `-p never` option:

```
 % logadm -p now locallog
```

EXAMPLE 6 Rotating the apache Error and Access Logs

The following example rotates the apache error and access logs monthly to filenames based on current year and month. It keeps the 24 most recent copies and tells apache to restart after renaming the logs.

This command is run once, and since the `-w` option is specified, an entry is made in `/etc/logadm.conf` so the apache logs are rotated from now on.

```
 % logadm -w apache -p 1m -C 24\
     -t '/var/apache/old-logs/$basename.%Y-%m'\
     -a '/usr/apache/bin/apachectl graceful'\
     '/var/apache/logs/*{access,error}_log'
```

This example also illustrates that the entry name supplied with the `-w` option doesn't have to match the log file name. In this example, the entry name is `apache` and once the line has been run, the entry in `/etc/logadm.conf` can be forced to run by executing the following command:

```
% logadm -p now apache
```

Because the expression matching the apache log file names was enclosed in quotes, the expression is stored in `/etc/logadm.conf`, rather than the list of files that it expands to. This means that each time `logadm` runs from `cron` it expands that expression and checks all the log files in the resulting list to see if they need rotating.

The following command is an example without the quotes around the log name expression. The shell expands the last argument into a list of log files that exist at the time the command is entered, and writes an entry to `/etc/logadm.conf` that rotates the files.

```
logadm -w apache /var/apache/logs/*_log
```

FILES

`/etc/logadm.conf` configuration file for logadm command

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO `chgrp`(1), `chmod`(1), `chown`(1), `csh`(1), `gzip`(1), `cron`(1M), `ctime`(3C), `strftime`(3C), `logadm.conf`(4), `attributes`(5)

NOTES When `logadm` applies expire conditions (supplied by the `-A`, `-C`, and `-S` options), it deletes files, the oldest first, until the conditions are satisfied. If the template used for naming the old logs contained `$n` or `$N`, `logadm` picks the highest value of `$n` or `$N` found in the old log file names first. If the template used is something else, `logadm` uses the modification time to determine which files to expire first. This may not be the expected behavior if an old log file has been modified since it was rotated.

NAME

logins – list user and system login information

SYNOPSIS

/usr/bin/logins [-admopstux] [-g *group...*] [-l *login...*]

DESCRIPTION

This command displays information on user and system logins known to the system. Contents of the output is controlled by the command options and can include the following: user or system login, user id number, `passwd` account field value (user name or other information), primary group name, primary group id, multiple group names, multiple group ids, home directory, login shell, and four password aging parameters. The default information is the following: login id, user id, primary group name, primary group id and the account field value. Output is sorted by user id, system logins, followed by user logins.

OPTIONS

The following options are supported:

Options may be used together. If so, any login that matches any criteria will be displayed.

`-a` Add two password expiration fields to the display. The fields show how many days a password can remain unused before it automatically becomes inactive, and the date that the password will expire.

`-d` Selects logins with duplicate uids.

`-m` Displays multiple group membership information.

`-o` Formats output into one line of colon-separated fields.

`-p` Selects logins with no passwords.

`-s` Selects all system logins.

`-t` Sorts output by login instead of by uid.

`-u` Selects all user logins.

`-x` Prints an extended set of information about each selected user. The extended information includes home directory, login shell and password aging information, each displayed on a separate line. The password information consists of password status (PS for password, NP for no password or LK for locked). If the login is passworded, status is followed by the date the password was last changed, the number of days required between changes, and the number of days allowed before a change is required. The password aging information shows the time interval that the user will receive a password expiration warning message (when logging on) before the password expires.

`-g` *group* Selects all users belonging to `group`, sorted by login. Multiple groups can be specified as a comma-separated list. When the `-l` and `-g` options are combined, a user will only be listed once, even if the user belongs to more than one of the selected groups.

`-l login` Selects the requested login. Multiple logins can be specified as a comma-separated list. Depending on the nameservice lookup types set in `/etc/nsswitch.conf`, the information can come from the `/etc/passwd` and `/etc/shadow` files and other nameservices. When the `-l` and `-g` options are combined, a user will only be listed once, even if the user belongs to more than one of the selected groups.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`attributes`(5)

NAME

lpadmin – configure the LP print service

SYNOPSIS

lpadmin -p *printer options*

lpadmin -x *dest*

lpadmin -d [*dest*]

lpadmin -S *print-wheel* -A *alert-type* [-W *minutes*] [-Q *requests*]

lpadmin -M -f *form-name* [-a [-o filebreak] [-t *tray-number*]]

DESCRIPTION

`lpadmin` configures the LP print service by defining printers and devices. It is used to add and change printers, to remove printers from service, to set or change the system default destination, to define alerts for printer faults, and to mount print wheels.

Adding or Changing a Printer

The first form of the `lpadmin` command (`lpadmin -p` *printer options*) configures a new printer or changes the configuration of an existing printer. It also starts the print scheduler.

When creating a new printer, one of three options (`-v`, `-U`, or `-s`) must be supplied. In addition, only one of the following may be supplied: `-e`, `-i`, or `-m`; if none of these three options is supplied, the model standard is used. The `-h` and `-l` options are mutually exclusive. Printer and class names may be no longer than 14 characters and must consist entirely of the characters `A-Z`, `a-z`, `0-9`, dash (`-`) and underscore (_). If `-s` is specified, the following options are invalid: `-A`, `-e`, `-F`, `-h`, `-i`, `-l`, `-M`, `-m`, `-o`, `-U`, `-v`, and `-W`.

The following *printer options* may appear in any order.

`-A` *alert-type* [`-W` *minutes*]
The `-A` option is used to define an alert that informs the administrator when a printer fault is detected, and periodically thereafter, until the printer fault is cleared by the administrator. The *alert-types* are:

`mail`
Send the alert message using mail (see `mail`(1)) to the administrator.

`write`
Write the message to the terminal on which the administrator is logged in. If the administrator is logged in on several terminals, one is chosen arbitrarily.

`quiet`
Do not send messages for the current condition. An administrator can use this option to temporarily stop receiving further messages about a known problem. Once the fault has been cleared and printing resumes, messages will again be sent when another fault occurs with the printer.

`showfault`
Attempt to execute a fault handler on each system that has a print job in the queue. The fault handler is `/etc/lp/alerts/printer`. It is invoked with three parameters: *printer_name*, `date`, *file_name*. The *file_name* is the name of a file containing the fault message.

`none`
Do not send messages; any existing alert definition for the printer will be removed. No alert will be sent when the printer faults until a different alert-type (except `quiet`) is used.

shell-command
Run the *shell-command* each time the alert needs to be sent. The shell command should expect the message in standard input. If there are blank spaces embedded in the command, enclose the command in quotes. Notice that the `mail` and `write` values for this option are equivalent to the values `mail` *user-name* and `write` *user-name* respectively, where *user-name* is the current name for the administrator. This will be the login name of the person submitting this command unless he or she has used the `su` command to change to another user ID. If the `su` command has been used to change the user ID, then the *user-name* for the new ID is used.

`list`
Display the type of the alert for the printer fault. No change is made to the alert.

The message sent appears as follows:

```
The printer printer has stopped printing for the reason given below.
Fix the problem and bring the printer back on line.
Printing has stopped, but will be restarted in a few minutes;
issue an enable commant if you want to restart sooner.
```

Unless someone issues the change request:

`lp -i` *request-id* `-P` `...`to change the page list to print, the current request will be reprinted from the beginning. The reason(s) it stopped (multiple reasons indicate reprinted attempts):*reason*

The LP print service can detect printer faults only through an adequate fast filter and only when the standard interface program or a suitable customized interface program is used. Furthermore, the level of recovery after a fault depends on the capabilities of the filter.

If the *printer* is `all`, the alerting defined in this command applies to all existing printers.

If the `-W` option is not used to arrange fault alerting for *printer*, the default procedure is to mail one message to the administrator of *printer* per fault. This is equivalent to specifying `-W once` or `-W 0`. If *minutes* is a number greater than zero, an alert will be sent at intervals specified by *minutes*.

`-c` *class*
Insert *printer* into the specified *class*. *class* will be created if it does not already exist.

`-D` *comment*
Save this *comment* for display whenever a user asks for a full description of *printer* (see `lpstat`(1)). The LP print service does not interpret this comment.

`-e` *printer*
Copy the interface program of an existing *printer* to be the interface program for *printer*. (Options `-i` and `-m` may not be specified with this option.)

`-f allow:`*form-list*
`-f deny:`*form-list*
Allow or deny the forms in *form-list* to be printed on *printer*. By default no forms are allowed on a new printer.

For each printer, the LP print service keeps two lists of forms: an ''allow-list'' of forms that may be used with the printer, and a ''deny-list'' of forms that may not be used with the printer. With the `-f allow` option, the forms listed are added to the allow-list and removed from the deny-list. With the `-f deny` option, the forms listed are added to the deny-list and removed from the allow-list.

If the allow-list is not empty, only the forms in the list may be used on the printer, regardless of the contents of the deny-list. If the allow-list is empty, but the deny-list is not, the forms in the deny-list may not be used with the printer. All forms can be excluded from a printer by specifying `-f deny:all`. All forms can be used on a printer (provided the printer can handle all the characteristics of each form) by specifying `-f allow:all`.

The LP print service uses this information as a set of guidelines for determining where a form can be mounted. Administrators, however, are not restricted from mounting a form on any printer. If mounting a form on a particular printer is in disagreement with the information in the allow-list or deny-list, the administrator is warned but the mount is accepted. Nonetheless, if a user attempts to issue a print or change request for a form and printer combination that is in disagreement with the information, the request is accepted only if the form is currently mounted on the printer. If the form is later unmounted before the request can print, the request is canceled and the user is notified by mail.

If the administrator tries to specify a form as acceptable for use on a printer that doesn't have the capabilities needed by the form, the command is rejected.

Notice the other use of `-f`, with the `-M` option, below.

The `-T` option must be invoked first with `lpadmin` to identify the printer type before the `-f` option can be used.

`-F` *fault-recovery*
This option specifies the recovery to be used for any print request that is stopped because of a printer fault, according to the value of *fault-recovery*:

`continue`	Continue printing on the top of the page where printing stopped. This requires a filter to wait for the fault to clear before automatically continuing.
`beginning`	Start printing the request again from the beginning.

`wait` Disable printing on *printer* and wait for the administrator or a user to enable printing again.

During the wait, the administrator or the user who submitted the stopped print request can issue a change request that specifies where printing should resume. (See the `-i` option of the `lp` command.) If no change request is made before printing is enabled, printing resumes at the top of the page where stopped, if the filter allows; otherwise, the request is printed from the beginning.

`-h`
Indicate that the device associated with the printer is hardwired. If neither of the mutually exclusive options, `-h` and `-l`, is specified, `-h` is assumed.

`-i` *interface*
Establish a new interface program for *printer*. *interface* is the pathname of the new program. (The `-e` and `-m` options may not be specified with this option.)

`-I` *content-type-list*
Allow *printer* to handle print requests with the content types listed in a *content-type-list*. If the list includes names of more than one type, the names must be separated by commas or blank spaces. (If they are separated by blank spaces, the entire list must be enclosed in double quotes.)

The type `simple` is recognized as the default content type for files in the UNIX system. A `simple` type of file is a data stream containing only printable ASCII characters and the following control characters:

Control Char	Octal Value	Meaning
BACKSPACE	10	Move back one char, except at beginning of line
TAB	11	Move to next tab stop
LINEFEED (newline)	12	Move to beginning of next line
FORMFEED	14	Move to beginning of next page
RETURN	15	Move to beginning of current line

To prevent the print service from considering `simple` a valid type for the printer, specify either an explicit value (such as the printer type) in the *content-type-list*, or an empty list. If you do want `simple` included along with other types, you must include `simple` in the *content-type-list*.

In addition to content types defined by the print administrator, the type `PostScript` is recognized and supported by the Solaris print subsystem. This includes filters to support `PostScript` as the printer content type.

The type `any` is recognized as a special content type for files. When declared as the input type for a printer, it signals the print sub-system not to do any filtering on the file before sending it to the printer.

Except for `simple` and `any`, each *content-type* name is freely determined by the administrator. If the printer type is specified by the `-T` option, then the printer type is implicitly considered to be also a valid content type.

`-l`
Indicate that the device associated with *printer* is a login terminal. The LP scheduler (`lpsched`) disables all login terminals automatically each time it is started. (The `-h` option may not be specified with this option.)

`-m` *model*
Select *model* interface program, provided with the LP print service, for the printer. (Options `-e` and `-i` may not be specified with this option.)

`-M -f` *form-name* [`-a` [`-o filebreak`]] [`-t` *tray-number*]]
Mount the form *form-name* on *printer*. Print requests that need the pre-printed form *form-name* will be printed on *printer*. If more than one printer has the form mounted and the user has specified `any` (with the `-d` option of the `lp` command) as the printer destination, then the print request will be printed on the one printer that also meets the other needs of the request.

The page length and width, and character and line pitches needed by the form are compared with those allowed for the printer, by checking the capabilities in the `terminfo` database for the type of printer. If the form requires attributes that are not available with the printer, the administrator is warned but the mount is accepted. If the form lists a print wheel as mandatory, but the print wheel mounted on the printer is different, the administrator is also warned but the mount is accepted.

If the `-a` option is given, an alignment pattern is printed, preceded by the same initialization of the physical printer that precedes a normal print request, with one exception: no banner page is printed. Printing is assumed to start at the top of the first page of the form. After the pattern is printed, the administrator can adjust the mounted form in the printer and press return for another alignment pattern (no initialization this time), and can continue printing as many alignment patterns as desired. The administrator can quit the printing of alignment patterns by typing `q`.

If the `-o filebreak` option is given, a formfeed is inserted between each copy of the alignment pattern. By default, the alignment pattern is assumed to correctly fill a form, so no formfeed is added.

If the `-t` *tray-number* option is specified, printer tray *tray-number* will used.

A form is "unmounted" either by mounting a new form in its place or by using the `-f none` option. By default, a new printer has no form mounted.

Notice the other use of `-f` without the `-M` option above.

`-M -S` *print-wheel*
Mount the *print-wheel* on *printer*. Print requests that need the *print-wheel* will be printed on *printer*. If more than one printer has *print-wheel* mounted and the user has specified `any` (with the `-d` option of the `lp` command) as the printer destination, then the print request will be printed on the one printer that also meets the other needs of the request.

If the *print-wheel* is not listed as acceptable for the printer, the administrator is warned but the mount is accepted. If the printer does not take print wheels, the command is rejected.

A print wheel is "unmounted" either by mounting a new print wheel in its place or by using the option `-S none`. By default, a new printer has no print wheel mounted.

Notice the other uses of the `-S` option without the `-M` option described below.

`-o` *option*
The `-o` option defines default printer configuration values given to an interface program. The default may be explicitly overwritten for individual requests by the user (see `lp`(1)), or taken from a preprinted form description (see `lpforms`(1M) and `lp`(1)).

There are several options which are pre-defined by the system. In addition, any number of key-value pairs may be defined. Each of the predefined and undefined options are described.

`The Predefined Options`

The following options are predefined: adjusting printer capabilities, adjusting printer port characteristics, configuring network printers, and controlling the use of banner.

`Adjusting Printer Capabilities`

`length=`*scaled-decimal-number*
`width=`*scaled-decimal-number*
`cpi=`*scaled-decimal-number*
`lpi=`*scaled-decimal-number*

The term *scaled-decimal-number* refers to a non-negative number used to indicate a unit of size. The type of unit is shown by a ''trailing'' letter attached to the number. Three types of *scaled-decimal-numbers* can be used with the LP print service: numbers that show sizes in centimeters (marked with a trailing `c`); numbers that show sizes in inches (marked with a trailing `i`); and numbers that show sizes in units appropriate to use (without a trailing letter), that is, lines, characters, lines per inch, or characters per inch.

The option values must agree with the capabilities of the type of physical printer, as defined in the terminfo database for the printer type. If they do not, the command is rejected.

The defaults are defined in the `terminfo` entry for the specified printer type. The defaults may be reset by:

lpadmin -p *printername* **-o length=**
lpadmin -p *printername* **-o width=**
lpadmin -p *printername* **-o cpi=**
lpadmin -p *printername* **-o lpi=**

Adjusting Printer Port Characteristics

stty="'*stty-option-list*'"

The *stty-option-list* is not checked for allowed values, but is passed directly to the stty program by the standard interface program. Any error messages produced by `stty` when a request is processed (by the standard interface program) are mailed to the user submitting the request.

The default for `stty` is:

```
stty="'9600 cs8 -cstopb -parenb ixon
     -ixany opost -olcuc onlcr
     -ocrnl -onocr
     -onlret -ofill nl0 cr0 tab0 bs0 vt0 ff0'"
```

The default may be reset by:

lpadmin -p *printername* **-o stty=**

Configuring Network Printers

dest=*string* protocol=*string* bsdctrl=*string* \
timeout=*non-negative-integer-seconds*

These four options are provided to support network printing. Each option is passed directly to the interface program; any checking for allowed values is done there.

The value of `dest` is the name of the destination for the network printer; the semantics for value `dest` are dependent on the printer and the configuration. There is no default.

The value of option `protocol` sets the over-the-wire protocol to the printer. The default for option `protocol` is `bsd`. The value of option `bsdctrl` sets the print order of control and data files (BSD protocol only); the default for this option is `control file first`. The value of option `timeout` sets the seed value for backoff time when the printer is busy. The default value for the `timeout` option is `10` seconds. The defaults may be reset by:

lpadmin -p *printername* **-o protocol=**
lpadmin -p *printername* `-o bsdctrl=`
lpadmin -p *printername* **-o timeout=**

`Controlling the Use of the Banner Page`

`nobanner`
Allow a user to submit a print request specifying that no banner page be printed.

`banner`
Force a banner page to be printed with every print request, even when a user asks for no banner page. This is the default. Specify `-o nobanner` to allow users to specify `-o nobanner` with the `lp` command. `Undefined Options`

key=value
Each *key=value* is passed directly to the interface program. Any checking for allowed values is done in the interface program.

Any default values for a given *key=value* option are defined in the interface program. If a default is provided, it may be reset by typing the key without any value:

lpadmin -p *printername* **-o** *key***=**

`-P` *paper-name*
Specify a paper type list that the printer supports.

`-r` *class*
Remove *printer* from the specified *class*. If *printer* is the last member of *class*, then *class* will be removed.

`-S` *list*
Allow either the print wheels or aliases for character sets named in *list* to be used on the printer.

If the printer is a type that takes print wheels, then *list* is a comma or space separated list of print wheel names. (Enclose the list with quotes if it contains blank spaces.) These will be the only print wheels considered mountable on the printer. (You can always force a different print wheel to be mounted.) Until the option is used to specify a list, no print wheels will be considered mountable on the printer, and print requests that ask for a particular print wheel with this printer will be rejected.

If the printer is a type that has selectable character sets, then *list* is a comma or blank separated list of character set name ''mappings'' or aliases. (Enclose the list with quotes if it contains blank spaces.) Each ''mapping'' is of the form

known-name=alias The *known-name* is a character set number preceded by `cs` (such as `cs3` for character set three) or a character set name from the `terminfo` database entry `csnm`. See `terminfo`(4). If this option is not used to specify a list, only the names already known from the `terminfo` database or numbers with a prefix of `cs` will be acceptable for the printer. If *list* is the word `none`, any existing print wheel lists or character set aliases will be removed.

Notice the other uses of the `-S` with the `-M` option described above.

The `-T` option must be invoked first with `lpadmin` to identify the printer type before the `-S` option can be used.

`-s` *system-name*[`!`*printer-name*]
Make a remote printer (one that must be accessed through another system) accessible to users on your system. *system-name* is the name of the remote system on which the remote printer is located it. *printer-name* is the name used on the remote system for that printer. For example, if you want to access *printer1* on *system1* and you want it called *printer2* on your system:

`-p` *printer2* `-s` *system1*`!`*printer1*

`-T` *printer-type-list*
Identify the printer as being of one or more *printer-type*s. Each *printer-type* is used to extract data from the `terminfo` database; this information is used to initialize the printer before printing each user's request. Some filters may also use a *printer-type* to convert content for the printer. If this option is not used, the default *printer-type* will be `unknown`; no information will be extracted from `terminfo` so each user request will be printed without first initializing the printer. Also, this option must be used if the following are to work: `-o cpi`, `-o lpi`, `-o width`, and `-o length` options of the `lpadmin` and `lp` commands, and the `-S` and `-f` options of the `lpadmin` command.

If the *printer-type-list* contains more than one type, then the *content-type-list* of the `-I` option must either be specified as `simple`, as empty (`-I ""`), or not specified at all.

`-t` *number-of-trays*
Specify the number of trays when creating the printer.

`-u allow:`*login-ID-list*
`-u deny:`*login-ID-list*
Allow or deny the users in *login-ID-list* access to the printer. By default all users are allowed on a new printer. The *login-ID-list* argument may include any or all of the following constructs:

login-ID	a user on any system
system-name`!`*login-ID*	a user on system *system-name*
system-name`!all`	all users on system *system-name*

`all!`*login-ID*	a user on all systems
`all`	all users on all systems

For each printer, the LP print service keeps two lists of users: an ''allow-list'' of people allowed to use the printer, and a ''deny-list'' of people denied access to the printer. With the `-u allow` option, the users listed are added to the allow-list and removed from the deny-list. With the `-u deny` option, the users listed are added to the deny-list and removed from the allow-list.

If the allow-list is not empty, only the users in the list may use the printer, regardless of the contents of the deny-list. If the allow-list is empty, but the deny-list is not, the users in the deny-list may not use the printer. All users can be denied access to the printer by specifying `-u deny:all`. All users may use the printer by specifying `-u allow:all`.

`-U` *dial-info*
The `-U` option allows your print service to access a remote printer. (It does not enable your print service to access a remote printer service.) Specifically, `-U` assigns the ''dialing'' information *dial-info* to the printer. *dial-info* is used with the `dial` routine to call the printer. Any network connection supported by the Basic Networking Utilities will work. *dial-info* can be either a phone number for a modem connection, or a system name for other kinds of connections. Or, if `-U direct` is given, no dialing will take place, because the name `direct` is reserved for a printer that is directly connected. If a system name is given, it is used to search for connection details from the file `/etc/uucp/Systems` or related files. The Basic Networking Utilities are required to support this option. By default, `-U direct` is assumed.

`-v` *device*
Associate a *device* with *printer*. *device* is the path name of a file that is writable by `lp`. Notice that the same *device* can be associated with more than one printer.

Removing a Printer Destination

The `-x` *dest* option removes the destination *dest* (a printer or a class), from the LP print service. If *dest* is a printer and is the only member of a class, then the class will be deleted, too. If *dest* is `all`, all printers and classes are removed. If there are no remaining local printers and the scheduler is still running, the scheduler is shut down.

No other *options* are allowed with `-x`.

Setting/Changing the System Default Destination

The `-d` [*dest*] option makes *dest* (an existing printer or class) the new system default destination. If *dest* is not supplied, then there is no system default destination. No other *options* are allowed with `-d`.

Setting an Alert for a Print Wheel

`-S` *print-wheel* `-A` *alert-type* [`-W` *minutes*] [`-Q` *requests*]
The `-S` *print-wheel* option is used with the `-A` *alert-type* option to define an alert to mount the print wheel when there are jobs queued for it. If this command is not used to arrange alerting for a print wheel, no alert will be sent for the print wheel. Notice the other use of `-A`, with the `-p` option, above.

The *alert-types* are:

`mail`	Send the alert message using the `mail` command to the administrator.
`write`	Write the message, using the `write` command, to the terminal on which the administrator is logged in. If the administrator is logged in on several terminals, one is arbitrarily chosen.
`quiet`	Do not send messages for the current condition. An administrator can use this option to temporarily stop receiving further messages about a known problem. Once the *print-wheel* has been mounted and subsequently unmounted, messages will again be sent when the number of print requests reaches the threshold specified by the `-Q` option.
`none`	Do not send messages until the `-A` option is given again with a different *alert-type* (other than `quiet`).
shell-command	Run the *shell-command* each time the alert needs to be sent. The shell command should expect the message in standard input. If there are blanks embedded in the command, enclose the command in quotes. Notice that the `mail` and `write` values for this option are equivalent to the values `mail` *user-name* and `write` *user-name* respectively, where *user-name* is the current name for the administrator. This will be the login name of the person submitting this command unless he or she has used the `su` command to change to another user ID. If the `su` command has been used to change the user ID, then the *user-name* for the new ID is used.
`list`	Display the type of the alert for the print wheel on standard output. No change is made to the alert.

The message sent appears as follows:

```
The print wheel print-wheel needs to be mounted
on the printer(s):
printer(integer1requests) integer2 print requests
await this print wheel.
```

The printers listed are those that the administrator had earlier specified were candidates for this print wheel. The number *integer1* listed next to each printer is the number of requests eligible for the printer. The number *integer2* shown after the printer list is the total number of requests awaiting the print wheel. It will be less than the sum of the other numbers if some requests can be handled by more than one printer.

If the *print-wheel* is `all`, the alerting defined in this command applies to all print wheels already defined to have an alert.

If the -W option is not given, the default procedure is that only one message will be sent per need to mount the print wheel. Not specifying the -W option is equivalent to specifying -W once or -W 0. If *minutes* is a number greater than zero, an alert will be sent at intervals specified by *minutes*.

If the -Q option is also given, the alert will be sent when a certain number (specified by the argument *requests*) of print requests that need the print wheel are waiting. If the -Q option is not given, or *requests* is 1 or any (which are both the default), a message is sent as soon as anyone submits a print request for the print wheel when it is not mounted.

EXAMPLES

In the following examples, *prtr* can be any name up to 14 characters and can be the same name as the ping(1M) name.

EXAMPLE 1 Configuring an HP Postscript Printer with a Jet Direct Network Interface

The following example configures an HP postscript printer with a jet direct network interface:

```
example# lpadmin -p prtr -v /dev/null -m netstandard \
          -o dest=ping_name_of_prtr:9100 -o protocol=tcp -T PS -I postscript
example# enable prtr
example# accept prtr
```

EXAMPLE 2 Configuring a Standard Postscript Network Printer

The following example configures a standard postscript network printer:

```
example# lpadmin -p prtr -v /dev/null -m netstandard \
         -o dest=ping_name_of_prtr -T PS -I postscript
example# enable prtr
example# accept prtr
```

EXIT STATUS

The following exit values are returned:

0 Successful completion.

non-zero An error occurred.

FILES

/var/spool/lp/*

/etc/lp

/etc/lp/alerts/printer fault handler for lpadmin.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpcu

SEE ALSO | `enable`(1), `lp`(1), `lpstat`(1), `mail`(1), `stty`(1), `accept`(1M), `lpforms`(1M), `lpsched`(1M), `lpsystem`(1M), `ping`(1M), `dial`(3NSL), `terminfo`(4), `attributes`(5)

System Administration Guide: Basic Administration

NAME

lpfilter – administer filters used with the LP print service

SYNOPSIS

/usr/sbin/lpfilter `-f` *filter-name* {`-` | `-i` | `-l` | `-x` | `-F` *pathname*}

DESCRIPTION

The `lpfilter` command is used to add, change, delete, or list a filter used with the LP print service. These filters convert the content of a file to have a content type acceptable to a printer.

OPTIONS

Arguments consist of the `-f` *filter-name* option and exactly one of the arguments appearing within braces ({ }) in the SYNOPSIS.

–	Adds or changes a filter as specified from standard input. The format of the input is specified below. If `-f all` is specified with the – option, the specified change is made to all existing filters. This is not useful.
`-f` *filter-name*	Specifies the *filter-name* of the filter to be added, changed, reset, deleted, or listed. The filter name `all` is a special filter name defined below. The `-f` option is required.
`-F` *pathname*	Adds or changes a filter as specified by the contents of the file *pathname*. The format of the file's contents is specified below. If `-f all` is specified with the `-F` option, the specified change is made to all existing filters. This is not useful.
`-i`	Resets a filter to its default settings. Using `-f all` with the `-i` option restores all filters for which predefined settings are available to their original settings.
`-l`	Lists a filter description. Using `-f all` with the `-l` option produces a list of all filters.
`-x`	Deletes a filter. Using `-f all` with the `-x` option results in all filters being deleted.

Adding or Changing a Filter

The filter named in the `-f` option is added to the filter table. If the filter already exists, its description is changed to reflect the new information in the input.

When – is specified, standard input supplies the filter description. When `-F` is specified, the file *pathname* supplies the filter description. One of these two options must be specified to add or change a filter.

When an existing filter is changed with the `-F` or – option, lines in the filter description that are not specified in the new information are not changed. When a new filter is added with this command, unspecified lines receive default values. See below.

Filters are used to convert the content of a request from its initial type into a type acceptable to a printer. For a given print request, the LP print service knows the following:

- The content type of the request (specified by `lp -T` or determined implicitly).
- The name of the printer (specified by `lp -d`).

- The printer type (specified by `lpadmin -T`).

  The printer type is intended to be a printer model, but some people specify it with a content type even though `lpadmin -I` is intended for this purpose.
- The content types acceptable to the printer (specified by `lpadmin -I`).

  The values specified by the `lpadmin -T` are treated as if they were specified by the `-I` option as well.
- The modes of printing asked for by the originator of the request (specified by various options to `lp`).

The system uses the above information to construct a list of one or more filters that converts the document's content type into a content type acceptable to the printer and consumes all `lp` arguments that invoke filters (`-y` and `-P`).

The contents of the file (specified by the `-F` option) and the input stream from standard input (specified by –) must consist of a series of lines, such that each line conforms to the syntax specified by one of the seven lines below. All lists are comma or space separated. Each item contains a description.

`Input types:` *content-type-list*
`Output types:` *content-type-list*
`Printer types:` *printer-type-list*
`Printers:` *printer-list*
`Filter type:` *filter-type*
`Command:` *shell-command*
`Options:` *template-list*

`Input types`	This gives the content types that can be accepted by the filter. The default is `any`. The document content type must be a member of this list for the initial filter in the sequence.
`Output types`	This gives the content types that the filter can produce from any of the input (content) types. The default is `any`. The intersection of the output types of this list and the content types acceptable to the printer (from `lpadmin -I` and `lpadmin -T`) must be non-null for the last filter in the sequence. For adjacent filters in the sequence, the intersection of output types of one and the input types of the next must be non-null.
`Printer types`	This gives the printer types for which this printer can be used. The LP print service will restrict the use of the filter to these printer types (from `lpadmin -T`). The default is `any`.
`Printers`	This gives the names of the printers for which the filter can be used. The LP print service will restrict the use of the filter to just the printers named. The default is `any`.
`Filter type`	This marks the filter as a `slow` filter or a `fast` filter. Slow filters are generally those that take a long time to convert their input (that is, minutes or hours). They are run before the job is scheduled

for a printer, to keep the printers from being tied up while the filter is running. If a listed printer is on a remote system, the filter type for it must have the value `slow`. That is, if a client defines a filter, it must be a slow filter. Fast filters are generally those that convert their input quickly (that is, faster than the printer can process the data), or those that must be connected to the printer when run. Fast filters will be given to the interface program to run while connected to the physical printer.

`Command` This specifies which program to run to invoke the filter. The full program pathname as well as fixed options must be included in the *shell-command*; additional options are constructed, based on the characteristics of each print request and on the `Options` field. A command must be given for each filter. The command must accept a data stream as standard input and produce the converted data stream on its standard output. This allows filter pipelines to be constructed to convert data not handled by a single filter.

`Options` This is a comma-separated list of templates used by the LP print service to construct options to the filter from the characteristics of each print request listed in the table later. The `-y` and `-P` arguments to the `lp` command cause a filter sequence to be built even if there is no need for a conversion of content types.

In general, each template is of the following form:

keyword pattern = *replacement*

The *keyword* names the characteristic that the template attempts to map into a filter-specific option; each valid *keyword* is listed in the table below.

A *pattern* is one of the following: a literal pattern of one of the forms listed in the table, a single asterisk (`*`), or a regular expression. If *pattern* matches the value of the characteristic, the template fits and is used to generate a filter-specific option. The *replacement* is what will be used as the option.

Regular expressions are the same as those found on the `regexp`(5) manual page. This includes the `\(...\)` and `\`*n* constructions, which can be used to extract portions of the *pattern* for copying into the *replacement*, and the `&`, which can be used to copy the entire *pattern* into the *replacement*.

The *replacement* can also contain a `*`; it too, is replaced with the entire *pattern*, just like the `&` of `regexp`(5).

The keywords are:

lp Option	Characteristic	*keyword*	Possible *patterns*
-T	Content type (input)	INPUT	content-type
Not applicable	Content type (output)	OUTPUT	content-type
not applicable	Printer type	TERM	printer-type
-d	Printer name	PRINTER	*printer-name*
-f, -o cpi=	Character pitch	CPI	integer
-f, -o lpi=	Line pitch	LPI	integer
-f, -o length=	Page length	LENGTH	integer
-f, -o width=	Page width	WIDTH	integer
-P	Pages to print	PAGES	page-list
-S	Character set Print wheel	CHARSET CHARSET	character-set-name print-wheel-name
-f	Form name	FORM	form-name
-y	Modes	MODES	mode
-n	Number of copies	COPIES	*integer*

Resetting a Filter to Defaults

If the filter named is one originally delivered with the LP print service, the `-i` option restores the original filter description.

Deleting a Filter

The `-x` option is used to delete the filter specified in filter-name from the LP filter table.

Listing a Filter Description

The `-l` option is used to list the description of the filter named in filter-name. If the command is successful, the following message is sent to standard output:

```
Input types: content-type-list
Output types: content-type-list
Printer types: printer-type-list
Printers: printer-list
Filter type: filter-type
Command: shell-command
Options: template-list
```

If the command fails, an error message is sent to standard error.

Large File Behavior

See `largefile`(5) for the description of the behavior of `lpfilter` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXAMPLES

EXAMPLE 1 Printing with the landscape option

For example, the template

```
MODES landscape = -l
```

shows that if a print request is submitted with the `-y landscape` option, the filter will be given the option `-l`.

EXAMPLE 2 Selecting the printer type

As another example, the template

```
TERM * = -T *
```

shows that the filter will be given the option `-T` *printer-type* for whichever *printer-type* is associated with a print request using the filter.

EXAMPLE 3 Using the keywords table

Consider the template

```
MODES prwidth\=\(.*\) = -w\1
```

Suppose a user gives the command

```
lp -y prwidth=10
```

From the table above, the LP print service determines that the `-y` option is handled by a `MODES` template. The `MODES` template here works because the pattern prwidth=) matches the prwidth=10 given by the user. The replacement -w1 causes the LP print service to generate the filter option `-w10`. If necessary, the LP print service will construct a filter pipeline by concatenating several filters to handle the user's file and all the print options. See `sh`(1) for a description of a pipeline. If the print service constructs a filter pipeline, the `INPUT` and `OUTPUT` values used for each filter in the pipeline are the types of input and output for that filter, not for the entire pipeline.

EXIT STATUS

The following exit values are returned:

`0`	Successful completion.
non-zero	An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpsu

SEE ALSO

`lp`(1), `sh`(1), `lpadmin`(1M), `attributes`(5), `largefile`(5), `regexp`(5)

System Administration Guide: Basic Administration

NOTES

If the `lp` command specifies more than one document, the filtering chain is determined by the first document. Other documents may have a different format, but they will print correctly only if the filter chain is able to handle their format.

NAME

lpforms – administer forms used with the LP print service

SYNOPSIS

lpforms -f *form-name option*

lpforms -f *form-name* -A *alert-type* [-P *paper-name* [-d]] [-Q *requests*] [-W *minutes*]

DESCRIPTION

The `lpforms` command administers the use of preprinted forms, such as company letterhead paper, with the LP print service. A form is specified by its *form-name*. Users may specify a form when submitting a print request (see `lp`(1)). The argument `all` can be used instead of *form-name* with either of the command lines shown above. The first command line allows the administrator to add, change, and delete forms, to list the attributes of an existing form, and to allow and deny users access to particular forms. The second command line is used to establish the method by which the administrator is alerted that the form *form-name* must be mounted on a printer.

OPTIONS

The following options are supported:

-f *form-name* Specify a form.

The first form of `lpforms` requires that one of the following *option*s (–, `-l`, `-F`, `-x`) must be used:

-F *pathname* To add or change form *form-name*, as specified by the information in *pathname*.

– To add or change form *form-name*, as specified by the information from standard input.

`-l` To list the attributes of form *form-name*.

`-x` To delete form *form-name* (this option must be used separately; it may not be used with any other option).

The second form of the `lpforms` command requires the `-A` *alert-type* option. The other options are optional.

-A *alert-type* Defines an alert to mount the form when there are queued jobs which need it.

-P *paper-name* [-d] Specify the paper name when creating the form. If `-d` is specified, this paper is the default.

-Q *requests* An alert will be sent when a certain number of print requests that need the form are waiting.

-W *minutes* An alert will be sent at intervals specified by minutes.

Adding or Changing a Form

The `-F` *pathname* option is used to add a new form, *form-name*, to the LP print service, or to change the attributes of an existing form. The form description is taken from *pathname* if the `-F` option is given, or from the standard input if the – option is used. One of these two options must be used to define or change a form.

pathname is the path name of a file that contains all or any subset of the following information about the form.

```
Page length: scaled-decimal-number1
Page width: scaled-decimal-number2
Number of pages: integer
Line pitch: scaled-decimal-number3
Character pitch: scaled-decimal-number4
Character set choice: character-set/print-wheel [mandatory]
Ribbon color: ribbon-color
Comment:
comment
Alignment pattern: [content-type]
content
```

The term ''scaled-decimal-number'' refers to a non-negative number used to indicate a unit of size. The type of unit is shown by a ''trailing'' letter attached to the number. Three types of scaled decimal numbers can be used with the LP print service: numbers that show sizes in centimeters (marked with a trailing `c`); numbers that show sizes in inches (marked with a trailing `i`); and numbers that show sizes in units appropriate to use (without a trailing letter); lines, characters, lines per inch, or characters per inch.

Except for the last two lines, the above lines may appear in any order. The `Comment:` and *comment* items must appear in consecutive order but may appear before the other items, and the `Alignment pattern:` and the *content* items must appear in consecutive order at the end of the file. Also, the *comment* item may not contain a line that begins with any of the key phrases above, unless the key phrase is preceded with a > sign. Any leading > sign found in the *comment* will be removed when the comment is displayed. There is no case distinction among the key phrases.

When this command is issued, the form specified by *form-name* is added to the list of forms. If the form already exists, its description is changed to reflect the new information. Once added, a form is available for use in a print request, except where access to the form has been restricted, as described under the `-u` option. A form may also be allowed to be used on certain printers only.

A description of each form attribute is below:

`Page length` and `Page Width`
: Before printing the content of a print request needing this form, the generic interface program provided with the LP print service will initialize the physical printer to handle pages *scaled-decimal-number1* long, and *scaled-decimal-number2* wide using the printer type as a key into the `terminfo`(4) database. The page length and page width will also be passed, if possible, to each filter used in a request needing this form.

`Number of pages`
: Each time the alignment pattern is printed, the LP print service will attempt to truncate the *content* to a single form by, if possible, passing to each filter the page subset of 1-*integer*.

`Line pitch` and `Character pitch`
Before printing the content of a print request needing this form, the interface program provided with the LP print service will initialize the physical printer to handle these pitches, using the printer type as a key into the `terminfo`(4) database. Also, the pitches will be passed, if possible, to each filter used in a request needing this form. *scaled-decimal-number3* is in lines-per-centimeter if a `c` is appended, and lines-per-inch otherwise; similarly, *scaled-decimal-number4* is in characters-per-centimeter if a `c` is appended, and characters-per-inch otherwise. The character pitch can also be given as `elite` (12 characters-per-inch), `pica` (10 characters-per-inch), or `compressed` (as many characters-per-inch as possible).

`Character set choice`
When the LP print service alerts an administrator to mount this form, it will also mention that the print wheel *print-wheel* should be used on those printers that take print wheels. If printing with this form is to be done on a printer that has selectable or loadable character sets instead of print wheels, the interface programs provided with the LP print service will automatically select or load the correct character set. If `mandatory` is appended, a user is not allowed to select a different character set for use with the form; otherwise, the character set or print wheel named is a suggestion and a default only.

`Ribbon color`
When the LP print service alerts an administrator to mount this form, it will also mention that the color of the ribbon should be *ribbon-color*.

`Comment`
The LP print service will display the *comment* unaltered when a user asks about this form (see `lpstat`(1)).

`Alignment pattern`
When mounting this form, an administrator can ask for the *content* to be printed repeatedly, as an aid in correctly positioning the preprinted form. The optional *content-type* defines the type of printer for which *content* had been generated. If *content-type* is not given, `simple` is assumed. Note that the *content* is stored as given, and will be readable only by the user `lp`.

When an existing form is changed with this command, items missing in the new information are left as they were. When a new form is added with this command, missing items will get the following defaults:

```
Page Length: 66
Page Width: 80
Number of Pages: 1
Line Pitch: 6
Character Pitch: 10
Character Set Choice: any
Ribbon Color: any
```

Deleting a Form

LP print service" The `-x` option is used to delete the form *form-name* from the LP print service.

Listing Form Attributes

The `-l` option is used to list the attributes of the existing form *form-name*. The attributes listed are those described under `Adding and Changing a Form,` above. Because of the potentially sensitive nature of the alignment pattern, only the administrator can examine the form with this command. Other people may use the `lpstat`(1) command to examine the non-sensitive part of the form description.

Allowing and Denying Access to a Form

The `-u` option, followed by the argument `allow:`*login-ID-list* or `-u deny:`*login-ID-list* lets you determine which users will be allowed to specify a particular form with a print request. This option can be used with the `-F` or – option, each of which is described above under `Adding or Changing a Form`.

The *login-ID-list* argument may include any or all of the following constructs:

login-ID	A user on any system
system_name`!`*login-ID*	A user on system *system_name*
system_name`!all`	All users on system *system_name*
`all!`*login-ID*	A user on all systems
`all`	All users on all systems

The LP print service keeps two lists of users for each form: an ''allow-list'' of people allowed to use the form, and a ''deny-list'' of people that may not use the form. With the `-u allow` option, the users listed are added to the allow-list and removed from the deny-list. With the `-u deny` option, the users listed are added to the deny-list and removed from the allow-list. (Both forms of the `-u` option can be run together with the `-F` or the – option.)

If the allow-list is not empty, only the users in the list are allowed access to the form, regardless of the content of the deny-list. If the allow-list is empty but the deny-list is not, the users in the deny-list may not use the form, (but all others may use it). All users can be denied access to a form by specifying `-f deny:all`. All users can be allowed access to a form by specifying `-f allow:all`. (This is the default.)

Setting an Alert to Mount a Form

The `-f` *form-name* option is used with the `-A` *alert-type* option to define an alert to mount the form when there are queued jobs which need it. If this option is not used to arrange alerting for a form, no alert will be sent for that form.

The method by which the alert is sent depends on the value of the *alert-type* argument specified with the `-A` option. The *alert-types* are:

`mail`	Send the alert message using the `mail` command to the administrator.
`write`	Write the message, using the `write` command, to the terminal on which the administrator is logged in. If the administrator is logged in on several terminals, one is arbitrarily chosen.
`quiet`	Do not send messages for the current condition. An administrator can use this option to temporarily stop receiving further messages

about a known problem. Once the form *form-name* has been mounted and subsequently unmounted, messages will again be sent when the number of print requests reaches the threshold specified by the `-Q` option.

`showfault` — Attempt to execute a form alert handler on each system that has a print job for that form in the queue. The fault handler is `/etc/lp/alerts/form`. It is invoked with three parameters: *form_name*, `date`, *file_name*. *file_name* is the name of a file containing the form alert message.

`none` — Do not send messages until the `-A` option is given again with a different *alert-type* (other than `quiet`).

shell-command — Run the *shell-command* each time the alert needs to be sent. The shell command should expect the message in standard input. If there are blank spaces embedded in the command, enclose the command in quotes. Note that the `mail` and `write` values for this option are equivalent to the values `mail` *login-ID* and `write` *login-ID* respectively, where *login-ID* is the current name for the administrator. This will be the login name of the person submitting this command unless he or she has used the `su` command to change to another login-ID. If the `su` command has been used to change the user ID, then the *user-name* for the new ID is used.

`list` — Display the type of the alert for the form on standard output. No change is made to the alert.

The message sent appears as follows:

```
The form form-name needs to be mounted
on the printer(s):printer (integer1 requests).
integer2 print requests await this form.
Use the ribbon-color ribbon.
Use the print-wheel print wheel, if appropriate.
```

The printers listed are those that the administrator has specified as candidates for this form. The number *integer1* listed next to each printer is the number of requests eligible for the printer. The number *integer2* shown after the list of printers is the total number of requests awaiting the form. It will be less than the sum of the other numbers if some requests can be handled by more than one printer. The *ribbon-color* and *print-wheel* are those specified in the form description. The last line in the message is always sent, even if none of the printers listed use print wheels, because the administrator may choose to mount the form on a printer that does use a print wheel.

Where any color ribbon or any print wheel can be used, the statements above will read:

```
Use any ribbon.
Use any print-wheel.
```

If *form-name* is `any`, the *alert-type* defined in this command applies to any form for which an alert has not yet been defined. If *form-name* is `all`, the *alert-type* defined in this command applies to all forms.

If the `-W` *minutes* option is not given, the default procedure is that only one message will be sent per need to mount the form. Not specifying the `-W` option is equivalent to specifying `-W once` or `-W 0`. If *minutes* is a number greater than `0`, an alert will be sent at intervals specified by *minutes*.

If the `-Q` *requests* option is also given, the alert will be sent when a certain number (specified by the argument *requests*) of print requests that need the form are waiting. If the `-Q` option is not given, or the value of *requests* is `1` or `any` (which are both the default), a message is sent as soon as anyone submits a print request for the form when it is not mounted.

Listing the Current Alert

The `-f` option, followed by the `-A` option and the argument `list` is used to list the *alert-type* that has been defined for the specified form *form-name*. No change is made to the alert. If *form-name* is recognized by the LP print service, one of the following lines is sent to the standard output, depending on the type of alert for the form.

- When *requests* `requests are queued: alert with` *shell-command* `every` *minutes* `minutes`
- When *requests* `requests are queued: write to` *user-name* `every` *minutes* `minutes`
- When *requests* `requests are queued: mail to` *user-name* `every` *minutes* `minutes`
- `No alert`

The phrase `every` *minutes* `minutes` is replaced with `once` if *minutes* (`-W`*minutes*) is 0.

Terminating an Active Alert

The `-A quiet` option is used to stop messages for the current condition. An administrator can use this option to temporarily stop receiving further messages about a known problem. Once the form has been mounted and then unmounted, messages will again be sent when the number of print requests reaches the threshold *requests*.

Removing an Alert Definition

No messages will be sent after the `-A none` option is used until the `-A` option is given again with a different *alert-type*. This can be used to permanently stop further messages from being sent as any existing alert definition for the form will be removed.

Large File Behavior

See `largefile`(5) for the description of the behavior of `lpforms` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXIT STATUS

The following exit values are returned:

`0`	Successful completion.
non-zero	An error occurred.

FILES

`/etc/lp/alerts/form`	Fault handler for `lpform`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpsu

SEE ALSO

`lp`(1), `lpstat`(1), `lpadmin`(1M), `terminfo`(4), `attributes`(5), `largefile`(5)

System Administration Guide: Basic Administration

NAME

lpget – get printing configuration

SYNOPSIS

lpget [-k *key*] [*destination*... | list]

DESCRIPTION

The lpget utility reads printing configuration information from the configuration databases in $HOME/.printers, /etc/printers.conf, printers.conf.byname, printers.org_dir, and FNS printer contexts. This information, called a *configuration report*, is displayed to the standard output. See printers(4) and printers.conf(4) for information about the printer configuration databases.

lpget displays a configuration report for all keys for the specified destination or destinations by default. Use the -k option to display a configuration report for specific keys. Use the list operand to display a configuration report for all configured destinations.

OPTIONS

The following option is supported:

-k *key* Displays a configuration report for *key*. See printers.conf(4) for information about specifying *key*.

OPERANDS

The following operands are supported:

destination Displays a configuration report for *destination*. Destination can be either a printer of a class of printers, (see lpadmin(1M)). Specify *destination* using atomic, POSIX-style (*server*:*destination*), or Federated Naming Service (FNS) (. . ./service/printer/. . .) names. See printers.conf (4) for information regarding the naming conventions for atomic and FNS names, and standards(5) for information concerning POSIX.

list Displays a configuration report for all configured destinations.

EXAMPLES

EXAMPLE 1 Displaying a Configuration Report for the bsdaddr Key

The following example displays a configuration report for the bsdaddr key for printer catalpa.

```
example% lpget -k bsdaddr catalpa
```

EXAMPLE 2 A Configuration Report for all Keys for all Configured Destinations

The following example displays a configuration report for all keys for all configured destinations.

```
example% lpget list
```

EXIT STATUS

The following exit values are returned:

0 Successful completion.

non-zero An error occurred.

FILES

`/etc/printers.conf`
System printer configuration database.

`$HOME/.printers`
User-configurable printer database.

`printers.conf.byname`
NIS version of `/etc/printers.conf`.

`printers.org_dir`
NIS+ version of `/etc/printers.conf`.

`fns.ctx_dir.`*domain*
FNS version of `/etc/printers.conf`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpcu
Stability Level	Stable

SEE ALSO

`ldap`(1), `lp`(1), `lpc`(1B), `lpq`(1B), `lpr`(1B), `lpstat`(1), `lpadmin` (1M), `lpset`(1M), `printers`(4), `printers.conf`(4), `attributes`(5), `standards`(5)

System Administration Guide: Naming and Directory Services

NOTES

Be mindful of the following if the LDAP database is used as the name service. If the `ldapclient`(1M) server is a replica LDAP server, LDAP printer database updates may not appear immediately, as the replica server may not not have been updated by the master server and can be out of sync. For example, a printer that you deleted by using `lpset`(1M) may still appear in the printer list you display with `lpget` until the replica is updated from the master. Replica servers vary as to how often they are updated from the master. Refer to the *System Administration Guide: Naming and Directory Services* for more information on LDAP replication.

NAME

lpmove – move print requests

SYNOPSIS

lpmove [*request-ID*] *destination*

lpmove *destination1 destination2*

DESCRIPTION

The `lpmove` command moves print requests queued by `lp`(1) or `lpr`(1B) between destinations.

The first form of `lpmove` moves specific print requests (*request-ID*) to a specific *destination*.

The second form of the `lpmove` command moves all print requests from one destination (*destination1*) to another (*destination2*). This form of `lpmove` also rejects new print requests for *destination1*.

`lpmove` moves individual requests or entire queues only between local printers or between remote printers, not between a local and a remote printer. You can move only requests that were not previously transferred to the server.

When moving requests, `lpmove` does not check the acceptance status of the destination to which the print requests are being moved (see `accept`(1M)). `lpmove` does not move requests that have options (for example, content type or requiring a special form) that cannot be handled by the new destination.

OPERANDS

The following operands are supported:

request-ID — The specific print request to be moved. Specify *request-ID* as the identifier associated with a print request as reported by `lpstat`. See `lpstat`(1).

destination — The name of the printer or class of printers (see `lpadmin`(1M)) to which `lpmove` moves a *specified* print request. Specify *destination* using atomic, POSIX-style (*server*`:`*destination*) syntax.

destination1 — The name of the destination from which `lpmove` moves *all* print requests. Specify *destination* using atomic, POSIX-style (*server*`:`*destination*) syntax.

destination2 — The name of the destination to which `lpmove` moves all print requests. Specify *destination* using atomic, POSIX-style (*server*`:`*destination*) syntax.

See `printers.conf`(4) for information regarding the naming conventions for atomic names and `standards`(5) for information regarding POSIX.

EXIT STATUS

The following exit values are returned:

`0` — Successful completion.

non-zero — An error occurred.

FILES

`/var/spool/print/*` — LP print queue.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpcu

SEE ALSO

`lp`(1), `lpr`(1B), `lpstat`(1), `accept`(1M), `lpadmin`(1M), `lpsched`(1M), `printers.conf`(4), `attributes`(5), `standards`(5)

System Administration Guide: Advanced Administration

NAME

lpsched – start the LP print service

SYNOPSIS

lpsched [-f *num_filters*] [-n *num_notifiers*] [-p *fd_limit*] [-r *reserved_fds*]

DESCRIPTION

The `lpsched` command starts or restarts the LP print service.

The `lpshut` command stops the LP print service. Printers that are restarted using `lpsched` reprint (in their entirety) print requests that were stopped by `lpshut`. See `lpshut`(1M).

OPTIONS

The following options are supported:

`-f` *num_filters* — Specifies the number of concurrent slow filters that may be run on a print server. A default value of `1` is used if none is specified. Depending on server configuration, a value of `1` may cause printers to remain idle while there are jobs queued to them.

`-n` *num_notifiers* — Specifies the number of concurrent notification processes that can run on a print server. A default value of `1` is used when none is specified.

`-p` *fd_limit* — Specifies the file descriptor resource limit for the `lpsched` process. A default value of `4096` is used if none is specified. On extremely large and active print servers, it may be necessary to increase this value.

`-r` *reserved_fds* — Specifies the number of file descriptors that the scheduler reserves for internal communications under heavy load. A default value of `2` is used when none is specified. It should not be necessary to modify this value unless instructed to do so when troubleshooting problems under high load.

EXIT STATUS

The following exit values are returned:

`0` — Successful completion.

non-zero — An error occurred.

FILES

`/var/spool/lp/*` — LP print queue.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpsu

SEE ALSO

`lp`(1), `lpstat`(1), `lpadmin`(1M), `lpmove`(1M), `lpshut`(1M), `attributes`(5)

System Administration Guide: Basic Administration

NAME

lpset – set printing configuration in /etc/printers.conf or other supported databases

SYNOPSIS

```
lpset [-n system | nisplus | fns | ldap] [-x] [ [-D binddn] [-w
     passwd] [-h ldaphost]] [-a key=value] [-d key] destination
```

DESCRIPTION

The `lpset` utility sets printing configuration information in the system configuration databases. Use `lpset` to create and update printing configuration in `/etc/printers.conf`, `printers.org_dir` (NIS+), or Federated Naming System (FNS). See `nsswitch.conf`(4), `printers.conf`(4), and `fns`(5).

Only a superuser or a member of Group 14 may execute `lpset`.

OPTIONS

The following options are supported:

`-n system|nisplus|fns|ldap`
Create or update the configuration information for the *destination* entry in `/etc/printers.conf`, `printers.org_dir` (NIS+), LDAP or FNS printer contexts. `system` specifies that the information is created or updated in `/etc/printers.conf`. `nisplus` specifies that the information is created or updated in the `printers.org_dir` NIS+ table. `ldap` specifies that the information is written to an LDAP server. See NOTES. `fns` specifies that the information is written using federated naming context.

If `-n` is not specified, `system` is the default.

`-x`
Remove all configuration for the *destination* entry from the database specified by the `-n` option.

`-a` *key=value*
Configure the specified *key=value* pair for the *destination*. See `printers.conf`(4) for information regarding the specification of *key=value* pairs.

`-d` *key*
Delete the configuration option specified by *key* for the *destination* entry. See `printers.conf`(4) for information regarding the specification of *key* and *key=value* pairs.

`-D binddn`
Use the distinguished name (DN) `binddn` to bind to the LDAP directory server.

`-w passwd`
Use `passwd` as the password for authentication to the LDAP directory server.

`-h ldaphost`
Specify an alternate host on which the LDAP server is running. This option is only used when `ldap` is specified as the naming service. If this option is not specified, the default is the current host system.

OPERANDS

The following operand is supported:

destination Specifies the entry in `/etc/printers.conf`, `printers.org_dir`, LDAP, or FNS, in which to create or modify

information. *destination* names a printer of class of printers. See `lpadmin`(1M). Each entry in `printers.conf` describes one destination. Specify *destination* using atomic or Federated Naming Service (FNS) (`.../service/printer/...`) names. POSIX-style destination names are not acceptable. See `printers.conf`(4) for information regarding the naming conventions for atomic and FNS names and `standards`(5) for information regarding POSIX.

EXAMPLES

EXAMPLE 1 Removing All Existing Printing Configuration Information

The following example removes all existing printing configuration information for destination `dogs` from `/etc/printers.conf`:

```
example% lpset -x dogs
```

EXAMPLE 2 Setting a `key=value` Pair

The following example sets the `user-equivalence =true` *key=value* pair for destination `tabloid` in FNS context:

```
example% lpset -n fns -a user-equivalence=true tabloid
```

EXAMPLE 3 Setting a `key=value` Pair in LDAP

```
example% lpset -n ldap -h ldapl.xyz.com -D "cn=Directory Manager" \
   -w passwd -a key1=value1 printer1
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

non-zero An error occurred.

FILES

`/etc/printers.conf`
System configuration database.

`printer.org_dir` (NIS+)
NIS+ version of `/etc/printers.conf`.

`fns.ctx_dir.`*domain*
FNS version of `/etc/printers.conf`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpcu
Stability Level	Stable

SEE ALSO `ldap`(1), `lp`(1), `lpc`(1B), `lpq`(1B), `lpr`(1B), `lpstat`(1), `ldapclient`(1M), `lpadmin`(1M), `lpget`(1M), `nsswitch.conf`(4), `printers`(4), `printers.conf`(4), `attributes`(5), `fns`(5), `standards`(5)

System Administration Guide: Naming and Directory Services

NOTES If the `ldap` database is used, the printer administrator should be mindful of the following when updating printer information.

1. Because the domain information for the printer being updated is extracted from the `ldapclient`(1M) configuration, the LDAP server being updated must host the same domain that is used by the current `ldapclient`(1M) server.
2. If the LDAP server being updated is a replica LDAP server, the updates will be referred to the master LDAP server and completed there. The updates might be out of sync and not appear immediatedly, as the replica server may not have been updated by the master server. For example, a printer that you deleted by using `lpset` may still appear in the printer list you display with `lpget` until the replica is updated from the master. Replica servers vary as to how often they are updated from the master. See *System Administration Guide: Advanced Administration* for information on LDAP server replication.
3. Although users can use the LDAP command line utilities `ldapadd`(1) and `ldapmodify`(1) to update printer entries in the directory, the preferred method is to use `lpset`. Otherwise, if the `ldapadd` and `ldapmodify` utilities are used, the administrator must ensure that the `printer-name` attribute value is unique within the `ou=printers` container on the LDAP server. If the value is not unique, the result of modifications done using `lpset` or the Solaris Print Manager, `printmgr`(1M) may be unpredictable.

NAME

lpshut – stop the LP print service

SYNOPSIS

`lpshut`

DESCRIPTION

The `lpshut` command stops the LP print service.

Printers that are printing when `lpshut` is invoked stop printing. Start or restart printers using `lpsched`(1M).

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

non-zero An error occurred.

FILES

`/var/spool/lp/*` LP print queue.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpsu

SEE ALSO

`lp`(1), `lpstat`(1), `lpadmin`(1M), `lpmove`(1M), `lpsched`(1M), `attributes`(5)

System Administration Guide: Basic Administration

NAME

lpsystem – register remote systems with the print service

DESCRIPTION

The `lpsystem` command is obsolete, and could be removed at any time. The print system no longer uses the information generated by `lpsystem`. See `lpadmin`(1M), `lpusers`(1M) or `printers.conf`(4) for equivalent functionality.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpcu
Stability Level	Obsolete*

* This command could be removed at any time.

SEE ALSO

`lpadmin`(1M), `lpusers`(1M), `printers.conf`(4), `attributes`(5)

NAME

lpusers – set printing queue priorities

SYNOPSIS

lpusers -d *priority-level*

lpusers -q *priority-level* -u *login-ID-list*

lpusers -u *login-ID-list*

lpusers -q *priority-level*

lpusers -l

DESCRIPTION

The `lpusers` command sets limits to the queue priority level that can be assigned to jobs submitted by users of the LP print service.

The first form of the command (with `-d`) sets the system-wide priority default to *priority-level*, where *priority-level* is a value of 0 to 39, with 0 being the highest priority. If a user does not specify a priority level with a print request (see `lp`(1)), the default priority level is used. Initially, the default priority level is 20.

The second form of the command (with `-q` and `-u`) sets the default highest *priority-level* (0-39) that the users in *login-ID-list* can request when submitting a print request. The *login-ID-list* argument may include any or all of the following constructs:

login-ID	A user on any system
system_name`!`*login-ID*	A user on the system *system_name*
system_name`!all`	All users on system *system_name*
`all!`*login-ID*	A user on all systems
`all`	All users on all systems

Users that have been given a limit cannot submit a print request with a higher priority level than the one assigned, nor can they change a request that has already been submitted to have a higher priority. Any print requests submitted with priority levels higher than allowed will be given the highest priority allowed.

The third form of the command (with `-u`) removes any explicit priority level for the specified users.

The fourth form of the command (with `-q`) sets the default highest priority level for all users not explicitly covered by the use of the second form of this command.

The last form of the command (with `-l`) lists the default priority level and the priority limits assigned to users.

OPTIONS

The following options are supported:

`-d` *priority-level*
Set the system-wide priority default to *priority-level*.

`-l`
List the default priority level and the priority limits assigned to users.

`-q` *priority-level*
Set the default highest priority level for all users not explicitly covered.

`-q` *priority-level* `-u` *login-ID-list*
Set the default highest *priority-level* that the users in *login-ID-list* can request when submitting a print request.

`-u` *login-ID-list*
Remove any explicit priority level for the specified users.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

non-zero An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpsu

SEE ALSO

`lp`(1), `attributes`(5)

NAME

lu – FMLI-based interface to Live Upgrade functions

SYNOPSIS

/usr/sbin/lu

DESCRIPTION

The `lu` program is part of a suite of commands that make up the Live Upgrade feature of the Solaris operating environment. See `live_upgrade`(5) for a description of the Live Upgrade feature.

The `lu` program is a Forms and Menu Language Interpreter-based user interface. (See `fmli`(1) for a description of the Forms and Menu Language Interpreter.) `lu` enables you to create and upgrade boot environments (BEs) and perform other administrative tasks on BEs. The `lu` program performs a subset of the functions provided by the Live Upgrade command-line utilities.

The output of the `lu` command is not internationalizable. The command and its FMLI-based interface is subject to change in a future release.

Invocation of the `lu` command requires root privileges.

The `lu` command accepts no arguments. After invoking `lu`, you receive a display with the following options:

Activate	Activate a boot environment. This option designates that the system boot from the specified BE upon next reboot. This option is equivalent to the command-line `luactivate`(1M) utility.
Cancel	Cancel a copy job. Live Upgrade allows you to schedule the copy, upgrade, and flash functions (all described below) at a later time. The cancel function enables you to cancel a scheduled job. This function is equivalent to the command-line `lucancel`(1M) utility.
Compare	Compare the contents of BEs. Enables you to obtain a detailed comparison of two BEs. Equivalent to the command-line `lucompare`(1M) utility.
Copy	Start/schedule a copy. Copies the contents of one BE to another. Equivalent of the command-line `lumake`(1M) utility. At any time, you can have only one Live Upgrade operation scheduled.
Create	Create a boot environment. Implements a subset of the functions performed by the command-line `lucreate`(1M) utility.
Current	Display the name of the current boot environment. Equivalent of the command-line `lucurr`(1M) utility.
Delete	Delete a boot environment. Equivalent of the command-line `ludelete`(1M) utility.
List	List the file systems of a boot environment. Equivalent of the command-line `lufslist`(1M) utility.
Rename	Change the name of a boot environment. Equivalent of the command-line `lurename`(1M) utility.

Status	List the status of all boot environments. Equivalent of the command-line `lustatus`(1M) utility.
Upgrade	Upgrade a boot environment or upgrade the OS on an inactive BE. This option enables you to upgrade to a new operating system or install new packages or patches on a specified BE. Implements a subset of the functions performed by the command-line `luupgrade`(1M) utility.
Flash	Flash a boot environment. This option enables you to install an operating system on a BE from a flash archive. You can perform the same function with `luupgrade`(1M).
Help	Displays help information. There are also context-specific help screens for many of the options.
Exit	Exit `lu`.

Navigation

You navigate through `lu`'s various screens using arrow keys and function keys (usually F2 through F9 on the keyboard of a Sun desktop system). Available key functions are displayed at the base of the `lu` screen. You can use Ctrl-F, plus a number key, to duplicate a function key. For example, press Ctrl-F and the number key `2` to duplicate the F2 key.

In a screen for a given option, you can press Esc to obtain context-specific help.

Display Issues

When viewing the FMLI interface remotely, such as over a `tip` line, you might need to set the `TERM` environment variable to `VT220`. When using the FMLI interface in a CDE environment use `dtterm`, rather than `xterm`, as the value of the `TERM` variable.

The `lu` command supports only single-byte environments.

Common Functions

Most of the options listed above offer the following functions. These functions are accessible through function keys indicated at the base of the screen.

Choice	Available to you whenever you have a field that can be filled in. Pressing the Choice function key gives you a popup screen displaying a list of alternatives. For example, for options involving copying or upgrading BEs, you receive a list of available BEs. You can then use arrow and function keys to make a selection from this popup. The choice function is useful because it prevents you from selecting an invalid alternative. In our example, it prevents you from choosing a BE that is not available for a copy or upgrade operation. Such non-availability might occur when a BE is in the midst of an upgrade.
Cancel	Cancel an operation.
Save	Proceed with an operation.

Other Functions

The "Create" option, described above, offers the following functions:

Split	Split a file system. For example, you can split a `/` file system into `/`, `/usr`, and `/var`. To split a file system, you must have disk slices available on which

to mount the separated file system(s). If you do not, `lu` invokes the `format`(1M) utility, in which you can use the `partition` option to create a new disk slice.

Merge Join one or more file systems with its (or their) parent file system. For example, using a source BE that has separate `/`, `/usr`, and `/var` file systems, you can merge these file systems under `/` on a target BE.

FILES

`/etc/lutab` list of BEs on the system

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWluu

SEE ALSO

`luactivate`(1M), `lucancel`(1M), `lucompare`(1M), `lucreate`(1M), `lucurr`(1M), `ludelete`(1M), `ludesc`(1M), `lufslist`(1M), `lumake`(1M), `lumount`(1M), `lurename`(1M), `lustatus`(1M), `luupgrade`(1M), `lutab`(4), `attributes`(5), `live_upgrade`(5)

Solaris Installation Guide

NAME

luactivate – activate a boot environment

SYNOPSIS

/usr/sbin/luactivate [-l *error_log*] [-o *outfile*] [-s] [*BE_name*] [-X]

DESCRIPTION

The `luactivate` command is part of a suite of commands that make up the Live Upgrade feature of the Solaris operating environment. See `live_upgrade`(5) for a description of the Live Upgrade feature.

The `luactivate` command, with no arguments, displays the name of the boot environment (BE) that will be active upon the next reboot of the system. When an argument (a BE) is specified, `luactivate` activates the specified BE.

`luactivate` activates a BE by making the BE's root partition bootable. On an IA machine, this might require that you take steps following the completion of `luactivate`. If so, `luactivate` displays the correct steps to take.

To successfully activate a BE, that BE must meet the following conditions:

- The BE must have a status of "complete," as reported by `lustatus`(1M).
- If the BE is not the current BE, you cannot have mounted the partitions of that BE (using `lumount`(1M) or `mount`(1M)).
- The BE you want to activate cannot be involved in an `lucompare`(1M) operation.

After activating a specified BE, `luactivate` displays the steps to be taken for fallback in case of any problem on the next reboot. Make note of these instructions and follow them exactly, if necessary.

The first time you boot from a newly created BE, Live Upgrade software synchronizes this BE with the BE that was last active. (This is not necessarily the BE that was the source for the newly created BE.) "Synchronize" here means that certain system files and directories are copied from the last-active BE to the BE being booted. Live Upgrade software does not perform this synchronization after a BE's initial boot, unless you use the `-s` option, described below.

If `luactivate` detects conflicts between files that are subject to synchronization, it issues a warning and does not perform the synchronization for those files. Activation can complete successfully, in spite of such a conflict. A conflict can occur if you upgrade one BE or another to a new operating system version or if you modify system files (for example, `/etc/passwd`) on one of the BEs.

The `luactivate` command requires root privileges.

OPTIONS

The `luactivate` command has the following options:

`-l` *error_log* — Error and status messages are sent to *error_log*, in addition to where they are sent in your current environment.

`-o` *outfile* — All command output is sent to *outfile*, in addition to where it is sent in your current environment.

`-s` Causes synchronization to occur (see DESCRIPTION) even if next boot of a specified BE is not the first boot of that BE. Use this option with great caution, because you might not be aware or in control of changes that might have occurred in the last-active BE.

If using `-s`, take special care when booting to an earlier release of Solaris than what is installed on the last-active BE. For example, consider that the last-active BE contains Solaris 9 and you want to activate a BE that contains Solaris 2.6. If you forced synchronization with the `-s` option, the BE containing Solaris 2.6 might be synchronized with files that, while compatible with Solaris 9, might not work under Solaris 2.6.

`-X` Enable XML output. Characteristics of XML are defined in DTD, in `/usr/share/lib/xml/dtd/lu_cli.dtd.`<*num*>, where <*num*> is the version number of the DTD file.

OPERANDS

BE_name Name of the BE to be activated.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

FILES

`/etc/lutab`
list of BEs on the system

`/usr/share/lib/xml/dtd/lu_cli.dtd.`<*num*>
Live Upgrade DTD (see `-X` option)

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWluu

SEE ALSO

`lu`(1M), `lucancel`(1M), `lucompare`(1M), `lucreate`(1M), `lucurr`(1M), `ludelete`(1M), `ludesc`(1M), `lufslist`(1M), `lumake`(1M), `lumount`(1M), `lurename`(1M), `lustatus`(1M), `luupgrade`(1M), `lutab`(4), `attributes`(5), `live_upgrade`(5)

NAME

lucancel – cancel a scheduled Live Upgrade copy/create procedure

SYNOPSIS

/usr/sbin/lucancel [-l *error_log*] [-o *outfile*] [-X]

DESCRIPTION

The `lucancel` command is part of a suite of commands that make up the Live Upgrade feature of the Solaris operating environment. See `live_upgrade`(5) for a description of the Live Upgrade feature.

The `lucancel` command cancels a boot environment (BE) creation or upgrade that was scheduled in the FMLI-based interface, `lu`(1M), or the repopulation of a BE, scheduled with `lumake`(1M). `lucancel` does not cancel a job that is active (that is, is in the process of creation or repopulation).

The `lucancel` command requires root privileges.

OPTIONS

The `lucancel` command has the following options:

`-l` *error_log* Error and status messages are sent to *error_log*, in addition to where they are sent in your current environment.

`-o` *outfile* All command output is sent to *outfile*, in addition to where it is sent in your current environment.

`-X` Enable XML output. Characteristics of XML are defined in DTD, in `/usr/share/lib/xml/dtd/lu_cli.dtd.`<*num*>, where <*num*> is the version number of the DTD file.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

FILES

`/etc/lutab` list of BEs on the system

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWluu

SEE ALSO

`lu`(1M), `luactivate`(1M), `lucompare`(1M), `lucreate`(1M), `lucurr`(1M), `ludelete`(1M), `ludesc`(1M), `lufslist`(1M), `lumake`(1M), `lumount`(1M), `lurename`(1M), `lustatus`(1M), `luupgrade`(1M), `lutab`(4), `attributes`(5), `live_upgrade`(5)

NAME

lucompare – compare boot environments

SYNOPSIS

/usr/sbin/lucompare [-i *infile* | -t] [-o *outfile*] *BE_name* [-X]

/usr/sbin/lucompare [-C *file* [-o *outfile*]] [-X]

DESCRIPTION

The `lucompare` command is part of a suite of commands that make up the Live Upgrade feature of the Solaris operating environment. See `live_upgrade`(5) for a description of the Live Upgrade feature.

The `lucompare` command compares the contents of the current boot environment (BE) with the contents of another BE. With the `-C` option, `lucompare` compares file statistics so that you can determine which files have changed on a BE since a specified time, such as the creation time of a BE. A specified BE must be inactive and in the complete state, as reported by the `lustatus`(1M) command. Also, a BE cannot have a copy job scheduled, which is also reported by `lustatus`(1M). A specified BE cannot have any partitions mounted with `lumount`(1M) or `mount`(1M).

For each file system defined for a specified BE, `lucompare` compares all files with the files with the same pathnames in the current BE. The files present in the active BE, but not in the specified BE, and vice-versa, are reported. You also have the option to specify a list of files to be compared.

If you specify the `-C` option, instead of doing an absolute comparison of the current BE with a target BE, `lucompare` compares the files in a specified BE with the list of files recorded in a file. When a BE is created, `lucreate`(1M) creates a file named `:`<*BE_name*> in `/etc/lu/compare`. You can use the `-C` option to compare the files in a specified BE to this snapshot in `/etc/lu/compare` or you can compare the BE to a file previously created with the `-o` option. Comparing a BE to its own snapshot in `/etc/lu/compare` enables you to determine which files have changed on the BE since its creation.

By default, the output of `lucompare` is written to stdout. With the `-C` option, you must use the `-o` option to specify an output file. The output for `lucompare` is a list of files that differ in permissions, owner, group, or sum, along with the reason for difference. The output format is shown below:

```
> active BE
< BE_name
reason
> file_name:owner:group:number_of_links:mode:type: size
or major_minor number:checksum
< file_name:owner:group:number_of_links:mode:type: size
or major_minor number:checksum
```

The above fields are obtained from the `stat`(2) structure of the file.

The *type* field can be one of the following:

`SYMLINK`	symbolic link
`FIFO`	FIFO file

CHRSPC	character special
BLKSPC	block special
DIR	directory
REGFIL	regular file
UNKNOW	unknown file type

`lucompare` computes checksums only if the file on the specified BE matches its counterpart on the active BE in all of the fields described above. If the checksums differ, `lucompare` appends the differing checksums to the entries for the compared files.

The `lucompare` command requires root privileges.

OPTIONS

The `lucompare` command has the following options:

`-C` *file*	Compare file statistics of BE with those recorded in *file*. *file* can be the snapshot created at BE creation time, `/etc/lu/compare/:`<*BE_name*>, or a file previously created with the `-o` option. You must use the `-o` option with this option.
`-i` *infile*	Compare files listed in *infile*. The files to be compared should be an absolute filename. If the entry in the file is a directory, then comparison is recursive with respect to the directory. Mutually exclusive of `-t`.
`-o` *outfile*	Send output of differences to *outfile*. You must use this option if you use `-C`.
`-t`	Compare only nonbinary files. This is achieved by performing a `file`(1) command on each file in the tree walk and only comparing text files. Mutually exclusive of `-i`.
`-X`	Enable XML output. Characteristics of XML are defined in DTD, in `/usr/share/lib/xml/dtd/lu_cli.dtd.`<*num*>, where <*num*> is the version number of the DTD file.

OPERANDS

BE_name	Name of the BE to which the active BE will be compared. You cannot specify a BE that is involved in another Live Upgrade operation, or specify a BE for which you have mounted partitions (using `lumount`(1M) or `mount`(1M)).

EXAMPLES

EXAMPLE 1 Checking Differences Since BE Creation

The following command lists the differences in the BE `s8u5` between its creation time and the present.

EXAMPLE 1 Checking Differences Since BE Creation *(Continued)*

```
# lucompare -C /etc/lu/compare/:s8u5 -o /var/tmp/compare.out s8u5
```

Note that `/etc/lu/compare/:s8u5` is the file created by `lucreate` upon creation of a BE. The list of differences is sent to `/var/tmp/compare.out`.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

FILES

`/etc/lutab`
list of BEs on the system

`/usr/share/lib/xml/dtd/lu_cli.dtd.`*<num>*
Live Upgrade DTD (see `-X` option)

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWluu

SEE ALSO

`lu`(1M), `luactivate`(1M), `lucancel`(1M), `lucreate`(1M), `lucurr`(1M), `ludelete`(1M), `ludesc`(1M), `lufslist`(1M), `lumake`(1M), `lumount`(1M), `lurename`(1M), `lustatus`(1M), `luupgrade`(1M), `lutab`(4), `attributes`(5), `live_upgrade`(5)

NOTES

The `lucompare` command makes no attempt to reconcile any differences it detects between BEs.

NAME

lucreate – create a new boot environment

SYNOPSIS

```
/usr/sbin/lucreate [-A BE_description] [-c BE_name] [-C ( boot_device | - )] -n BE_name [-l error_log] [-o outfile] [-s ( - | source_BE_name )] [ [-M slice_list] [-m mountpoint:device:fs_type [-m...]]] [-X]
```

DESCRIPTION

The `lucreate` command is part of a suite of commands that make up the Live Upgrade feature of the Solaris operating environment. See `live_upgrade`(5) for a description of the Live Upgrade feature and its associated terminology.

The `lucreate` command offers a set of command line options that enable you to perform the following functions:

- Create a new boot environment (BE), based on the current BE.
- Create a new BE, based on a BE other than the current BE.
- Join or separate the file systems of a BE onto a new BE. For example, join `/var` and `/opt` under `/`, or separate these directories to be mounted under different disk slices.
- Create the file systems for a BE, but leave those file systems unpopulated.

You can perform the preceding functions using only `lucreate` command-line options or you can omit the `-m` and `-M` options (described below), which automatically invokes an FMLI-based interface that provides curses-based screens for Live Upgrade administration.

The creation of a BE includes selecting the disk or device slices for all the mount points of the BE. You can also change the mount points of the BE using the SPLIT and MERGE functions of the FMLI-based configuration screen.

Upon successful creation of a BE, you can use `lustatus`(1M) to view the state of that BE and `lufslist`(1M) to view the BE's file systems. You use `luupgrade`(1M) to upgrade the OS on that BE and `luactivate`(1M) to make a BE active, that is, designate it as the BE to boot from at the next reboot of the system.

The `lucreate` command makes a distinction between the file systems that contain the OS—`/`, `/usr`, `/var`, and `/opt`—and those that do not, such as `/export`, `/home`, and other, user-defined file systems. The file systems in the first category cannot be shared between the source BE and the BE being created; they are always copied from the source BE to the target BE. By contrast, the user-defined file systems are shared by default. For Live Upgrade purposes, the file systems that contain the OS are referred to as non-shareable (or *critical*) file systems; other file systems are referred to as shareable. A non-shareable file system listed in the source BE's `vfstab` is always copied to a new BE. For a shareable file system, if you specify a destination slice, the file system is copied. If you do not, the file system is shared.

Except for a special use of the `-s` option, described below, you must have a source BE for the creation of a new BE. By default, it is the current BE. You can use the `-s` option to specify a BE other than the current BE.

By default, all swap partitions on a source BE are shared between the source and target BE. You can use the `-m` option (see below) to specify a subset of swap partitions on a source BE for sharing with a target BE.

The `lucreate` command allows you to assign a description to a BE. A description is an optional attribute of a BE that can be of any format or length. It might be, for example, a text string or binary data. After you create a BE, you can change a BE description with the `ludesc`(1M) utility.

The `lucreate` command requires root privileges.

OPTIONS

The `lucreate` command has the options listed below. Note that a BE name must not exceed 30 characters in length and must consist only of alphanumeric characters and other ASCII characters that are not special to the Unix shell. See the "Quoting" section of `sh`(1). The BE name can contain only single-byte, 8–bit characters; it cannot contain whitespace characters.

Omission of `-m` or `-M` options (described below) in an `lucreate` command line invokes the FMLI-based interface, which allows you to select disk or device slices for a BE.

`-A` *BE_description*
Assigns the *BE_description* to a BE. *BE_description* can be a text string or other characters that can be entered on a Unix command line. See `ludesc`(1M) for additional information on BE descriptions.

`-c` *BE_name*
Assigns the name *BE_name* to the current BE. This option is required only when the first BE is created. For the first time you run `lucreate`, if you omit `-c` you are prompted to name the current BE. If you use the `-c` option following the first BE creation, you receive an error message.

`-C` (*boot_device* | `–`)
Required when you have a mirrored root device on the source BE. Specifies the physical boot device from which the source BE is booted. Without this option, `lucreate` attempts to determine the physical device from which a BE boots. If the device on which the root file system is located is not a physical disk (for example, if root is on a metadevice) and `lucreate` is able to make a reasonable guess as to the physical device, you receive the query:

```
Is the physical device devname the boot device for
the logical device devname?
```

If you respond `y`, the command proceeds.

If you specify `-C` *boot_device*, `lucreate` skips the search for a physical device and uses the device you specify. The – (hyphen) with the `-C` option tells `lucreate` to proceed with whatever it determines is the boot device. If the command cannot find the device, you are prompted to enter it.

If you omit `-C` or specify `-C` *boot_device* and `lucreate` cannot find a boot device, you receive an error message.

Use of the `-C` – form is a safe choice, because `lucreate` either finds the correct boot device or gives you the opportunity to specify that device in response to a subsequent query.

`-l` *error_log*
Error messages and other status messages are sent to *error_log*, in addition to where they are sent in your current environment.

`-m` *mountpoint*:*device*:*fs_type*
[`-m` *mountpoint*:*device*:*fs_type*] ...
Specifies the `vfstab`(4) information for the new BE. The file systems specified as arguments to `-m` can be on the same disk or can be spread across multiple disks.

mountpoint can be any valid mount point or – (hyphen), indicating a swap partition. The *device* field can be one of the following:

- The name of a disk slice, of the form `/dev/dsk/c`*num*`t`*num*`d`*num*`s`*num*. Except for the mount point / (root), `lucreate` supports the names of metadevices in the *device* field. Root must be mounted on a physical device.
- The keyword `merged`, indicating that the file system at the specified mount point is to be merged with its parent.

The *fs_type* field can be `ufs`, indicating a UFS file system; `vxfs`, indicating a Veritas file system; or `swap`, indicating a swap file system.

At minimum, you must specify one disk or device slice, for root. You can do this with `-m`, `-M` (described below), or in the FMLI-based interface. You must specify an `-m` argument for each file system you want to create on a new BE. For example, if you have three file systems on a source BE (say, `/`, `/usr`, and `/var`) and want these three entities as separate file systems on a new BE, you must specify three `-m` arguments. If you were to specify only one, in our example, `/`, `/usr`, and `/var` would be merged on the new BE into a single file system, under `/`.

When using the `-m` option to specify swap partition(s), you can designate only swap partitions (all or a subset) on the source BE. Any swap assignment made with `-m` replaces (that is, does not add to) existing swap assignments. See EXAMPLES, below.

`-M` *slice_list*
List of `-m` options, collected in the file *slice_list*. Specify these arguments in the format specified for `-m`. Comment lines, beginning with a hash mark (`#`), are ignored. The `-M` option is useful where you have a long list of file systems for a BE. Note that you can combine `-m` and `-M` options. For example, you can store swap partitions in *slice_list* and specify `/` and `/usr` slices with `-m`.

The `-m` and `-M` options support the listing of multiple slices for a given mount point. In processing these slices, `lucreate` skips any unavailable slices and selects the first available slice. See EXAMPLES.

`-n` *BE_name*
The name of the BE to be created. *BE_name* must be unique on a given system.

`-o` *outfile*
All command output is sent to *outfile*, in addition to where it is sent in your current environment.

`-s` (`-` | *BE_name*)
Source for the creation of the new BE. This option enables you to use a BE other than the current BE as the source for creation of a new BE. If you specify a hyphen (`-`) as an argument to `-s`, `lucreate` creates the new BE, but does not populate it. You can then use `lumake`(1M) to populate the BE or `luupgrade`(1M) to install a flash archive on the BE. This option is especially useful for installing a flash archive. See `flar`(1M).

`-X`
Enable XML output. Characteristics of XML are defined in DTD, in `/usr/share/lib/xml/dtd/lu_cli.dtd.`<*num*>, where <*num*> is the version number of the DTD file.

EXAMPLES

The `lucreate` command produces copious output. In the following examples, this output is not reproduced, except where it is needed for clarity.

EXAMPLE 1 Creating a New Boot Environment for the First Time

The following command sequence creates a new boot environment on a machine on which a BE has never been created. Note that, in the first command, the `-c` option is omitted.

```
# lucreate -m /:/dev/dsk/c0t4d0s0:ufs -m /usr:/dev/dsk/c0t4d0s1:ufs \
-n second_disk
lucreate: Please wait while your system configuration is determined.
lucreate: No name for Current BE.
lucreate: ERROR: The current BE is not named - please provide the name
to use for the current BE with the <-c> option.
```

The same command is entered, with the addition of `-c`:

```
# lucreate -c first_disk -m /:/dev/dsk/c0t4d0s0:ufs \
-m /usr:/dev/dsk/c0t4d0s1:ufs -n second_disk
many lines of output
lucreate: Creation of Boot Environment <second_disk> successful.
```

Following creation of a BE, you use `luupgrade`(1M) to upgrade the OS on the new BE and `luactivate`(1M) to make that BE the BE you will boot from upon the next reboot of your machine. Note that the swap partition and all shareable file systems for `first_disk` will be available to (shared with) `second_disk`.

```
# luupgrade -u -n second_disk \
-s /net/installmachine/export/solarisX/OS_image
many lines of output
luupgrade: Upgrade of Boot Environment <second_disk> successful.

# luactivate second_disk
```

See `luupgrade`(1M) and `luactivate`(1M) for descriptions of those commands.

EXAMPLE 2 Creating a BE using a Source Other than the Current BE

The following command uses the `-s` option to specify a source BE other than the current BE.

```
# lucreate -s third_disk -m /:/dev/dsk/c0t4d0s0:ufs \
-m /usr:/dev/dsk/c0t4d0s1:ufs -n second_disk
many lines of output
lucreate: Creation of Boot Environment <second_disk> successful.
```

EXAMPLE 3 Creating a BE from a Flash Archive

Performing this task involves use of `lucreate` with the `-s` – option and `luupgrade`.

```
# lucreate -s - -m /:/dev/dsk/c0t4d0s0:ufs -m /usr:/dev/dsk/c0t4d0s1:ufs \
-n second_disk
brief messages
lucreate: Creation of Boot Environment <second_disk> successful.
```

With the `-s` option, the `lucreate` command completes it work within seconds. At this point, you can use `luupgrade` to install the flash archive:

```
# luupgrade -f -n second_disk \
-s /net/installmachine/export/solarisX/OS_image \
-J "archive_location http://example.com/myflash.flar"
```

See `luupgrade`(1M) for a description of that command.

EXAMPLE 4 Using Swap Partitions on Multiple Disks

The command below creates a BE on a second disk and specifies the sharing of swap partitions on both the first and second disks. Note that the current boot environment must already be using `/dev/dsk/c0t0d0s1` and `/dev/dsk/c0t4d0s1` (on the second disk) as its swap partitions before entering this command.

```
# lucreate -m /:/dev/dsk/c0t4d0s0:ufs -m -:/dev/dsk/c0t4d0s1:swap \
 -m -:/dev/dsk/c0t0d0s1:swap -n second_disk
many lines of output
lucreate: Creation of Boot Environment <second_disk> successful.
```

Following completion of the preceding command, the BE `second_disk` will use both `/dev/dsk/c0t0d0s1` and `/dev/dsk/c0t4d0s1` as swap partitions. These swap assignments take effect only after booting from `second_disk`. If you have a long list of swap partitions, it is useful to use the `-M` option, as shown below.

EXAMPLE 5 Using a Combination of `-m` and `-M` Options

In this example, a list of swap partitions is collected in the file `/etc/lu/swapslices`. The location and name of this file is user-defined. The contents of `/etc/lu/swapslices`:

```
-:/dev/dsk/c0t3d0s2:swap
-:/dev/dsk/c0t3d0s2:swap
-:/dev/dsk/c0t4d0s2:swap
```

EXAMPLE 5 Using a Combination of `-m` and `-M` Options *(Continued)*

```
-:/dev/dsk/c0t5d0s2:swap
-:/dev/dsk/c1t3d0s2:swap
-:/dev/dsk/c1t4d0s2:swap
-:/dev/dsk/c1t5d0s2:swap
```

This file is specified in the following command:

```
# lucreate -m /:/dev/dsk/c02t4d0s0:ufs -m /usr:/dev/dsk/c02t4d0s1:ufs \
-M /etc/lu/swapslices -n second_disk
many lines of output
lucreate: Creation of Boot Environment <second_disk> successful.
```

The BE `second_disk` will swap onto the partitions specified in `/etc/lu/swapslices`. As with the previous example, the current BE must already be using the swap partitions specified on the command line before you enter the `lucreate` command.

EXAMPLE 6 Copying Versus Sharing

The following command copies the user file system `/home` (in addition to the non–shareable file systems `/` and `/usr`) from the current BE to the new BE:

```
# lucreate /:/dev/dsk/c0t4d0s0:ufs -m /usr:/dev/dsk/c0t4d0s1:ufs \
-m /home:/dev/dsk/c0t4d0s4:ufs -n second_disk
```

The following command differs from the preceding in that the `-m` option specifying a destination for `/home` is omitted. The result of this is that `/home` will be shared between the current BE and the BE `second_disk`.

```
# lucreate /:/dev/dsk/c0t4d0s0:ufs -m /usr:/dev/dsk/c0t4d0s1:ufs \
-n second_disk
```

EXAMPLE 7 Invoking FMLI-based Interface

The command below, by omitting `-m` or `-M` options, invokes the FMLI-based interface for Live Upgrade operations. See `lu`(1M) for a description of this interface.

```
# lucreate -n second_disk
```

The preceding command uses the current BE as the source for the target BE `second_disk`. In the FMLI interface, you can specify the target disk slices for `second_disk`. The following command is a variation on the preceding:

```
# lucreate -n second_disk -s third_disk
```

In the preceding command, a source for the target BE is specified. As before, the FMLI interface comes up, enabling you to specify target disk slices for the new BE.

EXAMPLE 8 Merging File Systems

The command below merges the `/usr/opt` file system into the `/usr` file system. First, here are the disk slices in the BE `first_disk`, expressed in the format used for arguments to the `-m` option:

EXAMPLE 8 Merging File Systems *(Continued)*

```
/:/dev/dsk/c0t4d0s0:ufs
/usr:/dev/dsk/c0t4d0s1:ufs
/usr/opt:/dev/dsk/c0t4d0s3:ufs
```

The following command creates a BE `second_disk` and performs the merge operation, merging `/usr/opt` with its parent, `/usr`.

```
# lucreate -m /:/dev/dsk/c0t4d0s0:ufs -m /usr:/dev/dsk/c0t4d0s1:ufs \
-m /usr/opt:merged:ufs -n second_disk
```

EXAMPLE 9 Splitting a File System

Assume a source BE with `/`, `/usr`, and `/var` all mounted on the same disk slice. The following command creates a BE `second_disk` that has `/`, `/usr`, and `/var` all mounted on different disk slices.

```
# lucreate -m /:/dev/dsk/c0t4d0s0:ufs -m /usr:/dev/dsk/c0t4d0s1:ufs \
/var:/dev/dsk/c0t4d0s3:ufs -n second_disk
```

This separation of a file system's (such as root's) components onto different disk slices is referred to as splitting a file system.

EXAMPLE 10 Specifying Alternative Slices

The following command uses multiple `-m` options as alternative disk slices for the new BE `second_disk`.

```
# lucreate -m /:/dev/dsk/c0t4d0s0:ufs -m /:/dev/dsk/c0t4d0s1:ufs \
-m /:/dev/dsk/c0t4d0s5:ufs -n second_disk
many lines of output
lucreate: Creation of Boot Environment <second_disk> successful.
```

The preceding command specifies three possible disk slices, `s0`, `s1`, and `s5` for the `/` file system. `lucreate` selects the first one of these slices that is not being used by another BE. Note that the `-s` option is omitted, meaning that the current BE is the source BE for the creation of the new BE.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

FILES

`/etc/lutab`
list of BEs on the system

`/usr/share/lib/xml/dtd/lu_cli.dtd.`*<num>*
Live Upgrade DTD (see `-X` option)

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWluu

SEE ALSO

`lu`(1M), `luactivate`(1M), `lucancel`(1M), `lucompare`(1M), `lucurr`(1M), `ludelete`(1M), `ludesc`(1M), `lufslist`(1M), `lumake`(1M), `lumount`(1M), `lurename`(1M), `lustatus`(1M), `luupgrade`(1M), `lutab`(4), `attributes`(5), `live_upgrade`(5)

NOTES

When splitting a directory into multiple mount points, hard links are not maintained across file systems. For example, if `/usr/test1/buglist` is hard linked to `/usr/test2/buglist`, and `/usr/test1` and `/usr/test2` are split into separate file systems, the link between the files will no longer exist. `lucreate` issues a warning message to that effect and a symbolic link is created to replace the lost hard link.

`lucreate` cannot prevent you from making invalid configurations with respect to non-shareable file systems. For example, you could enter an `lucreate` command that would create separate file systems for `/` and `/kernel`—an invalid division of `/`. The resulting BE would be unbootable. When creating file systems for a boot environment, the rules are identical to the rules for creating file systems for the Solaris operating environment.

Mindful of the principle described in the preceding paragraph, consider the following:

- In a source BE, you must have valid `vfstab` entries for every file system you want to copy to or share with a new BE.
- You cannot create a new BE on a disk with overlapping partitions. The `lucreate` command that specifies such a disk might complete, but the resulting BE would be unbootable.

NAME

lucurr – display the name of the active boot environment

SYNOPSIS

/usr/sbin/lucurr [-l *error_log*] [-m *mount_point*] [-o *outfile*] [-X]

DESCRIPTION

The `lucurr` command is part of a suite of commands that make up the Live Upgrade feature of the Solaris operating environment. See `live_upgrade`(5) for a description of the Live Upgrade feature.

The `lucurr` command displays the name of the currently running boot environment (BE). If no BEs are configured on the system, `lucurr` displays the message "No Boot Environments are defined". Note that `lucurr` reports only the name of the current BE, not the BE that will be active upon the next reboot. Use `lustatus`(1M) or `luactivate`(1M) for this information.

The `lucurr` command requires root privileges.

OPTIONS

The `lucurr` command has the following options:

`-l` *error_log* — Error and status messages are sent to *error_log*, in addition to where they are sent in your current environment.

`-m` *mount_point* — Returns the name of the BE that owns *mount_point*, where *mount_point* is the mount point of a BE's root file system. This can be a mount point of the current BE or the mount point of a BE other than the current BE. If the latter, the file system of the BE must have been mounted with `lumount`(1M) or `mount`(1M) before entering this option.

`-o` *outfile* — All command output is sent to *outfile*, in addition to where it is sent in your current environment.

`-X` — Enable XML output. Characteristics of XML are defined in DTD, in `/usr/share/lib/xml/dtd/lu_cli.dtd.`<*num*>, where <*num*> is the version number of the DTD file.

EXIT STATUS

The following exit values are returned:

`0` — Successful completion.

`>0` — An error occurred.

FILES

`/etc/lutab`
list of BEs on the system

`/usr/share/lib/xml/dtd/lu_cli.dtd.`<*num*>
Live Upgrade DTD (see `-X` option)

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWluu

SEE ALSO | `lu`(1M), `luactivate`(1M), `lucancel`(1M), `lucompare`(1M), `lucreate`(1M), `ludelete`(1M), `ludesc`(1M), `lufslist`(1M), `lumake`(1M), `lumount`(1M), `lurename`(1M), `lustatus`(1M), `luupgrade`(1M), `lutab`(4), `attributes`(5), `live_upgrade`(5)

NAME

ludelete – delete a boot environment

SYNOPSIS

/usr/sbin/ludelete [-l *error_log*] [-o *outfile*] *BE_name* [-X]

DESCRIPTION

The `ludelete` command is part of a suite of commands that make up the Live Upgrade feature of the Solaris operating environment. See `live_upgrade`(5) for a description of the Live Upgrade feature.

The `ludelete` command deletes all records associated with a boot environment (BE) on all defined complete BEs. A complete BE is one that is not participating in an `lucreate`(1M), `luupgrade`(1M), or `lucompare`(1M) operation. Use `lustatus`(1M) to determine a BE's status. You can delete neither the current BE, nor the BE that will become current upon the next reboot. Also, you cannot delete a BE that has file systems mounted with `lumount`(1M) or `mount`(1M).

`ludelete` does not alter any files on the BE being deleted.

The `ludelete` command requires root privileges.

OPTIONS

The `ludelete` command has the following options:

`-l` *error_log* Error and status messages are sent to *error_log*, in addition to where they are sent in your current environment.

`-o` *outfile* All command output is sent to *outfile*, in addition to where it is sent in your current environment.

`-X` Enable XML output. Characteristics of XML are defined in DTD, in `/usr/share/lib/xml/dtd/lu_cli.dtd.`<*num*>, where <*num*> is the version number of the DTD file.

OPERANDS

BE_name Name of the BE to be deleted.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

FILES

`/etc/lutab`
list of BEs on the system

`/usr/share/lib/xml/dtd/lu_cli.dtd.`<*num*>
Live Upgrade DTD (see `-X` option)

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWluu

SEE ALSO | `lu`(1M), `luactivate`(1M), `lucancel`(1M), `lucompare`(1M), `lucreate`(1M), `lucurr`(1M), `ludesc`(1M), `lufslist`(1M), `lumake`(1M), `lumount`(1M), `lurename`(1M), `lustatus`(1M), `luupgrade`(1M), `lutab`(4), `attributes`(5), `live_upgrade`(5)

NAME

ludesc – display or set boot environment description

SYNOPSIS

/usr/sbin/ludesc {`-A` *BE_description*} | {`-f` {*filename* | `-`}} [`-l` *error_log*] [`-o` *outfile*] [`-X`]

/usr/sbin/ludesc `-n` *BE_name* [`-f` *filename* | `-`] [`-l` *error_log*] [`-o` *outfile*] [`-X`]

/usr/sbin/ludesc `-n` *BE_name* [`-l` *error_log*] [`-o` *outfile*] [`-X`] *BE_description*

DESCRIPTION

The `ludesc` command is part of a suite of commands that make up the Live Upgrade feature of the Solaris operating environment. See `live_upgrade`(5) for a description of the Live Upgrade feature.

The `ludesc` command allows you to manipulate boot environment (BE) descriptions. A BE description is an optional attribute of a BE. It can be text or binary data. For example, it might be a string such as "S9 beta test BE" or it be a file that contains 8–bit multi-byte characters. The `ludesc` command in general and the options to manipulate binary-format descriptions in particular are suitable for use in programs.

You create a BE description using `ludesc` or `lucreate`(1M). Only `ludesc` allows you to change a BE description or add a description following BE creation.

While a BE description is associated with a BE name, it is not interchangeable with that name. No Live Upgrade command allows you to specify a BE description instead of a BE name when performing an operation on a BE.

A shell might restrict what you enter for a BE description (in both `ludesc` and `lucreate`(1M)). In entering a description, use the following guidelines:

- Always enclose a description in single quotes ('), unless the description includes a single quote.
- If your description includes a single quote, enclose the description in double quotes ("). You then must use an escape sequence (usually a backslash [`\`]) to enter a character that is special to the shell. See `sh`(1) for a list of special characters and a description of the escape sequence mechanism.

Descriptions that include many special characters might be more conveniently inserted in a file (`-f` option) than entered on a command line (`-A` option).

When `ludesc` outputs a BE description, it does so exactly as the description was entered. Because of this feature, a description that is a text string does not have a concluding newline, which means the system prompt immediately follows the last character of the description.

The `ludesc` command requires root privileges.

OPTIONS

The `ludesc` command has the following options:

`-A` *BE_description* Displays the BE name associated with *BE_description*.

`-f` {*filename* | –}	Specify the BE description contained in *filename* or read from `stdin`. When used without `-n`, displays the BE name associated with the specified BE description. Used with `-n`, changes the description for the specified BE to the description specified with `-f`.
`-l` *error_log*	Error and status messages are sent to *error_log*, in addition to where they are sent in your current environment.
`-n` *BE_name*	With no other arguments, displays the BE description for the specified BE. With the `-f` option or the *BE_description* operand, changes the description for the specified BE to that specified with `-f` or *BE_description*.
`-o` *outfile*	All command output is sent to *outfile*, in addition to where it is sent in your current environment.
`-X`	Enable XML output. Characteristics of XML are defined in DTD, in `/usr/share/lib/xml/dtd/lu_cli.dtd.`<*num*>, where <*num*> is the version number of the DTD file.

OPERANDS

BE_description	Used only with the `-n` option. *BE_description* replaces the current BE description for the specified BE.

EXAMPLES

The following are examples of the use of `ludesc`.

EXAMPLE 1 Basic Use

The first command, below, assigns a description to a BE. The second command returns the name of the BE associated with the specified description. The last command returns the description associated with a specified BE.

```
# ludesc -n first_disk 'Test disk'
Setting description for boot environment <first_disk>.
Propagating the change of BE description to all BEs.

# ludesc -A 'Test disk'
first_disk
#

# ludesc -n first_disk
Test disk#
```

As seen above and noted in the DESCRIPTION, `ludesc` does not append a newline to the display of BE description that is a text string.

EXAMPLE 2 Using Binary Files

The following commands are analogs of the preceding examples, substituting a binary file–here, a file containing a description in Russian, using the Cyrillic alphabet—for a text string. In the third command, note the use of a file to capture output. Sending output of a binary file to the console can produce erratic results.

EXAMPLE 2 Using Binary Files *(Continued)*

```
# ludesc -n first_disk -f arrayBE.ru
Setting description for boot environment <first_disk>.
Propagating the change of BE description to all BEs.

# ludesc -f arrayBE.ru
first_disk

# ludesc -n first_disk > /tmp/arrayBE.out
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

FILES

`/etc/lutab`
list of BEs on the system

`/usr/share/lib/xml/dtd/lu_cli.dtd.`*<num>*
Live Upgrade DTD (see `-X` option)

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWluu

SEE ALSO

`lu`(1M), `luactivate`(1M), `lucancel`(1M), `lucompare`(1M), `lucreate`(1M), `ludelete`(1M), `lufslist`(1M), `lumake`(1M), `lumount`(1M), `lurename`(1M), `lustatus`(1M), `luupgrade`(1M), `lutab`(4), `attributes`(5), `live_upgrade`(5)

NAME

lufslist – list configuration of a boot environment

SYNOPSIS

/usr/sbin/lufslist [-l *error_log*] [-o *outfile*] *BE_name* [-X]

DESCRIPTION

The `lufslist` command is part of a suite of commands that make up the Live Upgrade feature of the Solaris operating environment. See `live_upgrade`(5) for a description of the Live Upgrade feature.

The `lufslist` command lists the configuration of a boot environment (BE). The output contains the disk slice (file system), file system type, and file system size for each BE mount point.

The following is an example of `lufslist` output.

```
# lufslist BE_name
Filesystem              fstype       size(Mb) Mounted on
-------------------------------------------------------------------
/dev/dsk/c0t0d0s1       swap           512.11 -
/dev/dsk/c0t4d0s3       ufs           3738.29 /
/dev/dsk/c0t4d0s4       ufs            510.24 /opt
```

File system type can be `ufs`, `swap`, or `vxfs`, for a Veritas file system. Under the `Filesystem` heading can be a disk slice or a logical device, such as a disk metadevice used by volume management software.

The `lufslist` command requires root privileges.

OPTIONS

The `lufslist` command has the following options:

`-l` *error_log* — Error and status messages are sent to *error_log*, in addition to where they are sent in your current environment.

`-o` *outfile* — All command output is sent to *outfile*, in addition to where it is sent in your current environment.

`-X` — Enable XML output. Characteristics of XML are defined in DTD, in `/usr/share/lib/xml/dtd/lu_cli.dtd.`<*num*>, where <*num*> is the version number of the DTD file.

OPERANDS

BE_name — Name of the BE for which file systems are to be reported. You cannot specify a BE that is involved in another Live Upgrade operation.

EXIT STATUS

The following exit values are returned:

`0` — Successful completion.

`>0` — An error occurred.

FILES

`/etc/lutab`
list of BEs on the system

`/usr/share/lib/xml/dtd/lu_cli.dtd.`<*num*>
Live Upgrade DTD (see `-X` option)

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWluu

SEE ALSO lu(1M), luactivate(1M), lucancel(1M), lucompare(1M), lucreate(1M), lucurr(1M), ludelete(1M), ludesc(1M), lumake(1M), lumount(1M), lurename(1M), lustatus(1M), luupgrade(1M), lutab(4), attributes(5), live_upgrade(5)

NAME

lumake – populate a boot environment

SYNOPSIS

```
/usr/sbin/lumake [-l error_log] [-o outfile] [-s source_BE] -n BE_name
     [-X]

/usr/sbin/lumake [-l error_log] -t time [-o outfile] [-s source_BE]
     -n BE_name [-m email_address] [-X]
```

DESCRIPTION

The `lumake` command is part of a suite of commands that make up the Live Upgrade feature of the Solaris operating environment. See `live_upgrade`(5) for a description of the Live Upgrade feature.

The `lumake` command populates (that is, copies files to) the file systems of a specified boot environment (BE) by copying files from the corresponding file systems of the active or a source (`-s`) BE. Any existing data on the target BE are destroyed. All file systems on the target BE are re-created.

The target BE must already exist. Use `lucreate`(1M) to create a new BE.

The `lumake` command requires root privileges.

OPTIONS

The `lumake` command has the following options:

`-n` *BE_name*	Name of the BE to be populated.
`-s` *source_BE*	The optional name of a source BE. If you omit this option, `lumake` uses the current BE as the source. A BE must have the status "complete" before you can copy from it. Use `lustatus`(1M) to determine a BE's status.
`-l` *error_log*	Error and status messages are sent to *error_log*, in addition to where they are sent in your current environment.
`-o` *outfile*	All command output is sent to *outfile*, in addition to where it is sent in your current environment.
`-t` *time*	Setup a batch job to populate the specified BE at a specified time. The time is given in the format specified by the `at`(1) man page. At any time, you can have only one Live Upgrade operation scheduled. You can use `lucancel`(1M) to cancel a scheduled `lumake` operation.
`-m` *email_address*	Allows you to email `lumake` output to a specified address upon command completion. There is no checking of *email_address*. You can use this option only in conjunction with `-t`.
`-X`	Enable XML output. Characteristics of XML are defined in DTD, in `/usr/share/lib/xml/dtd/lu_cli.dtd.`<*num*>, where <*num*> is the version number of the DTD file.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

FILES

`/etc/lutab`
list of BEs on the system

`/usr/share/lib/xml/dtd/lu_cli.dtd.<num>`
Live Upgrade DTD (see `-X` option)

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWluu

SEE ALSO

`lu`(1M), `luactivate`(1M), `lucancel`(1M), `lucompare`(1M), `lucreate`(1M), `lucurr`(1M), `ludelete`(1M), `ludesc`(1M), `lufslist`(1M), `lumount`(1M), `lurename`(1M), `lustatus`(1M), `luupgrade`(1M), `lutab`(4), `attributes`(5), `live_upgrade`(5)

NAME

lumount, luumount – mount or unmount all file systems in a boot environment

SYNOPSIS

/usr/sbin/lumount [-l *error_log*] [-o *outfile*] *BE_name* [*mount_point*] [-X]

/usr/sbin/luumount [-f] { [-n] *BE_name* | [-m] *mount_point* | *block_device*} [-l *error_log*] [-o *outfile*] [-X]

DESCRIPTION

The `lumount` and `luumount` commands are part of a suite of commands that make up the Live Upgrade feature of the Solaris operating environment. See `live_upgrade`(5) for a description of the Live Upgrade feature.

The `lumount` and `luumount` commands enable you to mount or unmount all of the filesystems in a boot environment (BE). This allows you to inspect or modify the files in a BE while that BE is not active. By default, `lumount` mounts the file systems on a mount point of the form `/.alt.`*BE_name*, where *BE_name* is the name of the BE whose file systems are being mounted. See NOTES.

The `lumount` and `luumount` commands require root privileges.

OPTIONS

The `lumount` and `luumount` commands have the following options:

`-f`
For `luumount` only, forcibly unmount a BE's file systems after attempting (and failing) an unforced unmount. This option is analogous to the `umount`(1M) `-f` option.

`-l` *error_log*
Error and status messages are sent to *error_log*, in addition to where they are sent in your current environment.

`-m` *mount_point*
`luumount` unmounts the file systems of the BE that owns *mount_point*. See description of *mount_point* under OPERANDS, below. The use of `-m` is optional when specifying a mount point for `luumount`.

`-n` *BE_name*
Name of the BE whose file systems will be unmounted. See description of *BE_name* under OPERANDS, below. The use of `-n` is optional when specifying a BE name for `luumount`.

`-o` *outfile*
All command output is sent to *outfile*, in addition to where it is sent in your current environment.

`-X`
Enable XML output. Characteristics of XML are defined in DTD, in `/usr/share/lib/xml/dtd/lu_cli.dtd.`<*num*>, where <*num*> is the version number of the DTD file.

For `luumount`, if you supply an argument and specify neither `-m` nor `-n`, the command determines whether your argument is a BE name, a mount point, or a block device. If it is one of these three and the argument is associated with a BE that has mounted file systems, `luumount` unmounts the file systems of that BE. Otherwise, `luumount` returns an error.

OPERANDS

BE_name — Name of the BE whose file systems will be mounted or unmounted. This is a BE on the current system other than the active BE. Note that, for successful completion of an `lumount` or `luumount` command, the status of a BE must be complete, as reported by `lustatus`(1M). Also, none of the BE's disk slices can be mounted (through use of `mount`(1M)).

mount_point — For `lumount`, a mount point to use instead of the default `/.alt.`*BE_name*. If *mount_point* does not exist, `lumount` creates it. For `luumount`, the BE associated with *mount_point* will have its file systems unmounted. Note that default mount points are automatically deleted upon unmounting with `luumount`. Mount points that you specify are not deleted.

block_device — For `luumount` only, *block_device* is the root slice of a BE, such as `/dev/dsk/c0t4d0s0`. `luumount` unmounts the file systems of the BE associated with *block_device*.

EXAMPLES

EXAMPLE 1 Specifying a Mount Point

The following command creates the mount point `/test` and mounts the file systems of the BE `second_disk` on `/test`.

```
# lumount second_disk /test
/test
```

You can then `cd` to `/test` to view the file systems of `second_disk`. If you did not specify `/test` as a mount point, `lumount` would create a default mount point named `/alt.second_disk`.

EXAMPLE 2 Unmounting File Systems

The following command unmounts the file systems of the BE `second_disk`. In this example, we `cd` to `/` to ensure we are not in any of the file systems in `second_disk`.

```
# cd /
# luumount second_disk
#
```

If `/dev/dsk/c0t4d0s0` were the root slice for `second_disk`, you could enter the following command to match the effect of the preceding command.

EXAMPLE 2 Unmounting File Systems *(Continued)*

```
# cd /
# luumount /dev/dsk/c0t4d0s0
#
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

FILES

`/etc/lutab` list of BEs on the system

`/usr/share/lib/xml/dtd/live_upgrade.dtd.1` Live Upgrade DTD (see `-X` option)

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWluu

SEE ALSO

`lu`(1M), `luactivate`(1M), `lucancel`(1M), `lucompare`(1M), `lucreate`(1M), `lucurr`(1M), `ludelete`(1M), `ludesc`(1M), `lufslist`(1M), `lumake`(1M), `lurename`(1M), `lustatus`(1M), `luupgrade`(1M), `lutab`(4), `attributes`(5), `live_upgrade`(5)

NOTES

If a BE name contains slashes (`/`), `lumount` replaces those slashes with colons in a default mount point name. For example:

```
# lumount 'first/disk'
/.alt.first:disk
```

NAME

lurename – change the name of a boot environment

SYNOPSIS

```
/usr/sbin/lurename -e BE_name -n new_name [-l error_log] [-o outfile]
     [-X]
```

DESCRIPTION

The `lurename` command is part of a suite of commands that make up the Live Upgrade feature of the Solaris operating environment. See `live_upgrade`(5) for a description of the Live Upgrade feature.

The `lurename` command renames the boot environment (BE) *BE_name* to *new_name*.

The string *new_name* must not exceed 30 characters in length and must consist only of alphanumeric characters and other ASCII characters that are not special to the Unix shell. See the "Quoting" section of `sh`(1). The BE name can contain only single-byte, 8–bit characters. It cannot contain whitespace characters. Also, *new_name* must be unique on the system.

A BE must have the status "complete" before you rename it. Use `lustatus`(1M) to determine a BE's status. Also, you cannot rename a BE that has file systems mounted with `lumount`(1M) or `mount`(1M).

Renaming a BE is often useful when you upgrade the BE from one Solaris release to another. For example, following an operating system upgrade, you might rename the BE `solaris7` to `solaris8`.

The `lurename` command requires root privileges.

OPTIONS

The `lurename` command has the options listed below.

`-e` *BE_name*	Name of the BE whose name you want to change.
`-l` *error_log*	Error and status messages are sent to *error_log*, in addition to where they are sent in your current environment.
`-n` *new_name*	New name of the BE. *new_name* must be unique on a given system.
`-o` *outfile*	All command output is sent to *outfile*, in addition to where it is sent in your current environment.
`-X`	Enable XML output. Characteristics of XML are defined in DTD, in `/usr/share/lib/xml/dtd/lu_cli.dtd.`<*num*>, where <*num*> is the version number of the DTD file.

EXIT STATUS

The following exit values are returned:

`0`	Successful completion.
`>0`	An error occurred.

FILES

`/etc/lutab`
list of BEs on the system

`/usr/share/lib/xml/dtd/lu_cli.dtd.`*<num>*
Live Upgrade DTD (see `-X` option)

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWluu

SEE ALSO

`lu`(1M), `luactivate`(1M), `lucancel`(1M), `lucompare`(1M), `lucreate`(1M), `lucurr`(1M), `ludelete`(1M), `ludesc`(1M), `lufslist`(1M), `lumake`(1M), `lumount`(1M), `lustatus`(1M), `luupgrade`(1M), `lutab`(4), `attributes`(5), `live_upgrade`(5)

NAME

lustatus – display status of boot environments

SYNOPSIS

/usr/sbin/lustatus [-l *error_log*] [-o *outfile*] [*BE_name*] [-X]

DESCRIPTION

The `lustatus` command is part of a suite of commands that make up the Live Upgrade feature of the Solaris operating environment. See `live_upgrade`(5) for a description of the Live Upgrade feature.

The `lustatus` command displays the status information of the boot environment (BE) *BE_name*. If no BE is specified, the status information for all BEs on the system is displayed.

The headings in the `lustatus` information display are described as follows:

`BE_name` Name of the BE.

`Complete` Indicates whether a BE is able to be booted. Any current activity or failure in an `lucreate`() or `luupgrade`(1M) operation causes a BE to be incomplete. For example, if there is a copy operation proceeding on or scheduled for a BE, that BE is considered incomplete.

`Active` Indicates whether the BE is currently active. The "active" BE is the one currently booted.

`ActiveOnReboot` Indicates whether the BE becomes active upon next reboot of the system.

`CopyStatus` Indicates whether the creation or repopulation of a BE is scheduled or active (that is, in progress). A status of `ACTIVE`, `COMPARING` (from `lucompare`(1M)), `UPGRADING`, or `SCHEDULED` prevents you performing Live Upgrade copy, rename, or upgrade operations.

The following is an example `lustatus` display:

```
BE_name             Complete  Active  ActiveOnReboot  CopyStatus
------------------------------------------------------------------------
disk_a_S7            yes       yes     yes              -
disk_b_S7database    yes       no      no              UPGRADING
disk_b_S8            no        no      no               -
```

Note that you could not perform copy, rename, or upgrade operations on `disk_b_S8`, because it is not complete, nor on `disk_b_S7database`, because a Live Upgrade operation is pending.

The `lustatus` command requires root privileges.

OPTIONS

The `lustatus` command has the following options:

`-l` *error_log* Error and status messages are sent to *error_log*, in addition to where they are sent in your current environment.

`-o` *outfile* All command output is sent to *outfile*, in addition to where it is sent in your current environment.

`-X` Enable XML output. Characteristics of XML are defined in DTD, in `/usr/share/lib/xml/dtd/lu_cli.dtd.`<*num*>, where <*num*> is the version number of the DTD file.

OPERANDS

BE_name Name of the BE for which to obtain status. If *BE_name* is omitted, `lustatus` displays status for all BEs in the system.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

FILES

`/etc/lutab`
list of BEs on the system

`/usr/share/lib/xml/dtd/lu_cli.dtd.`<*num*>
Live Upgrade DTD (see `-X` option)

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWluu

SEE ALSO

`lu`(1M), `luactivate`(1M), `lucancel`(1M), `lucompare`(1M), `lucreate`(1M), `lucurr`(1M), `ludesc`(1M), `ludelete`(1M), `lufslist`(1M), `lumake`(1M), `lumount`(1M), `lurename`(1M), `luupgrade`(1M), `lutab`(4), `attributes`(5), `live_upgrade`(5)

NAME

luupgrade – installs, upgrades, and performs other functions on software on a boot environment

SYNOPSIS

/usr/sbin/luupgrade [-iIufpPtTcC] [*options*]

DESCRIPTION

The `luupgrade` command is part of a suite of commands that make up the Live Upgrade feature of the Solaris operating environment. See `live_upgrade`(5) for a description of the Live Upgrade feature.

The `luupgrade` command enables you to install software on a specified boot environment (BE). Specifically, `luupgrade` performs the following functions:

- Upgrades an operating system image on a BE (`-u` option). The source for the image can be any valid Solaris installation medium.
- Run an installer program to install software from an installation medium (`-i` option).
- Extract a flash archive onto a BE (`-f` option). (See `flar`(1M).)
- Add a package to (`-p`) or remove a package from (`-P`) a BE.
- Add a patch to (`-t`) or remove a patch from (`-T`) a BE.
- Check (`-C`) or obtain information about (`-I`) packages.
- Check an operating system installation medium (`-c`).

Before using `luupgrade`, you must have created a BE, using either the `lucreate`(1M) command or `lu`(1M), the FMLI-based user interface. You can upgrade only BEs other than the current BE.

The functions described in the preceding list each has its own set of options, which are described separately for each function.

Note that, for successful completion of an `luupgrade` operation, the status of a BE must be complete, as reported by `lustatus`(1M). Also, the BE must not have any mounted disk slices, mounted either with `lumount`(1M) or `mount`(1M).

The `luupgrade` command requires root privileges.

Options that Apply to All Uses

The following options are available for all uses of `luupgrade`:

`-l` *error_log* — Error and status messages are sent to *error_log*, in addition to where they are sent in your current environment.

`-o` *outfile* — All command output is sent to *outfile*, in addition to where it is sent in your current environment.

`-N` — Dry-run mode. Enables you to determine whether your command arguments are correctly formed. Does not apply to the `-c` (check medium) function.

`-X` — Enable XML output. Characteristics of XML are defined in DTD, in `/usr/share/lib/xml/dtd/lu_cli.dtd.`<*num*>, where <*num*> is the version number of the DTD file.

Upgrading an Operating System Image

The `luupgrade` command uses `-u` to upgrade an operating system image. The syntax is as follows:

```
luupgrade -u -n BE_name -s os_image_path [ -j profile_path ]
[ -l error_log ] [ -o outfile ] [-N]
```

The first option, `-u`, indicates the function to perform—to install an OS image. The remaining options, shown above, are described as follows:

`-n` *BE_name*	Name of the BE to receive an OS upgrade.
`-s` *os_image_path*	Path name of a directory containing an OS image. This can be a directory on an installation medium such as a CD-ROM or can be an NFS or UFS directory.
`-j` *profile_path*	Path to a JumpStart profile. See `pfinstall`(1M) and the Solaris installation documentation for information on the JumpStart software.

Note that if you are upgrading from a medium with multiple components, such as from multiple CDs, use `luupgrade` with the `-i` option, as described in the section below, to install software from the second and any following media.

Running an Installer Program

The `luupgrade` command uses `-i` to run an installer program. The syntax is as follows:

```
luupgrade -i -n BE_name -s installation_medium [ -N ]
 [ -O "installer_options" ] [ -l error_log ] [ -o outfile ]
```

The first option, `-i`, indicates the function to perform—to run an installer program on the installation specified with `-s`. The remaining options, shown above, are described as follows:

`-n` *BE_name*	Name of the BE on which software is to be installed.
`-O` "*installer_options*"	Options passed directly to the Solaris installer program. See `installer`(1M) for descriptions of the installer options.
`-s` *installation_medium*	Path name of an installation medium. This can be a CD, or an NFS or UFS directory.

With the `-i` option, `luupgrade` looks for an installation program on the specified medium and runs that program.

The `-i` option has a special use when you use the `-u` option, described above, to install software from a multiple-component medium, such as multiple CDs. See EXAMPLES.

Upgrading from a Flash Archive

The `luupgrade` command uses `-f` to upgrade an operating system from a flash archive. The syntax is as follows:

```
luupgrade -f -n BE_name -s os_image_path ( -a archive | -j profile_path
 | -J "profile" ) [ -l error_log ] [ -o outfile ] [ -N ]
```

The first option, `-f`, indicates the function to perform—to upgrade an OS from a flash archive. The remaining options, shown above, are described as follows:

`-n` *BE_name*
Name of the BE to receive an OS installation.

`-s` *os_image_path*
Path name of a directory containing an OS image. This can be a directory on an installation medium, such as a CD-ROM, or can be an NFS or UFS directory.

`-a` *archive*
Path to the flash archive when the archive is available on the local file system. You must specify one of `-a`, `-j`, or `-J`.

`-j` *profile_path*
Path to a JumpStart profile that is configured for a flash installation. See `pfinstall`(1M) and the Solaris installation documentation for information on the JumpStart software. You must specify one of `-a`, `-j`, or `-J`.

`-J` *"profile"*
Entry from a JumpStart profile that is configured for a flash installation. See `pfinstall`(1M) and the Solaris installation documentation for information on the JumpStart software. You must specify one of `-a`, `-j`, or `-J`.

Note that the version of the OS image specified with `-a` must be identical to the version of the OS contained in the flash archive specified with the `-a`, `-j`, or `-J` options.

Add or Remove Packages

The `luupgrade` command uses `-p` to add a package and `-P` to remove a package. The syntax is as follows:

For adding packages:

```
luupgrade -p -n BE_name -s packages_path [ -l error_log ][ -o outfile ]
[ -O "pkgadd_options" ] [ -a admin ] [ pkginst [ pkginst...]] [ -N ]
```

For removing packages:

```
luupgrade -P -n BE_name [ -l error_log ][ -o outfile ]
[ -O "pkgrm_options" ] [ pkginst [ pkginst...]] [ -N ]
```

The first option, `-p`, to add packages, or `-P` to remove packages, indicates the function to perform. The remaining options, shown above, are described as follows:

`-n` *BE_name*
Name of the BE to which packages will be added or from which packages will be removed.

`-s` *packages_path*
(For adding packages only.) Path name of a directory containing packages to add. You can substitute `-d` for `-s`. The `-d` support is for `pkgadd`(1M) compatibility.

`-d` *packages_path*
Identical to `-s`. Use of `-s` is recommended.

`-O "`*pkgadd_options*`" or "`*pkgrm_options*`"`
Options passed directly to `pkgadd` (for `-p`) or `pkgrm` (for `-P`). See `pkgadd`(1M) and `pkgrm`(1M) for descriptions of the options for those commands.

`-a` *admin*
(For adding packages only.) Path to an admin file. Identical to the `pkgadd -a` option. Use of the `-a` option here is identical to `-O` "`-a` *admin*"

pkginst [*pkginst*...]
Zero or more packages to add or remove. For adding packages, the default is to add all of the packages specified with the `-s` option, above. Separate multiple package names with spaces.

It is critically important that any packages you add be compliant with the SVR4 Advanced Packaging Guidelines. See WARNINGS, below.

Add or Remove Patches

The `luupgrade` command uses `-t` to add a patch and `-T` to remove a patch. The syntax is as follows:

For adding patches:

```
luupgrade -t -n BE_name -s patch_path [ -l error_log ][ -o outfile ]
[ -O "patchadd_options" ] [ patch_name [ patch_name...]] [ -N ]
```

For removing patches:

```
luupgrade -T -n BE_name [ -l error_log ][ -o outfile ]
[ -O "patchrm_options" ] [ patch_name [ patch_name...]] [ -N ]
```

The first option, `-t`, to add patches, or `-T` to remove patches, indicates the function to perform. The remaining options, shown above, are described as follows:

`-n` *BE_name*
Name of the BE to which patches will be added or from which patches will be removed.

`-s` *patch_path*
(For adding patches only.) Path name of a directory containing patches to add.

`-O "`*patchadd_options*`" or "`*patchrm_options*`"`
Options passed directly to `patchadd` (for `-p`) or `patchrm` (for `-P`). See `patchadd`(1M) or `patchrm`(1M) for a description of these options.

patch_name [*patch_name*...]
Zero or more patches to add or remove. For adding patches, the default is to add all of the patches specified with the `-s` option, above. Separate multiple patch names with spaces.

It is critically important that any patches you add be compliant with the SVR4 Advanced Packaging Guidelines. See WARNINGS, below.

Check or Return Information on Packages

Use the `-C` to perform a `pkgchk`(1M) on all or the specified packages on a BE. Use the `-I` option to perform a `pkginfo`(1).

For performing a pkgchk:

```
luupgrade -C -n BE_name  [ -l error_log ][ -o outfile ]
[ -O "pkgchk_options" ][ pkginst [ pkginst...]] [ -N ]
```

For performing a pkginfo:

```
luupgrade -I -n BE_name [ -l error_log ][ -o outfile ]
[ -O "pkginfo_options" ][ pkginst [ pkginst...]] [ -N ]
```

The first option, -C, for pkgchk, or -I, for pkginfo, indicates the function to perform. The remaining options, shown above, are described as follows:

-n *BE_name*
: Name of the BE on which packages will be checked or on whose packages information will be returned.

-O "*pkgchk_options*" or "*pkginfo_options*"
: Options passed directly to pkgchk (for -C) or pkginfo (for -I). See pkgchk(1M) or pkginfo(1) for a description of these options.

pkginst [*pkginst*...]
: Zero or more packages to check or for which to have information returned. If you omit package names, luupgrade returns information on all of the packages on the BE. Separate multiple package names with spaces.

Check an OS Installation Medium

With the -c option, luupgrade allows you to check that a local or remote medium, such as a CD, is a valid installation medium. The -c option returns useful information about the specified medium. The syntax for this use of luupgrade is as follows:

```
luupgrade -c -s path_to_medium [ -l error_log ] [ -o outfile ]
```

The first option, -c, indicates the function to perform—to check on an installation medium. The -s option, shown above, is described as follows:

-s *path_to_medium* Path name to an installation medium such as a CD-ROM.

EXAMPLES

EXAMPLE 1 Removing, then Adding Packages

The following example removes from then adds a set of packages to a boot environment.

```
# luupgrade -P -n second_disk SUNWabc SUNWdef SUNWghi
```

Now, to add the same packages:

```
# luupgrade -p -n second_disk -s /net/installmachine/export/packages \
SUNWabc SUNWdef SUNWghi
```

The following command adds the -O option to the preceding command. This option passes arguments directly to pkgadd.

```
# luupgrade -p -n second_disk -s /net/installmachine/export/packages \
-O "-r /net/testmachine/export/responses" SUNWabc SUNWdef SUNWghi
```

See pkgadd(1M) for a description of the options for that command.

EXAMPLE 2 Upgrading to a New OS from a Combined Image

The following example upgrades the operating environment on a boot environment. The source image is stored as a combined image on a remote disk or on a DVD.

```
# luupgrade -u -n second_disk \
-s /net/installmachine/export/solarisX/OS_image
```

Following the command above you could enter the command below to activate the upgraded BE.

```
# luactivate second_disk
```

Then, upon the next reboot, `second_disk` would become the current boot environment. See `luactivate`(1M).

EXAMPLE 3 Upgrading to a New OS from Multiple CDs

The following example is a variation on the preceding. The OS upgrade resides on two CDs. To begin the upgrade on a SPARC machine, you enter:

```
# luupgrade -u -n second_disk -s /dev/cdrom/cdrom0/s0
```

On Intel Architecture (IA) machines, replace the `s0` in the argument to `-s` with `s2`.

When the installer is finished with the contents of the first CD, insert the next CD in the drive and enter the following:

```
# luupgrade -i -n second_disk -s /dev/cdrom/cdrom0 \
-O "-nodisplay -noconsole"
```

Note the use of `-i` rather than `-u` in the preceding. Were there additional CDs, you would enter the same command as the one immediately above. The `-O` options, above, are passed to `installer`(1M). If you omit these options, a graphical interface is invoked following the insertion and reading of the second CD. See `installer`(1M) for a description of the `-O` options.

Upon invocation of the first command (with `-u`) above, the `installer` command determines whether you have sufficient space on your target medium to hold the packages on all of the CDs in the set. This check takes into account packages that are being replaced by packages of the same name.

Note that a multiple-CD upgrade is not complete until you have entered and completed `luupgrade` commands for all of the CDs in a set. Following installation of packages from a CD, you might receive a message such as:

```
WARNING: <num> packages must be installed on boot environment <disk_device>.
```

Such a message indicates the requirement that you install packages from one or more additional CDs, as in the example above. If you do not complete package installation, you will not be able to use `luactivate` to activate (designate for booting) the upgraded BE.

EXAMPLE 4 Installing a New OS from a Flash Archive

The following example upgrades the operating environment on a boot environment, using a flash archive.

```
# luupgrade -f -n second_disk \
-s /net/installmachine/export/solarisX/OS_image \
-J "archive_location http://example.com/myflash.flar"
```

EXAMPLE 5 Obtaining Information on Packages

The following example runs a `pkgchk` on the packages `SUNWluu` and `SUNWlur`, passing to `pkgchk` the `-v` option.

```
# luupgrade -C -n second_disk -O "-v" SUNWluu SUNWlur
```

The following command runs `pkginfo` on the same set of packages:

```
# luupgrade -I -n second_disk -O "-v" SUNWluu SUNWlur
```

For both commands, if the package names were omitted, `luupgrade` returns package information on all of the packages in the specified BE. See `pkgchk`(1M) and `pkginfo`(1) for a description of the options for those commands.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

FILES

`/etc/lutab`
list of BEs on the system

`/usr/share/lib/xml/dtd/lu_cli.dtd.`*<num>*
Live Upgrade DTD (see `-X` option in "Options that Apply to All Uses," above)

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWluu

SEE ALSO

`installer`(1M), `lu`(1M), `luactivate`(1M), `lucancel`(1M), `lucompare`(1M), `lucreate`(1M), `lucurr`(1M), `ludelete`(1M), `ludesc`(1M), `lufslist`(1M), `lumake`(1M), `lumount`(1M), `lurename`(1M), `lustatus`(1M), `lutab`(4), `attributes`(5), `live_upgrade`(5)

WARNINGS

For adding packages or patches (`-p`, `-P`, `-t`, or `-T`), `luupgrade` requires packages or patches that comply with the SVR4 Advanced Packaging Guidelines and the guidelines spelled out in Appendix C of the *Solaris 9 Installation Guide*. This means that the package or patch is compliant with the `pkgadd`(1M) or `patchadd`(1M) `-R` option,

described in the man pages for those utilities. While nearly all Sun packages and patches conform to these guidelines, Sun cannot guarantee the conformance of packages and patches from third-party vendors. Some older Sun packages and patches might not be `-R` compliant. If you encounter such a package or patch, please report it to Sun. A non-conformant package can cause the package- or patch-addition software in `luupgrade` to fail or, worse, alter the current BE.

NAME

luxadm – administration program for the Sun Enterprise Network Array (SENA), RSM, SPARCstorage Array (SSA) subsystems, Sun Fire 880 internal storage subsystem, and individual Fiber Channel Arbitrated Loop (FC_AL) devices

SYNOPSIS

luxadm [*options*...] *subcommand* [*options*...] *enclosure* [,*dev*] | *pathname*...

DESCRIPTION

The luxadm program is an administrative command that manages the SENA, RSM, SPARCstorage Array subsystems, Sun Fire 880 internal storage subsystem, and individual FC_AL devices. luxadm performs a variety of control and query tasks depending on the command line arguments and options used.

The command line must contain a subcommand. The command line may also contain options, usually at least one enclosure name or pathname, and other parameters depending on the subcommand. You need specify only as many characters as are required to uniquely identify a subcommand.

Specify the device that a subcommand interacts with by entering a pathname. For the SENA subsystem, a disk device or enclosure services controller may instead be specified by entering the World Wide Name (WWN) for the device or a port to the device. The device may also be specified by entering the name of the SENA enclosure, and an optional identifier for the particular device in the enclosure. The individual FC_AL devices may be specified by entering the WWN for the device or a port to the device.

Pathname

Specify the device or controller by either a complete physical pathname or a complete logical pathname.

For SENA, a typical physical pathname for a device is:

```
/devices/sbus@1f,0/SUNW,socal@1,0/sf@0,0/ssd@w2200002037000f96,
        0:a,raw
```

or

```
/devices/io-unit@f,e0200000/sbi@0,0/SUNW,socal@2,0/sf@0,0/ssd@34,
        0:a,raw
```

For all SENA IBs (Interface Boards) and Sun Fire 880 SES device controllers on the system, a logical link to the physical paths is kept in the directory /dev/es. An example of a logical link is /dev/es.

The WWN may be used in place of the pathname to select an FC_AL device, SENA subsystem IB, or Sun Fire 880 internal storage subsystem. The WWN is a unique 16 hexadecimal digit value that specifies either the port used to access the device or the device itself. A typical WWN value is:

```
2200002037000f96
```

See NOTES for more information on the WWN formats.

For the SPARCstorage Array controller, a typical physical pathname is:

```
/devices/. . . /. . . /SUNW,soc@3,0/SUNW,pln@
     axxxxxxx,xxxxxxxx:ctlr
```

whereas, a typical physical pathname for an RSM controller is:

```
/devices/sbus@1f,0/QLGC,isp@1,10000:devctl
```

In order to make it easier to address the SPARCstorage Array or RSM controller, a logical pathname of the form c*N* is supported, where *N* is the logical controller number. `luxadm` uses the c*N* name to find an entry in the `/dev/rdsk` directory of a disk that is attached to the SPARCstorage Array or RSM controller. The `/dev/rdsk` entry is then used to determine the physical name of the SPARCstorage Array or RSM controller.

For a SPARCstorage Array disk, a typical physical pathname is:

```
/devices/. . . /. . . /SUNW,soc@3,0/SUNW,
     pln@axxxxxxx,xxxxxxxx/ssd@0,0:c,raw
```

and a typical logical pathname is:

```
/dev/rdsk/c1t0d0s2
```

For an RSM a typical physical pathname is:

```
/devices/sbus@1f,0/QLGC,isp@1,10000/sd@8,0:c,raw
```

and a typical logical pathname is:

```
/dev/rdsk/c2t8d0s2
```

For a disk in a Sun Fire 880 internal storage subsystem, a typical physical pathname is:

```
/devices/pci@8,600000/SUNW,qlc@2/fp@0,0/ssd@w2100002037a6303c,0:a
```

and a typical logical pathname is:

```
/dev/rdsk/c2t8d0s2
```

For individual FC_AL devices, a typical physical pathname is:

```
/devices/sbus@3.0/SUNW,socal@d,10000/sf@0,0/ssd@w2200002037049fc3,0:a,raw
```

and a typical logical pathname is:

```
/dev/rdsk/c1t0d0s2
```

Enclosure

For SENA, a device may be identified by its enclosure name and slotname:

box_name[,f*slot_number*]
box_name[,r*slot_number*]

box_name is the name of the SENA enclosure, as specified by the `enclosure_name` subcommand. When used without the optional *slot_number* parameter, the *box_name* identifies the SENA subsystem IB.

`f` or `r` specifies the front or rear slots in the SENA enclosure.

slot_number specifies the slot number of the device in the SENA enclosure, `0-6` or `0-10`.

For a Sun Fire 880 internal storage subsystem, a device may also be identified by its enclosure name and slot name. However, there is only one set of disks:

box_name[,s*slot_number*] *box_name* is the name of the Sun Fire 880 enclosure, as specified by the `enclosure_name` subcommand. When used without the optional *slot_number* parameter, *box_name* identifies the Sun Fire 880 internal storage subsystem enclosure services device. Use *s* to specify the disk slot number in the Sun Fire 880 internal storage subsystem, `0 - 11`.

See `disks`(1M) and `devlinks`(1M) for additional information on logical names for disks and subsystems.

OPTIONS

The following options are supported by all subcommands:

`-e` Expert mode. This option is not recommended for the novice user.

`-v` Verbose mode.

Options that are specific to particular subcommands are described with the subcommand in the `USAGE` section.

OPERANDS

The following operands are supported:

enclosure The box_name of the SENA. or Sun Fire 880 internal storage subsystem.

pathname The logical or physical path of a SENA IB,, Sun Fire 880 internal storage subsystem, SPARCstorage Array or RSM controller (`c`*N* name) or disk device. *pathname* can also be the WWN of a SENA IB, SENA disk, or individual FC_AL device.

Subcommands

`display` *enclosure*[,*dev*] . . . | *pathname* . . .
`display -p` *pathname* . . .
`display -r` *enclosure*[,*dev*] . . . | *pathname* . . .
`display -v` *enclosure*[,*dev*] . . . | *pathname* . . .
Displays enclosure or device specific data.

Subsystem data consists of enclosure environmental sense information and status for all subsystem devices, including disks.

Disk data consists of inquiry, capacity, and configuration information.

`-p`
Displays performance information for the device or subsystem specified by *pathname*. This option only applies to subsystems that accumulate performance information.

`-r`
Displays error information for the FC_AL device specified by the pathname, or, if the path is a SENA, for all devices on the loop. The `-r` option only applies to SENA subsystems and individual FC_AL devices.

`-v`
Displays in verbose mode, including mode sense data.

`download` [`-s`] [`-w` *WWN*] [`-f` *filename_path*] *enclosure*. . .
Download the prom image pointed to by *filename_path* to the SENA subsystem Interface Board unit, the Sun Fire 880 internal storage subsystem, or the SPARCstorage Array controllers specified by the enclosure or pathname. The SPARCstorage Array must be reset in order to use the downloaded code.

When the SENA's download is complete, the SENA will be reset and the downloaded code executed. If no filename is specified, the default prom image will be used. The default prom image for the SPARCstorage Array controller is in `usr/lib/firmware/ssa/ssafirmware`. The default prom image for the SENA is in the directory `usr/lib/locale/C/LC_MESSAGES` and is named `ibfirmware`

When the Sun Fire 880 internal storage subsystem's download is complete, the subsystem resets and the downloaded code begins execution. The default firmware image for the Sun Fire 880 internal storage subsystem is in: `/usr/platform/SUNW,Sun-Fire-880/lib/images/int_fcbpl_fw`.

`-s`
Save. The `-s` option is used to save the downloaded firmware in the FEPROM. If `-s` is not specified, the downloaded firmware will not be saved across power cycles.

The `-s` option does not apply to the SPARCstorage Array controller as it *always* writes the downloaded firmware into the FEPROM.

The `-s` option does not apply to the Sun Fire 880 internal storage subsystem as it always stores downloaded firmware in the flash memory.

When using the `-s` option, the `download` subcommand modifies the FEPROM on the subsystem and should be used with *caution*.

`-w` *WWN*
Change the SPARCstorage Array controller's World Wide Name. *WWN* is a 12-digit hex number; leading zeros are required. The `-w` option applies only to the SPARCstorage Array. The new SPARCstorage Array controller's image will have the least significant 6 bytes of the 8-byte World Wide Name modified to *WWN*.

`enclosure_name` *new_name enclosure* | *pathname*
Change the enclosure name of the enclosure or enclosures specified by the enclosure or pathname. The new name (*new_name*) must be 16 or less characters. Only alphabetic or numeric characters are acceptable. This subcommand applies only to the SENA and the Sun Fire 880 internal storage subsystem.

`failover primary | secondary` *pathname*
Select which Sun Storage T3 storage array partner group controller accesses a given logical volume. If `primary` is specified, the logical volume is accessed through the primary controller. If `secondary` is specified, the logical volume is accessed through the secondary controller specified by *pathname*.

`fc_s_download` [`-F`] [`-f` *fcode-file*]
Download the fcode contained in the file *fcode-file* into *all* the FC/S Sbus Cards. This command is interactive and expects user confirmation before downloading the fcode.

Use `fc_s_download` *only* in single-user mode. Using `fc_s_download` to update a host adapter while there is I/O activity through that adapter *will* cause the adapter to reset. Newly updated FCode will not be executed or visible until a system reboot.

`-f` *fcode-file*
When invoked without the `-f` *fcode-file* option, the current version of the fcode in each FC/S Sbus card is printed.

`-F`
Forcibly downloads the fcode, but the command still expects user confirmation before the download. The version of the FC/S Sbus Cards fcode that was released with this version of the Operating System is kept in the directory `usr/lib/firmware/fc_s` and is named `fc_s_fcode`.

`fcal_s_download` [`-f` *fcode-file*]
Download the fcode contained in the file *fcode-file* into *all* the FC100/S Sbus Cards. This command is interactive and expects user confirmation before downloading the fcode.

Use `fcal_s_download` *only* in single-user mode. Using `fcal_s_download` to update a host adapter while there is I/O activity through that adapter *will* cause the adapter to reset. Newly updated FCode will not be executed or visible until a system reboot.

`-f` *fcode-file*
When invoked without the `-f` option, the current version of the fcode in each FC100/S Sbus card is printed.

`fcode_download -p`
`fcode_download -d` *dir-name*
Locate the installed FC/S, FC100/S, FC100/P, or FC100/2P host bus adapter cards and download the FCode files in *dir-name* to the appropriate cards. The command determines the correct card for each type of file, and is interactive. User confirmation is required before downloading the FCode to each device.

Use `fcode_download` to load FCode only in single-user mode. Using `fcode_download` to update a host adapter while there is I/O activity through that adapter causes the adapter to reset. Newly updated FCode will not be executed or visible until a system reboot.

`-d` *dir-name*
Download the FCode files contained in the directory *dir-name* to the appropriate adapter cards.

`-p`
Prints the current version of FCode loaded on each card. No download is performed.

`inquiry` *enclosure*[,*dev*] . . . | *pathname* . . .
Display the inquiry information for the selected device specified by the enclosure or pathname.

`insert_device` [*enclosure*,*dev* . . .]
Assist the user in the hot insertion of a new device or a chain of new devices. Refer to `NOTES` for limitations on hotplug operations. This subcommand applies only to the SENA, Sun Fire 880 internal storage subsystem, RSM, and individual FC_AL drives. RSM, and individual FC_AL drives. For the SENA, if more than one enclosure has been specified, concurrent hot insertions on multiple busses can be performed. With no arguments to the subcommand, entire enclosures or individual FC_AL drives can be inserted. For the RSM, only one controller can be specified. For the SENA or the Sun Fire 880 internal storage subsystem, this subcommand guides the this subcommand guides the user interactively through the hot insertion steps of a new device or chain of devices. If a list of disks was entered it will ask the user to verify the list of devices to be inserted is correct, at which point the user can continue or quit. It then interactively asks the user to insert the disk(s) or enclosure(s) and then creates and displays the logical pathnames for the devices.

For the RSM, the following steps are taken:

- Quiesce the bus or buses which support quiescing and unquiescing.
- Inform the user that the device can be safely inserted .
- Request confirmation from the user that the device has been inserted.
- Unquiesce the bus or buses which support quiescing and unquiescing.
- Create the logical device name for the new device.

`led` *enclosure*,*dev* . . . | *pathname*. . .
Display the current state of the LED associated with the disk specified by the enclosure or pathname. This subcommand only applies to subsystems that support this functionality.

`led_blink` *enclosure,dev* . . . | *pathname* . . .
Requests the subsystem to start blinking the LED associated with the disk specified by the enclosure or pathname. This subcommand only applies to subsystems that support this functionality.

`led_off` *enclosure,dev* . . . | *pathname* . . .
Requests the subsystem to disable (turn off) the LED associated with the disk specified by the enclosure or pathname. On a SENA subsystem, this may or may not cause the LED to turn off or stop blinking depending on the state of the SENA subsystem. Refer to the SENA Array Installation and Service Manual (`p/n` 802-7573). This subcommand only applies to subsystems that support this functionality.

`led_on` *pathname* . . .
Requests the subsystem to enable (turn on) the LED associated with the disk specified by the pathname. This subcommand only applies to subsystems that support this functionality.

`power_off` [`-F`] *enclosure*[,*dev*] . . . | *pathname* . . .
`power_off` *pathname* [*enclosure-port*] . . . | *controller tray-number*
When a SENA is addressed, this subcommand causes the SENA subsystem to go into the power-save mode. The SENA drives are not available when in the power-save mode. When an Enclosure Services card within the SPARCstorage Array is addressed, the RSM tray is powered down. When a drive in a SENA is addressed the drive is set to the drive off/unmated state. In the drive off/unmated state, the drive is spun down (stopped) and in bypass mode. This command does not apply to the Sun Fire 880 internal storage subsystem.

`-F`
The force option only applies to the SENA. Instructs `luxadm` to attempt to power off one or more devices even if those devices are being used by this host (and are, therefore, busy).

Warning: Powering off a device which has data that is currently being used will cause unpredictable results. Users should attempt to power off the device normally (without `-F`) first, only resorting to this option when sure of the consequences of overriding normal checks.

`power_on` *enclosure*[, *dev*] . .
Causes the SENA subsystem to go out of the power-save mode, when this subcommand is addressed to a SENA. There is no programmatic way to power on the SPARCstorage Array RSM tray. When this subcommand is addressed to a drive the drive is set to its normal start-up state. This command does not apply to the Sun Fire 880 internal storage subsystem.

`probe` [`-p`]
Finds and displays information about all attached SENA subsystems, Sun Fire 880 internal storage subsystems, and individual FC_AL devices, including the logical pathname, the WWNs, and enclosure names. This subcommand warns the user if it finds different SENAs with the same enclosure names.

`-p`
Includes the physical pathname in the display.

`qlgc_s_download` [`-f` *fcode-file*]
Download the FCode contained in the file *fcode-file* into all the FC100/P, FC100/2P PCI host adapter cards. This command is interactive and expects user confirmation before downloading the FCode to each device. Only use `qlgc_s_download` in single-user mode. Using `qlgc_s_download` to update a host adapter while there is I/O activity through that adapter will cause the adapter to reset. Newly updated FCode will not be executed or visible until a system reboot.

`-f` *fcode-file*
When invoked without the `-f` option, the current version of the FCode in each FC100/P, FC100/2P PCI card is printed.

`release` *pathname*
Release a reservation held on the specified disk. The pathname should be the physical or logical pathname for the disk. If the pathname is of the SPARCstorage Array controller, then all of the disks in the SPARCstorage Array will be released.

This subcommand is included for historical and diagnostic purposes only.

`remove_device` [`-F`] *enclosure*[,*dev*] . . . | *pathname* . . .
Assists the user in hot removing a device or a chain of devices. This subcommand can also be used to remove entire enclosures. This subcommand applies to the SENA, Sun Fire 880 internal storage subsystem, RSM, and individual FC_AL drives. Refer to `NOTES` for limitations on hotplug operations. For the SENA, Sun Fire 880 internal storage subsystem, and individual FC_AL devices, this subcommand guides the user through the hot removal of a device or devices. During execution it will ask the user to verify the list of devices to be removed is correct, at which point the user can continue or quit. It then prepares the disk(s) or enclosure(s) for removal and interactively asks the user to remove the disk(s) or enclosure(s).

For the RSM, the steps taken are:

- Take the device offline.
- Quiesce the bus or buses which support quiescing and unquiescing.
- Inform user that the device can be safely removed.
- Request confirmation from the user that the device has been removed.
- Unquiesce the bus or buses which support quiescing and unquiescing.
- Bring the (now removed) device back online.
- Remove the logical device name for the removed device.

For Multi-Hosted disk, the steps taken are:

- Issue the `luxadm remove_device` command on the first host. When prompted to continue, wait.
- Issue the `luxadm remove_device` command on the secondary hosts. When prompted to continue, wait.

- Continue with the `remove_device` command on the first host. Remove the device when prompted to do so.
- Complete the `luxadm remove_device` command on the additional hosts.

`-F`
Instructs `luxadm` to attempt to hot plug one or more devices even if those devices are being used by this host (and are, therefore, *busy* or *reserved*), to *force* the hotplugging operation.

Warning: Removal of a device which has data that is currently being used will cause unpredictable results. Users should attempt to hotplug normally (without `-F`) first, only resorting to this option when sure of the consequences of overriding normal hotplugging checks.

`replace_device` [`-F`] *pathname*
This subcommand applies only to the RSM. Refer to `NOTES` for limitations on hotplug operations. This subcommand guides the user interactively through the hot replacement of a device.

For the RSM, the steps taken are:

- Take the device offline.
- Quiesce the bus or buses which support quiescing and unquiescing.
- Inform user that the device can be safely replaced.
- Request confirmation from the user that the device has been replaced.
- Unquiesce the bus or buses which support quiescing and unquiescing.
- Bring the device back online.

`-F`
Instructs `luxadm` to attempt to hot plug one or more devices even if those devices are *busy* or *reserved*, (that is, to *force* the hotplugging operation).

Warning: Removal of a device which has data that is currently being used will cause unpredictable results. Users should attempt to hotplug normally (without `-F`) first, only resorting to this option when sure of the consequences of overriding normal hotplugging checks.

`reserve` *pathname*
Reserve the specified disk for exclusive use by the issuing host. The pathname used should be the physical or logical pathname for the disk. If the pathname is of the SPARCstorage Array controller, then all of the disks in the SPARCstorage Array will be reserved.

This subcommand is included for historical and diagnostic purposes only.

`set_boot_dev` [`-y`] *pathname*
Set the boot-device variable in the system PROM to the physical device name specified by *pathname*, which can be a block special device or the pathname of the directory on which the boot file system is mounted. The command normally runs interactively requesting confirmation for setting the default boot-device in the PROM. The `-y` option can be used to run it non-interactively, in which case no confirmation is requested or required.

`start` [`-t` *tray-number*] *pathname* . . .
Spin up the specified disk(s). If *pathname* specifies the SPARCstorage Array controller, this action applies to all disks in the SPARCstorage Array.

`-t` Spin up all disks in the tray specified by tray-number. *pathname* must specify the SPARCstorage Array controller.

`stop` [`-t` *tray-number*] *pathname* . . .
Spin down the specified disk(s). If *pathname* specifies the SPARCstorage Array controller, this action applies to all disks in the SPARCstorage Array.

`-t`
Spin down all disks in the tray specified by tray-number. *pathname* must specify the SPARCstorage Array controller.

SPARCstorage Array Subcommands

`fast_write` [`-s`] `-c` *pathname*
`fast_write` [`-s`] `-d` *pathname*
`fast_write` [`-s`] `-e` *pathname*
Enable or disable the use of the NVRAM to enhance the performance of writes in the SPARCstorage Array. *pathname* refers to the SPARCstorage Array controller or to an individual disk.

`-s`
Cause the SPARCstorage Array to save the change so it will persist across power-cycles.

`-c`
Enable fast writes for synchronous writes only.

`-d`
Disable fast writes.

`-e`
Enable fast writes.

`nvram_data` *pathname*
Display the amount of fast write data in the NVRAM for the specified disk. This command can only be used for an individual disk.

`perf_statistics -d` *pathname*
`perf_statistics -e` *pathname*
Enable or disable the accumulation of performance statistics for the specified SPARCstorage Array controller. The accumulation of performance statistics must be enabled before using the display `-p` subcommand. This subcommand can be issued only to the SPARCstorage Array controller.

`-d`
Disable the accumulation of performance statistics.

`-e`
Enable the accumulation of performance statistics.

`purge` *pathname*
Purge any fast write data from NVRAM for one disk, or all disks if the controller is specified. This option should be used with caution, usually only when a drive has failed.

`sync_cache` *pathname*
Flush all outstanding writes for the specified disk from NVRAM to the media. If pathname specifies the controller, this action applies to all disks in the SPARCstorage Array subsystem.

Enclosure Services Card Subcommands

The `env_display` and `alarm*` subcommands apply only to an Enclosure Services Card (SES) in a RSM tray in a SPARCstorage Array. The RSM tray is addressed by using the logical or physical path of the SES device or by specifying the controller followed by the tray number. The controller is addressed by `c`*N* or the physical path to the SSA's controller.

`alarm` *pathname* | *controller tray_number*
Display the current state of audible alarm.

`alarm_off` *pathname* | *controller tray_number*
Disable the audible alarm for this RSM tray.

`alarm_on` *pathname* | *controller tray_number*
Enable the audible alarm for this RSM tray.

`alarm_set` *controller-pathname* | *controller tray_number* [*seconds*]
Set the audible alarm setting to seconds.

`env_display` *pathname* | *controller tray_number*
Display the environmental information for the specified unit.

SENA, Sun Fire 880 Internal Storage Subsystem, and Individual FC_AL Drive Expert Mode Subcommands

The following subcommands are for expert use only, and are applicable only to the SENA, Sun Fire 880 internal storage subsystem, and fiber channel loops. They should only be used by users that are knowledgeable about the SENA subsystem and fiber channel loops.

If you specify a disk to an expert subcommand that operates on a bus, the subcommand operates on the bus to which the specified disk is attached.

`-e forcelip` *enclosure*[, *dev*] `...` | *pathname* . . .
Force the link to reinitialize, using the Loop Initialization Primitive (LIP) sequence. The enclosure or pathname can specify any device on the loop. Use the pathname to specify a specific path for multiple loop configurations.

This is an expert only command and should be used with caution. It will reset all ports on the loop.

`-e rdls` *enclosure*[, *dev*] `...` | *pathname* . . .
Read and display the link error status information for all available devices on the loop that contains the device specified by the enclosure or pathname.

Other Expert Mode Subcommands

See `NOTES` for limitations of these subcommands. They should only be used by users that are knowledgeable about the systems they are managing.

These commands do not apply to the Sun Fire 880 internal storage subsystem.

`-e bus_getstate` *pathname*
Get and display the state of the specified bus.

`-e bus_quiesce` *pathname*
Quiesce the specified bus.

`-e bus_reset` *pathname*
Reset the specified bus only.

`-e bus_resetall` *pathname*
Reset the specified bus and all devices.

`-e bus_unquiesce` *pathname*
Unquiesce the specified bus. the specified device.

`-e dev_getstate` *pathname*
Get and display the state of the specified device.

`-e dev_reset` *pathname*
Reset the specified device.

`-e offline` *pathname*
Take the specified device offline.

`-e online` *pathname*
Put the specified device online.

EXAMPLES

EXAMPLE 1 Displaying the SENAs and Individual FC_AL Devices on a System

The following example finds and displays all of the SENAs and individual FC_AL devices on a system:

```
example% luxadm probe
```

EXAMPLE 2 Displaying an SSA

The following example displays an SSA:

```
example% luxadm display c1
```

EXAMPLE 3 Displaying a SENA or Sun Fire 880 Internal Storage Subsystem

The following example displays a SENA or Sun Fire 880 internal storage subsystem:

```
example% luxadm display /dev/es/ses0
```

EXAMPLE 4 Displaying Two Subsystems

The following example displays two subsystems using the enclosure names:

```
example% luxadm display BOB system1
```

EXAMPLE 5 Displaying Information about the First Disk

The following example displays information about the first disk in the front of the enclosure named `BOB`. Use `f` to specify the front disks. Use `r` to specify the rear disks.

EXAMPLE 5 Displaying Information about the First Disk *(Continued)*

```
example% luxadm display BOB,f0
```

EXAMPLE 6 Displaying Information on a Sun Fire 880 Internal Storage Subsystem

The Sun Fire 880 internal storage subsystem has only one set of disks. In this case, use *s* to specify the slot:

```
example% luxadm display BOB,s0
```

EXAMPLE 7 Displaying Information about a SENA disk, an Enclosure, or an Individual FC_AL Drive

The following example displays information about a SENA disk, an enclosure, or an individual FC_AL drive with the port WWN of `2200002037001246`:

```
example% luxadm display 2200002037001246
```

EXAMPLE 8 Using Unique Characters to Issue a Subcommand

The following example uses only as many characters as are required to uniquely identify a subcommand:

```
example% luxadm disp BOB
```

EXAMPLE 9 Displaying Error Information

The following example displays error information about the loop that the enclosure `BOB` is on:

```
example% luxadm display -r BOB
```

EXAMPLE 10 Downloading New Firmware into the Interface Board

The following example downloads new firmware into the Interface Board in the enclosure named `BOB` (using the default path for the file to download):

```
example% luxadm download -s BOB
```

EXAMPLE 11 Displaying Information from the SCSI Inquiry Command

The following example displays information from the SCSI inquiry command from all individual disks on the system, using only as many characters as necessary to uniquely identify the inquiry subcommand:

```
example% luxadm inq /dev/rdsk/c?t?d?s2
```

EXAMPLE 12 Hotplugging

The following example hotplugs a new drive into the first slot in the front of the enclosure named `BOB`:

EXAMPLE 12 Hotplugging *(Continued)*

```
example% luxadm insert_device BOB,f0
```

The following example hotplugs a new drive into the first slot in the Sun Fire 880 internal storage subsystem named SF880-1:

```
example% luxadm insert_device SF880-1,s0
```

EXAMPLE 13 Running an Expert Subcommand

The following example runs an expert subcommand. The subcommand forces a loop initialization on the loop that the enclosure `BOB` is on:

```
example% luxadm -e forcelip BOB
```

EXAMPLE 14 Using the Expert Mode Hot Plugging Subcommands

An example of using the expert mode hot plugging subcommands to hot remove a disk on a SSA follows. See `NOTES` for hot plugging limitations.

The first step reserves the SCSI device so that it can't be accessed by way of its second SCSI bus:

```
example# luxadm reserve /dev/rdsk/c1t8d0s2
```

EXAMPLE 15 Taking the Disk to be Removed Offline

The next two steps take the disk to be removed offline then quiesce the bus:

```
example# luxadm -e offline /dev/rdsk/c1t8d0s2
example# luxadm -e bus_quiesce /dev/rdsk/c1t8d0s2
```

EXAMPLE 16 Unquiescing the Bus

The user then removes the disk and continues by unquiescing the bus, putting the disk back online, then unreserving it:

```
example# luxadm -e bus_unquiesce /dev/rdsk/c1t8d0s2
example# luxadm -e online /dev/rdsk/c1t8d0s2
example# luxadm release /dev/rdsk/c1t8d0s2
```

ENVIRONMENT VARIABLES

See `environ`(5) for a description of the `LANG` environment variable that affects the execution of `luxadm`.

EXIT STATUS

The following exit values are returned:

`0`	Successful completion.
`−1`	An error occurred.

FILES

`usr/lib/firmware/fc_s/fc_s_fcode`

`usr/lib/firmware/ssa/ssafirmware`

`usr/lib/locale/C/LC_MESSAGES/ibfirmware`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

usr/sbin

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWluxop

SEE ALSO

`devlinks`(1M), `disks`(1M), `ssaadm`(1M), `attributes`(5), `environ`(5), `ses`(7D)

SENA Array Installation and Service Manual (p/n 802-7573).

RAID Manager 6.1 Installation and Support Guide Answerbook

RAID Manager 6.1 User's Guide Answerbook

NOTES

See the *SENA Array Installation and Service Manual* for additional information on the SENA. Refer to *Tutorial for SCSI use of IEEE Company_ID*, R. Snively, for additional information regarding the IEEE extended WWN. See `SEE ALSO`. Currently, only some device drivers support hot plugging. If hot plugging is attempted on a disk or bus where it is not supported, an error message of the form:

```
luxadm: can't acquire "PATHNAME": No such file or directory
```

will be displayed.

You must be careful not to quiesce a bus that contains the root or the `/usr` filesystems or any swap data. If you do quiesce such a bus a deadlock can result, requiring a system reboot.

NAME

m64config, SUNWm64_config – configure the M64 Graphics Accelerator

SYNOPSIS

/usr/sbin/m64config [-defaults] [-depth 8 | 24 | 32] [-dev *device-filename*] [-file machine | system] [-prconf] [-propt] [-res *video-mode* [now | try] [noconfirm | nocheck]]

/usr/sbin/m64config [-prconf] [-propt]

/usr/sbin/m64config [-help] [-res ?]

DESCRIPTION

`m64config` configures the M64 Graphics Accelerator and some of the X11 window system defaults for M64.

The first form of `m64config` stores the specified options in the OWconfig file. These options will be used to initialize the M64 device the next time the window system is run on that device. Updating options in the OWconfig file provides persistence of these options across window system sessions and system reboots.

The second and third forms which invoke only the `-prconf`, `-propt`, `-help`, and `-res` ? options do not update the OWconfig file. Additionally, for the third form all other options are ignored.

Options may be specified for only one M64 device at a time. Specifying options for multiple M64 devices requires multiple invocations of `m64config`.

Only M64-specific options can be specified through `m64config`. The normal window system options for specifying default depth, default visual class and so forth are still specified as device modifiers on the `openwin` command line. See the *OpenWindows Desktop Reference Manual* for details.

The user can also specify the OWconfig file that is to be updated. By default, the machine-specific file in the `/etc/openwin` directory tree is updated. The `-file` option can be used to specify an alternate file to use. For example, the system-global OWconfig file in the `/usr/openwin` directory tree can be updated instead.

Both of these standard OWconfig files can only be written by root. Consequently, the `m64config` program, which is owned by the root user, always runs with `setuid` root permission.

OPTIONS

`-defaults`
Resets all option values to their default values.

`-depth 8 | 24 | 32`
Sets the depth (bits per pixel) on the device. Possible values are `8`, `24`, or `32` (where `32` uses 24 bits per pixel). Log out of the current window system session and log back in for the change to take effect. `24` or `32` enables TrueColor graphics in the window system, at the expense of screen resolution.

The `32` setting enables simultaneous 8– and 24–bit color windows on `m64` devices that support it. With setting `32`, `-propt` shows depth 32 and `-prconf` shows depth 24. To check window depth, use the `xwininfo` utility. The `xwininfo` utility is usually shipped in the package containing frame buffer software (such as `SUNWxwplt`).

The maximum resolution that is available with 24 bits per pixel depends on the amount of memory installed on the PGX card. For 2-MB PGX cards, the maximum available resolution is 800x600. For 4-MB cards, it is 1152x900. For 8-MB cards, it is 1920x1080. If there is not enough memory for the specified combination of resolution and depth, `m64config` displays an error message and exits.

`-dev` *device-filename*
Specifies the M64 special file. If not specified, `m64config` will try `/dev/fbs/m640` through `/dev/fbs/m648` until one is found.

`-file machine|system`
Specifies which OWconfig file to update. If `machine`, the machine-specific OWconfig file in the `/etc/openwin` directory tree is used. If `system`, the global OWconfig file in the `/usr/openwin` directory tree is used. If the file does not exist, it is created.

`-help`
Prints a list of the `m64config` command line options, along with a brief explanation of each.

`-prconf`
Prints the M64 hardware configuration. The following is a typical display using the `-prconf` option:

```
--- Hardware Configuration for /dev/fbs/m640 ---
ASIC: version 0x41004754
DAC: version 0x0
PROM: version 0x0
Card possible resolutions:  640x480x60, 800x600x75, 1024x768x60
    1024x768x70, 1024x768x75, 1280x1024x75, 1280x1024x76
    1280x1024x60, 1152x900x66, 1152x900x76, 1280x1024x67
    960x680x112S, 960x680x108S, 640x480x60i, 768x575x50i, 1280x800x76
    1440x900x76, 1600x1000x66, 1600x1000x76, vga, svga, 1152, 1280
    stereo, ntsc, pal
Monitor possible resolutions:  720x400x70, 720x400x88, 640x480x60
    640x480x67, 640x480x72, 640x480x75, 800x600x56, 800x600x60
    800x600x72, 800x600x75, 832x624x75, 1024x768x87, 1024x768x60
    1024x768x70, 1024x768x75, 1280x1024x75, 1280x1024x76, 1152x900x66
    1152x900x76, 1280x1024x67, 960x680x112S, vga, svga, 1152, 1280
    stereo
Possible depths: 8, 24
Current resolution setting: 1280x1024x76
Current depth: 8
```

`-propt`
Prints the current values of all M64 options in the OWconfig file specified by the `-file` option for the device specified by the `-dev` option. Prints the values of

options as they will be in the OWconfig file after the call to `m64config` completes. The following is a typical display using the `-propt` option:

```
--- OpenWindows Configuration for /dev/fbs/m640 ---
OWconfig: machine
Video Mode: not set
Depth: 8
```

`-res` *video-mode* [`now` | `try` [`noconfirm` | `nocheck`]]

Specifies the video mode used to drive the monitor connected to the specified M64 device. Video modes are built-in. *video-mode* has the format of *width*x*height*x*rate*. *width* is the screen width in pixels, *height* is the screen height in pixels, and *rate* is the vertical frequency of the screen refresh. As a convenience, `-res` also accepts formats with `@` preceding the refresh rate instead of `x`. For example, `1280x1024@76`.

A list of valid video modes is obtained by issuing the following command: `m64config -res '?'`. Note that the `?` must be quoted. Not all resolutions are supported by both the video board and by the monitor. `m64config` will not permit you to set a resolution the board does not support, and will request confirmation before setting a resolution the monitor does not support.

`Symbolic names`

For convenience, some video modes have symbolic names defined for them. Instead of the form *width*x*height*x*rate*, one of these names may be supplied as the argument to `-res`. The meaning of the symbolic name `none` is that when the window system is run the screen resolution will be the video mode that is currently programmed in the device.

Name	Corresponding Video Mode
`svga`	1024x768x60
`1152`	1152x900x76
`1280`	1280x1024x76
`none`	(video mode currently programmed in device)

The `-res` option also accepts additional sub-options immediately following the video mode specification. Any or all of these may be present.

`nocheck` If present, the normal error checking based on the monitor sense code will be suspended. The video mode specified by the user will be accepted regardless of whether it is appropriate for the currently attached monitor. This option is useful if a different monitor is to be connected to the M64 device. *Use of this option implies* `noconfirm` *as well*.

`noconfirm` Using the `-res` option, the user could potentially put the system into an usable state, a state where there is no video output. This can happen if there is ambiguity in the monitor sense codes for the particular code read. To reduce the chance of this, the default behavior of `m64config` is to print a warning message to this effect and to prompt the user to find out if it is okay to continue. The noconfirm option instructs `m64config` to bypass this confirmation and to program the requested video mode anyway. This option is useful when `m64config` is being run from a shell script.

`now` If present, not only will the video mode be updated in the OWconfig file, but the M64 device will be immediately programmed to display this video mode. (This is useful for changing the video mode before starting the window system).

It is inadvisable to use this sub-option with `m64config` while the configured device is being used (for example, while running the window system); unpredictable results may occur. To run `m64config` with the `now` sub-option, first bring the window system down. If the `now` sub-option is used within a window system session, the video mode will be changed immediately, but the width and height of the affected screen won't change until the window system is exited and reentered again. Consequently, this usage is strongly discouraged.

`try` If present, the specified video mode will be programmed on a trial basis. The user will be asked to confirm the video mode by typing `y` within 10 seconds. Or the user may terminate the trial before 10 seconds are up by typing any character. Any character other than 'y' or carriage return is considered a no and the previous video mode will be restored and `m64config` will not change the video mode in the OWconfig file (other options specified will still take effect). If a carriage return is typed, the user is prompted for a yes or no answer on whether to keep the new video mode. This option implies the `now` sub-option. (See the warning note on the `now` sub-option).

DEFAULTS

For a given invocation of `m64config` command line if an option does not appear on the command line, the corresponding OWconfig option is not updated; it retains its previous value.

When the window system is run, if an M64 option has never been specified by `m64config`, a default value is used. The option defaults are as follows:

Option	Default
`-dev`	`/dev/fbs/m640`

`-file`	machine
`-res`	none

The default for the `-res` option of none means that when the window system is run the screen resolution will be the video mode that is currently programmed in the device.

This provides compatibility for users who are used to specifying the device resolution through the PROM. On some devices (for example, GX) this is the only way of specifying the video mode. This means that the PROM ultimately determines the default M64 video mode.

EXAMPLES

EXAMPLE 1 Switching the Monitor Type

The following example switches the monitor type to the maximum resolution of `1280` x `1024` at `76` Hz:

```
example% /usr/sbin/m64config -res 1280x1024x76
```

FILES

`/dev/fbs/m640`
device special file

`/etc/openwin/server/etc/OWconfig`
system config file

`/usr/lib/fbconfig/SUNWm64_config`
symbolic link to `usr/sbin/m64config`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWm64cf

SEE ALSO

`attributes`(5), `m64`(7D)

OpenWindows Desktop Reference Manual

NAME

mail.local – store mail in a mailbox

SYNOPSIS

/usr/lib/mail.local [-f *sender*] [-d] *recipient*

DESCRIPTION

`mail.local` reads the standard input up to an end-of-file and appends it to each user's mail file (mailbox). This program is intended to be used by `sendmail`(1M) as a mail delivery agent for local mail. It is not a user interface agent.

Messages are appended to the user's mail file in the `/var/mail` directory. The user must be a valid user name.

Each delivered mail message in the mailbox is preceded by a "Unix From line" with the following format:

`From` *sender_address time_stamp*

The *sender_address* is extracted from the SMTP envelope address (the envelope address is specified with the `-f` option).

A trailing blank line is also added to the end of each message.

The mail files are locked with a `.lock` file while mail is appended.

The mail files are created with mode `660`, owner is set to *recipient*, and group is set to `mail`. If the ''biff'' service is returned by `getservbyname`(3SOCKET), the biff server is notified of delivered mail. This program also computes the `Content-Length:` header which will be used by the mailbox reader to mark the message boundary.

OPTIONS

The following options are supported:

`-f` *sender*	Specifies the "envelope from address" of the message. This flag is technically optional, but should be used.
`-d`	Specifies the recipient of the message. This flag is also optional and is supported here for backward compatibility. That is, `mail.local` *recipient* is the same as `mail.local -d` *recipient*.
`-l`	Turn on LMTP mode.
`-r` *from*	Specify the sender's name (for backward compatibility).
`-7`	Do not advertise 8BITMIME support in LMTP mode.
`-b`	Return a permanent error instead of a temporary error if a mailbox exceeds quota.

OPERANDS

The following operand is supported:

recipient	The recipient of the mail message.

ENVIRONMENT VARIABLES

`TZ` Used to set the appropriate time zone on the timestamp.

EXIT STATUS

The following exit values are returned:

`0`	Successful operation.
`>0`	An error occurred.

FILES

`/tmp/local.`*XXXXXX*	temporary files
`/tmp/lochd.`*XXXXXX*	temporary files
`/var/mail/`*user_name*	user's mail file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsndmu

SEE ALSO

`mail`(1), `comsat`(1M), `sendmail`(1M), `getservbyname`(3SOCKET), `attributes`(5)

NAME

makedbm – make a dbm file, or get a text file from a dbm file

SYNOPSIS

```
makedbm [-b] [-l] [-s] [-E] [-i yp_input_file] [-o yp_output_name]
     [-d yp_domain_name] [-m yp_master_name] [-S delimiter]
     [-D number_of_delimiters] infile outfile

makedbm [-u dbmfilename]
```

DESCRIPTION

The `makedbm` utility takes the *infile* and converts it to a pair of files in `ndbm` format (see `ndbm`(3C)), namely *outfile*`.pag` and *outfile*`.dir`. Each line of the input file is converted to a single `dbm` record. All characters up to the first TAB or SPACE form the key, and the rest of the line is the data. If a line ends with '\' (backslash), the data for that record is continued on to the next line. `makedbm` does not treat '#' (pound-sign) as a special character.

Because `makedbm` is mainly used in generating `dbm` files for the NIS name service, it generates a special entry with the key *yp_last_modified*, which is the date of *infile* (or the current time, if *infile* is '–'). The entries that have keys with the prefix *yp_* are interpreted by NIS server utilities.

OPTIONS

The following options are supported:

`-b`	Insert the `YP_INTERDOMAIN` into the output. This key causes `ypserv`(1M) to use DNS for host name and address lookups for hosts not found in the maps.
`-d` *yp_domain_name*	Create a special entry with the key *yp_domain_name*.
`-D` *number_of delimiters*	Specify *number_of_delimiters* to skip before forming the key.
`-E`	Delimiters are escaped.
`-i` *yp_input_file*	Create a special entry with the key *yp_input_file*.
`-l`	Lower case. Convert the keys of the given map to lower case, so that, for example, host name matches succeed independent of upper or lower case distinctions.
`-m` *yp_master_name*	Create a special entry with the key *yp_master_name*. If no master host name is specified, *yp_master_name* is set to the local host name.
`-o` *yp_output_name*	Create a special entry with the key *yp_output_name*.
`-s`	Secure map. Accept connections from secure NIS networks only.
`-S` *delimiter*	Specify the *delimiter* to use instead of the default delimiter for forming the key.
`-u` *dbmfilename*	Undo a `dbm` file. Prints out the file in text format, one entry per line, with a single space separating keys from

values.

OPERANDS

The following operands are supported:

infile Input file for `makedbm`. If *infile* is '–' (dash), the standard input is read.

outfile One of two output files in `ndbm` format: *outfile*`.pag` and *outfile*`.dir`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`ypserv`(1M), `ndbm`(3C), `attributes`(5)

NAME

makemap – create database maps for sendmail

SYNOPSIS

```
makemap [-N] [-d] [-f] [-o] [-r] [-s] [-v] [-C file] [-c cachesize]
     [-e] [-l] [-t delim] [-u] mantype mapname
```

DESCRIPTION

`makemap` creates the database maps used by the keyed map lookups in `sendmail`(1M). `makemap` reads from the standard input and outputs to the specified *mapname*.

In all cases, `makemap` reads lines from the standard input consisting of two words separated by white space. The first is the database key, the second is the value. The value may contain `%`*n* strings to indicated parameter substitution. Literal percents should be doubled (`%%`). Blank lines and lines beginning with # are ignored.

`makemap` handles three different database formats. Database format is selected using the *maptype* parameter. See `OPERANDS`.

OPTIONS

The following options are supported:

`-c` *cachesize*	Use the specified hash and B-Tree cache size (*cachesize*).
`-C` *file*	Use the specified sendmail configuration file (*file*) for looking up the TrustedUser option.
`-d`	Allow duplicate keys in the map. This is only allowed on B-Tree format maps. If two identical keys are read, both be inserted into the map.
`-e`	Allow empty value (right hand side).
`-f`	Normally all upper case letters in the key are folded to lower case. This flag disables that behavior. This is intended to mesh with the `-f` flag in the `K` line in `sendmail.cf`. The value is never case folded.
`-l`	List supported map types.
`-N`	Include the null byte that terminates strings in the map. This must match the `-N` flag in the `K` line in `sendmail.cf`
`-o`	Append to an old file. This allows you to augment an existing file.
`-r`	Allow replacement of existing keys. Normally `makemap` complains if you repeat a key, and does not do the insert.
`-s`	Ignore safety checks on maps being created. This includes checking for hard or symbolic links in world writable directories.
`-t` *delim*	Use the specified delimiter (*delim*) instead of white space.
`-u`	Dump (unmap) the content of the database to standard output.
`-v`	Verbosely print what it is doing.

OPERANDS

The following operands are supported:

mapname File name of the database map being created.

maptype Specifies the database format. The following *maptype* parameters are available:

`dbm` Specifies DBM format maps.

`btree` Specifies B-Tree format maps.

`hash` Specifies hash format maps.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsndmu

SEE ALSO

`editmap`(1M), `sendmail`(1M), `attributes`(5)

NAME

makeuuid – generate Universal Unique Identifiers

SYNOPSIS

makeuuid [-e *ether*] [-n *count*] [-R *root*]

DESCRIPTION

The makeuuid command generates UUIDs (Universal Unique Identifiers) conforming to the OSF DCE specification for UUIDs. The specification states:

"A UUID is an identifier that is unique across both space and time, with respect to the space of all UUIDs. A UUID can be used for multiple purposes, from tagging objects with an extremely short lifetime, to reliably identifying very persistent objects across a network.

“The generation of UUIDs does not require a registration authority for each single identifier. Instead, it requires a unique value over space for each UUID generator. This spatially unique value is [normally] specified as an IEEE 802 address, which is usually already applied to network-connected systems."

The makeuuid command generates one or more UUIDs on the standard output.

OPTIONS

The makeuuid command supports the following options:

-e *ether*
: Supplies an alternate address to be used in the generation of the UUIDs. Normally, the system’s Ethernet address is acquired and used during the generation of a UUID. However, this requires root privileges to open and read the network devices. If this is not possible, you must supply an alternate Ethernet address.

-n *count*
: Generate multiple UUIDs. This option generates the specified number of UUIDs, one per line. Using this form is more efficient than, and functionally equivalent to, calling the makeuuid command multiple times. This can be used, for example, when a large number of UUIDs need to be generated for a given application.

-R *root*
: Use *root* as the root filesystem path when updating the shared state file (see FILES). The shared state file must be writable by the user running makeuuid, otherwise no UUIDs will be generated and the command will return in failure.

USAGE

Normally, you run the makeuuid command with root privileges, as the Ethernet address and state files can be easily accessed and updated. If this is not possible, you must use the -R and -e options to specify an alternate root and Ethernet address to use when calculating the UUIDs.

EXAMPLES

EXAMPLE 1 Generating Multiple UUIDs

The following command generates 3000 UUIDs:

```
example# makeuuid -n 3000
```

EXAMPLE 2 Invoking Without Root Privileges

If you cannot obtain root privileges, you must specify an alternate Ethernet address and state file location:

EXAMPLE 2 Invoking Without Root Privileges *(Continued)*

```
example% makeuuid -e 11:22:33:44:55:66 -R /export/root/example2
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` Out of memory.

`-1` Invalid Ethernet address given or access denied.

FILES

`/var/sadm/system/uuid_state` UUID state file. Use of time values is one way that UUID generators, such as `makeuuid`, guarantee uniqueness. A state file is a mechanism that allows `makeuuid` to "remember" the last time value it used so it can increment that value for use in a new UUID. See the Internet Draft "UUIDs and GUIDs," dated February 4, 1998, for details on the state file mechanism.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWwsr2

SEE ALSO

`prodreg`(1M), `intro`(3), `libwsreg`(3LIB), `attributes`(5)

NOTES

The formal UUID specification is in the OSF DCE specification, available at `www.opengroup.org`. As of the date of publication of this man page, a copy of the specification is available at:

`http://www.opengroup.org/onlinepubs/9629399/apdxa.htm`

Sun has no control over the availability of documents on the `www.opengroup.org` web site.

NAME

mdmonitord – daemon to monitor metadevices

SYNOPSIS

/usr/sbin/mdmonitord [-t *time_interval*]

DESCRIPTION

The mdmonitord utility is part of Solaris Volume Manager. It checks the accessibility of the physical components of a metadevice. There are two methods for checking metadevices:

- At fixed time intervals.
- When a metadevice fails. Metadevice failure generates an error event which triggers a check of all metadevices.

OPTIONS

-t Time interval in seconds. Recommended probe interval 1800 seconds (1/2 hour). The default value is 0, which causes probes to occur only upon an error.

EXIT STATUS

The following exit values are returned:

0 Successful completion.

>0 An error occurred.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdu

SEE ALSO

metaclear(1M), metadb(1M), metadetach(1M), metahs(1M), metainit(1M), metaoffline(1M), metaonline(1M), metaparam(1M), metareplace(1M), metaroot(1M), metaset(1M), metastat(1M), metattach(1M), md.tab(4), md.cf(4), mddb.cf(4), attributes(5)

Solaris Volume Manager Administration Guide

NOTES

Since frequent probes can affect performance, it is recommended that the intervals between probes be limited.

This daemon is started at boot time by S95svm.sync in /etc/rc2.d.

NAME

medstat – check the status of mediator hosts for a given diskset

SYNOPSIS

/usr/sbin/medstat [-q] -s *setname*

DESCRIPTION

If a specified diskset has been configured for mediators, `medstat` attempts to contact these hosts to see if they are accessible and returns the results of the communication.

OPTIONS

`-q` This optional argument disables the printing of informative text. When used with `-q`, `medstat` still prints error messages and returns a result code.

`-s` *setname* Specifies the name of a diskset on which `medstat` will work.

EXAMPLES

EXAMPLE 1 Checking diskset

This example checks the mediator hosts for the selected diskset.

```
# medstat -s relo-red
```

The name of the diskset is `relo-red`. The `medstat` command prints the status for each mediator host. Additionally, if the mediator quorum is met, either through a ''golden'' mediator host or because half+1 of the mediator hosts respond, the exit code is 0. If the quorum is not met, then the exit code is 1. If no mediator hosts have been configured for the named diskset, the exit code is 2. The status field will contain one of the following values: Unreachable, Bad, Fatal, or Ok, where Unreachable indicates an RPC/communication problem, Bad indicates an error in the mediator data, Fatal indicates any other error condition, and Ok indicates no error conditions.

FILES

`/etc/lvm/meddb` Contains the mediator data for a host that has been selected as a mediator host.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

>0 An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdu
Interface Stability	Evolving

SEE ALSO

`metaclear`(1M), `metadb`(1M), `metadetach`(1M), `metahs`(1M), `metainit`(1M), `metaoffline`(1M), `metaonline`(1M), `metaparam`(1M), `metareplace`(1M), `metaroot`(1M), `metaset`(1M), `metastat`(1M), `metasync`(1M), `metattach`(1M), `md.cf`(4), `md.tab`(4), `mddb.cf`(4), `meddb`(4), `mediator`(7D)

Sun Cluster documentation, *Solaris Volume Manager Administration Guide*

NOTES

This command is designed for use in the high availability product.

NAME

metaclear – delete active metadevices and hot spare pools

SYNOPSIS

/usr/sbin/metaclear `-h`

/usr/sbin/metaclear `[-s` *setname*`] -a [-f]`

/usr/sbin/metaclear *component*

/usr/sbin/metaclear `[-s` *setname*`] [-f]` *metadevice... hot_spare_pool...*

/usr/sbin/metaclear `[-s` *setname*`] -r [-f]` *metadevice... hot_spare_pool...*

/usr/sbin/metaclear `[-s` *setname*`] -p` *component*

/usr/sbin/metaclear `[-s` *setname*`] -p` *metadevice*

DESCRIPTION

The `metaclear` command deletes the specified metadevice or *hot_spare_pool.*, or purges all soft partitions from the designated component. Once a metadevice or hot spare pool is deleted, it must be re-created using `metainit` before it can be used again.

Any metadevice currently in use (open) cannot be deleted.

OPTIONS

Root privileges are required for all of the following options except `-h`.

`-a`	Deletes all metadevices and configured hot spare pools in the set named by `-s`, or the local set by default.
`-f`	Deletes (forcibly) a metadevice that contains a subcomponent in an error state.
`-h`	Displays usage message.
`-p`	Deletes (purges) all soft partitions from the specified metadevice or component.
`-r`	Recursively deletes specified metadevices and hot spare pools, but does not delete metadevices on which others depend.
`-s` *setname*	Specifies the name of the diskset on which `metaclear` will work. Using the `-s` option causes the command to perform its administrative function within the specified diskset. Without this option, the command performs its function on local metadevices and/or hot spare pools.

OPERANDS

metadevice ...	Specifies the name(s) of the metadevice(s) to be deleted.
component	Specifies the `c*d*t*s*` name(s) of the components containing soft partitions to be deleted.
hot_spare_pool ...	Specifies the name(s) of the hot spare pools to be deleted in the form `hsp`*nnn*, where *nnn* is a number in the range 000-999.

EXAMPLES

EXAMPLE 1 Deleting Various Devices

The following example deletes a metadevice named `d10`.

```
# metaclear /dev/md/dsk/d10
```

The following example deletes all local metadevices and hot spare pools on the system.

```
# metaclear -a
```

The following example deletes a mirror, `d20`, with an submirror in an error state.

```
# metaclear -f d20
```

The following example deletes a hot spare pool, `hsp001`.

```
# metaclear hsp001
```

The following example deletes a soft partition, `d23`.

```
# metaclear d23
```

The following example purges all soft partitions on the slice `c2t3d5s2` if those partitions are not being used by other metadevices or are not open.

```
# metaclear -p c2t3d5s2
```

The following example purges soft partitions from a metadevice.

```
# metaclear -p d2
d3: Soft Partition is cleared
d4: Soft Partition is cleared
d5: Soft Partition is cleared
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdu

SEE ALSO

`metadb`(1M), `metadetach`(1M), `metahs`(1M), `metainit`(1M), `metaoffline`(1M), `metaonline`(1M), `metaparam`(1M), `metarecover`(1M), `metareplace`(1M), `metaroot`(1M), `metaset`(1M), `metastat`(1M), `metasync`(1M), `metattach`(1M), `md.tab`(4), `md.cf`(4), `mddb.cf`(4), `attributes`(5)

Solaris Volume Manager Administration Guide

NAME

metadb – create and delete replicas of the metadevice state database

SYNOPSIS

/sbin/metadb `-h`

/sbin/metadb `[-s` *setname*`]`

/sbin/metadb `[-s` *setname*`] -a [-f] [-k` *system-file*`] mddb` *nnn*

/sbin/metadb `[-s` *setname*`] -a [-f] [-k` *system-file*`] [-c` *number*`] [-l` *length*`]` *slice...*

/sbin/metadb `[-s` *setname*`] -d [-f] [-k` *system-file*`] mddb` *nnn*

/sbin/metadb `[-s` *setname*`] -d [-f] [-k` *system-file*`]` *slice...*

/sbin/metadb `[-s` *setname*`] -i`

/sbin/metadb `[-s` *setname*`] -p [-k` *system-file*`] [`*mddb.cf-file*`]`

DESCRIPTION

The `metadb` command creates and deletes replicas of the metadevice state database. State database replicas can be created on dedicated slices, or on slices that will later become part of a simple metadevice (concatenation or stripe), RAID5 metadevice, or trans metadevice.

The metadevice state database contains the configuration of all metadevices and hot spare pools in the system. Additionally, the metadevice state database keeps track of the current state of metadevices and hot spare pools, and their components. Solaris Volume Manager automatically updates the metadevice state database when a configuration or state change occurs. A submirror failure is an example of a state change. Creating a new metadevice is an example of a configuration change.

The metadevice state database is actually a collection of multiple, replicated database copies. Each copy, referred to as a replica, is subject to strict consistency checking to ensure correctness.

Replicated databases have an inherent problem in determining which database has valid and correct data. To solve this problem, Volume Manager uses a *majority consensus algorithm*. This algorithm requires that a majority of the database replicas be available before any of them are declared valid. This algorithm strongly encourages the presence of at least three initial replicas, which you create. A consensus can then be reached as long as at least two of the three replicas are available. If there is only one replica and the system crashes, it is possible that all metadevice configuration data can be lost.

The majority consensus algorithm is conservative in the sense that it will fail if a majority consensus cannot be reached, even if one replica actually does contain the most up-to-date data. This approach guarantees that stale data will not be accidentally used, regardless of the failure scenario. The majority consensus algorithm accounts for the following: the system will stay running with exactly half or more replicas; the system will panic when less than half the replicas are available; the system will not reboot without one more than half the total replicas.

When used with no options, the `metadb` command gives a short form of the status of the metadevice state database. Use `metadb -i` for an explanation of the flags field in the output.

The initial state database is created using the `metadb` command with both the `-a` and `-f` options, followed by the slice where the replica is to reside. The `-a` option specifies that a replica (in this case, the initial) state database should be created. The `-f` option forces the creation to occur, even though a state database does not exist. (The `-a` and `-f` options should be used together only when no state databases exist.)

Additional replicas beyond those initially created can be added to the system. They contain the same information as the existing replicas, and help to prevent the loss of the configuration information. Loss of the configuration makes operation of the metadevices impossible. To create additional replicas, use the `metadb -a` command, followed by the name of the new slice(s) where the replicas will reside. All replicas that are located on the same slice must be created at the same time.

To delete all replicas that are located on the same slice, the `metadb -d` command is used, followed by the slice name.

When used with the `-i` option, `metadb` displays the status of the metadevice state databases. The status can change if a hardware failure occurs or when state databases have been added or deleted.

To fix a replica in an error state, delete the replica and add it back again.

OPTIONS

Root privileges are required for all of the following options except `-h` and `-i`.

The following options can be used with the `metadb` command. Not all the options are compatible on the same command line. Refer to the above synopsis line to see the supported use of the options.

`-a` — Attach a new database device. The `/kernel/drv/md.conf` file is automatically updated with the new information and the `/etc/lvm/mddb.cf` file is updated as well. An alternate way to create replicas is by defining them in the `/etc/lvm/md.tab` file and specifying the assigned name at the command line in the form, `mddb`*nn*, where *nn* is a two-digit number given to the replica definitions. Refer to the `md.tab`(4) man page for instructions on setting up replicas in that file.

`-c` *number* — Specifies the number of replicas to be placed on each device. The default number of replicas is 1.

`-d` — Deletes all replicas that are located on the specified *slice*. The `/kernel/drv/md.conf` file is automatically updated with the new information and the `/etc/lvm/mddb.cf` file is updated as well.

`-f` The `-f` option is used to create the initial state database. It is also used to force the deletion of replicas below the minimum of one. (The `-a` and `-f` options should be used together only when no state databases exist.)

`-h` Displays a usage message.

`-i` Inquire about the status of the replicas. The output of the `-i` option includes characters in front of the device name that represent the status of the state database. Explanations of the characters are displayed following the replica status and are as follows:

- `d` replica does not have an associated device ID.
- `o` replica active prior to last `mddb` configuration change
- `u` replica is up to date
- `l` locator for this replica was read successfully
- `c` replica's location was in `/etc/lvm/mddb.cf`
- `p` replica's location was patched in kernel
- `m` replica is master, this is replica selected as input
- `W` replica has device write errors
- `a` replica is active, commits are occurring to this
- `M` replica had problem with master blocks
- `D` replica had problem with data blocks
- `F` replica had format problems
- `S` replica is too small to hold current database
- `R` replica had device read errors

`-k` *system-file* Specifies the name of the kernel file where the replica information should be written. The default *system-file* is `/kernel/drv/md.conf`. This option is for use with the local diskset only.

`-l` *length* Specifies the size of each replica. The default *length* is 8192 blocks, which should be appropriate for most configurations.

`-p` Specifies updating the system file (`/kernel/drv/md.conf`) with entries from the `/etc/lvm/mddb.cf` file. This option is normally used to update a newly built system before it is booted for the first time. If the system has been built on a system other than the one where it will run, the location of the `mddb.cf` on the local

machine can be passed as an argument. The system file to be updated can be changed using the `-k` option. This option is for use with the local diskset only.

`-s` *setname* — Specifies the name of the diskset on which the `metadb` command will work. Using the `-s` option will cause the command to perform its administrative function within the specified diskset. Without this option, the command will perform its function on local database replicas.

slice — Specifies the logical name of the physical slice (partition), such as `/dev/dsk/c0t0d0s2`.

EXAMPLES

EXAMPLE 1 Creating Initial State Database Replicas

The following example creates the initial state database replicas on a new system.

```
# metadb -a -f c0t0d0s7 c0t1d0s2 c1t0d0s7 c1t1d0s2
```

The `-a` and `-f` options force the creation of the initial database and replicas. You could then create metadevices with these same slices, making efficient use of the system.

EXAMPLE 2 Adding Two Replicas on Two New Disks

This example shows how to add two replicas on two new disks that have been connected to a system currently running Volume Manager.

```
# metadb -a c0t2d0s2 c1t1d0s2
```

EXAMPLE 3 Deleting Two Replicas

This example shows how to delete two replicas from the system. Assume that replicas have been set up on `/dev/dsk/c0t2d0s2` and `/dev/dsk/c1t1d0s2`.

```
# metadb -d c0t2d0s2 c1t1d0s2
```

Although you can delete all replicas, you should never do so while metadevices still exist. Removing all replicas causes existing metadevices to become inoperable.

FILES

`/etc/lvm/mddb.cf`	Contains the location of each copy of the metadevice state database.
`/etc/lvm/md.tab`	Workspace file for metadevice database configuration.
`/kernel/drv/md.conf`	Contains database replica information for all metadevices on a system. Also contains Solaris Volume Manager configuration information.

EXIT STATUS

The following exit values are returned:

`0`	successful completion
`>0`	an error occurred

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdr

SEE ALSO

`metaclear`(1M), `metadetach`(1M), `metahs`(1M), `metainit`(1M), `metaoffline`(1M), `metaonline`(1M), `metaparam`(1M), `metareplace`(1M), `metaroot`(1M), `metaset`(1M), `metastat`(1M), `metasync`(1M), `metattach`(1M), `md.tab`(4), `md.cf`(4), `mddb.cf`(4)

Solaris Volume Manager Administration Guide

NAME

metadevadm – update metadevice information

SYNOPSIS

/usr/sbin/metadevadm [-h] [-n] [-r] [-u *disk_specifier*] [-v]

DESCRIPTION

The `metadevadm` command facilitates the administration of device ID entries in Solaris Volume Manager. Use this command when the pathname stored in the metadevice state database no longer correctly addresses the device or when a disk drive has had its device ID changed.

OPTIONS

Root privileges are required for all of the following options except `-h`.

`-h`	Provides a help display.
`-n`	Emulates the effect of a command, without making any changes to the system.
`-r`	Recomputes the pathname and disk specifier (including slice) associated with all devices in the metadevice state database if a device ID is present for the device. Use this option when the disk has been moved or readdressed.
`-u` *disk_specifier*	Obtain the device ID associated with the *disk_specifier* (for example, `c1t2d0`) of a device and update the metadevice state database. If the device ID has not changed this option does nothing. Use this option when a disk drive has had its device ID changed during a firmware upgrade or due to changing the controller of a storage subsystem.
`-v`	Verbose mode. Has no effect when used with `-u` (verbose is the default).

EXAMPLES

EXAMPLE 1 Updating Device ID of Disk

The following example updates the device `c2t3d0`:

```
# metadevadm -u c2t3d0
Updating SLVM device relocation information for c2t3d0.
Old device reloc information: id1928019239129312301201201001201209139 8
New device reloc information: id1938019239129312301201201001201209139 8
```

The following example is a variation of the preceding, using the full pathname.

```
# metadevadm -u /dev/dsk/c2t3d0
```

The following example uses the `-n` option, which means that the command is emulated, but does not take effect. Note that when the `-v` option is used with `-u`, `-v` has no effect (verbose is the default).

```
# metadevadm -u -v -n c2t3d0
Updating SLVM device relocation information for c2t3d0.
Old device reloc information: id1928019239129312301201201001201209139 8
New device reloc information: id1938019239129312301201201001201209139 8
```

EXAMPLE 1 Updating Device ID of Disk *(Continued)*

EXAMPLE 2 Recomputing Pathnames

In the following example, all device names are valid.

```
# metadevadm -r
Disk movement detected.
Updating device names in SLVM.
```

In the following example, once again device names are valid.

```
# metadevadm -r -v
Disk movement detected.
Updating device names in SLVM.
c0t0d0s0 changed to c0t0d1s0 from device relocation information
id120981231kmklsdjaasdkfjadfjakds
```

In the following example, `metadevadm` detects an invalid device name.

```
# metadevadm -r
Invalid device relocation information detected in SLVM.
Please check status of following disk(s):
c3t0d0
```

RETURN VALUES

The `metadevadm` command has the following return values:

`0` Command was successful.

`1` `metadevadm` encountered an error condition.

`2` An invalid device ID was detected when using the `-r` option. This is for use in the `rc2.d` script. See `init.d`(4).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdu

SEE ALSO

`metaclear`(1M), `metadb`(1M), `metadetach`(1M), `metahs`(1M), `metainit`(1M), `metaparam`(1M), `metarecover`(1M), `metareplace`(1M), `metaroot`(1M), `metaset`(1M), `metastat`(1M), `metasync`(1M), `metattach`(1M), `md.tab`(4), `md.cf`(4), `mddb.cf`(4)

Solaris Volume Manager Administration Guide

NAME

metahs – manage hot spares and hot spare pools

SYNOPSIS

/usr/sbin/metahs [-s *setname*] -a all *component*

/usr/sbin/metahs [-s *setname*] -a *hot_spare_pool* [*component*]

/usr/sbin/metahs [-s *setname*] -d *hot_spare_pool* [*component*]

/usr/sbin/metahs [-s *setname*] -d all *component*

/usr/sbin/metahs [-s *setname*] -e *component*

/usr/sbin/metahs [-s *setname*] -r *hot_spare_pool component-old*

/usr/sbin/metahs [-s *setname*] -r all *component-old component-new*

/usr/sbin/metahs [-s *setname*] -i [*hot_spare_pool*...]

DESCRIPTION

The `metahs` command manages existing hot spares and hot spare pools. It is used to add, delete, enable, and replace components (slices) in hot spare pools. Like the `metainit` command, the `metahs` command can also create an initial hot spare pool. The `metahs` command does not replace a component of a metadevice. This function is performed by the `metareplace` command.

Hot spares are always in one of three states: available, in-use, or broken. Available hot spares are running and ready to accept data, but are not currently being written to or read from. In-use hot spares are currently being written to and read from. Broken hot spares are out of service and should be repaired. The status of hot spares is displayed when `metahs` is invoked with the `-i` option.

OPTIONS

Root privileges are required for any of the following options except `-i`.

`-a all` *component*
Adds *component* to all hot spare pools. `all` is not case sensitive.

`-a` *hot_spare_pool* [*component*]
Adds the *component* to the specified *hot_spare_pool*. *hot_spare_pool* is created if it does not already exist.

`-d all` *component*
Deletes *component* from all the hot spare pools. The *component* cannot be deleted if it is in the in-use state.

`-d` *hot_spare_pool* [*component*]
Deletes *hot_spare_pool*, if the *hot_spare_pool* is both empty and not referenced by a metadevice. If *component* is specified, it is deleted from the *hot_spare_pool*. Hot spares in the in-use state cannot be deleted.

`-e` *component*
Enables *component* to be available for use as a hot spare. The *component* can be enabled if it is in the broken state and has been repaired.

`-i` [*hot_spare_pool* . . .]
Displays the status of the specified *hot_spare_pool* or for all hot spare pools if one is not specified.

`-r all` *component-old component-new*
Replace *component-old* with *component-new* in all hot spare pools which have the component associated. Components cannot be replaced from any hot spare pool if the old hot spare is in the in-use state.

`-r` *hot_spare_pool component-old component-new*
Replaces *component-old* with *component-new* in the specified *hot_spare_pool*. Components cannot be replaced from a hot spare pool if the old hot spare is in the in-use state.

`-s` *setname*
Specifies the name of the diskset on which `metahs` will work. Using the `-s` option will cause the command to perform its administrative function within the specified diskset. Without this option, the command will perform its function on local hot spare pools.

component
The logical name for the physical slice (partition) on a disk drive, such as `/dev/dsk/c0t0d0s2`.

hot_spare_pool
Hot spare pools must be of the form `hsp`*nnn*, where *nnn* is a number in the range 000-999.

EXAMPLES

EXAMPLE 1 Adding a hot spare to hot spare pool

The following example adds a hot spare `/dev/dsk/c0t0d0s7` to a hot spare pool `hsp003`.

```
# metahs -a hsp003 c0t0d0s7
```

When the hot spare is added to the pool, the existing order of the hot spares already in the pool is preserved. The new hot spare is added at the end of the list of hot spares in the hot spare pool specified.

EXAMPLE 2 Adding a hot spare to all pools currently defined

This example adds a hot spare to the hot spare pools that are currently defined.

```
# metahs -a all c0t0d0s7
```

The keyword `all` in this example specifies adding the hot spare, `/dev/dsk/c0t0d0s7`, to all the hot spare pools.

EXAMPLE 3 Deleting a hot spare

This example deletes a hot spare, `/dev/dsk/c0t0d0s7`, from a hot spare pool, `hsp003`.

```
# metahs -d hsp003 c0t0d0s7
```

When you delete a hot spare, the position of the remaining hot spares in the pool changes to reflect the new order. For instance, if in this example `/dev/dsk/c0t0d0s7` were the second of three hot spares, after deletion the third hot spare would move to the second position.

EXAMPLE 4 Replacing a hot spare

This example replaces a hot spare that was previously defined.

```
# metahs -r hsp001 c0t1d0s0 c0t3d0s0
```

In this example, the hot spare `/dev/dsk/c0t1d0s0` is replaced by `/dev/dsk/c0t3d0s0`. The order of the hot spares does not change.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdu

SEE ALSO

`metaclear`(1M), `metadb`(1M), `metadetach`(1M), `metainit`(1M), `metaoffline`(1M), `metaonline`(1M), `metaparam`(1M), `metareplace`(1M), `metaroot`(1M), `metaset`(1M), `metastat`(1M), `metasync`(1M), `metattach`(1M), `md.tab`(4), `md.cf`(4), `mddb.cf`(4), `attributes`(5)

Solaris Volume Manager Administration Guide

NAME

metainit – configure metadevices

SYNOPSIS

/sbin/metainit `-h`

/sbin/metainit [*generic options*] *concat/stripe numstripes width component...* [`-i` *interlace*]

/sbin/metainit [*width component...* [`-i` *interlace*]] [`-h` *hot_spare_pool*]

/sbin/metainit [*generic options*] *mirror* `-m` *submirror* [*read_options*] [*write_options*] [*pass_num*]

/sbin/metainit [*generic options*] *RAID* `-r` *component...* [`-i` *interlace*] [`-h` *hot_spare_pool*] [`-k`] [`-o` *original_column_count*]

/sbin/metainit [*generic options*] *trans* `-t` *master* [*log*]

/sbin/metainit [*generic options*] *hot_spare_pool* [*hotspare...*]

/sbin/metainit [*generic options*] *metadevice-name*

/sbin/metainit [*generic options*] `-a`

/sbin/metainit [*generic options*] *softpart* `-p` [`-e`] *component size*

/sbin/metainit `-r`

DESCRIPTION

The `metainit` command configures metadevices and hot spares according to the information specified on the command line. Alternatively, you can run `metainit` so that it uses configuration entries you specify in the `/etc/lvm/md.tab` file (see `md.tab`(4)). All metadevices must be set up by the `metainit` command before they can be used.

If you edit the `/etc/lvm/md.tab` file to configure metadevices, specify one complete configuration entry per line. You then run the `metainit` command with either the `-a` option, to activate all metadevices you entered in the `/etc/lvm/md.tab` file, or with the metadevice name corresponding to a specific configuration entry.

Note – Volume Manager never updates the `/etc/lvm/md.tab` file. Complete configuration information is stored in the metadevice state database, not `md.tab`. The only way information appears in `md.tab` is through editing it by hand.

When setting up a disk mirror, the *first* step is to use `metainit` create a one-on-one concatenation for the root slice. See EXAMPLES.

GENERIC OPTIONS

Root privileges are required for all of the following options except `-h`.

`-f` Forces the `metainit` command to continue even if one of the slices contains a mounted file system or is being used as `swap`. This option is useful when configuring mirrors on root (`/`), `swap`, and `/usr`.

`-h` Displays usage message.

`-n` Checks the syntax of your command line or `md.tab` entry without actually setting up the metadevice. If used with `-a`, all devices are checked but not initialized.

`-r` Only used in a shell script at boot time. Sets up all metadevices that were configured before the system crashed or was shut down. The information about previously configured metadevices is stored in the metadevice state database (see `metadb`(1M)).

`-s` *setname* Specifies the name of the diskset on which `metainit` will work. Without the `-s` option, the `metainit` command operates on your local metadevices and/or hotspares.

CONCAT/STRIPE OPTIONS

concat/stripe Specifies the metadevice name of the concatenation, stripe, or concatenation of stripes being defined.

numstripes Specifies the number of individual stripes in the metadevice. For a simple stripe, *numstripes* is always 1. For a concatenation, *numstripes* is equal to the number of slices. For a concatenation of stripes, *numstripes* will vary according to the number of stripes.

width Specifies the number of slices that make up a stripe. When *width* is greater than 1, the slices are striped.

component The logical name for the physical slice (partition) on a disk drive, such as `/dev/dsk/c0t0d0s2`. For RAID level 5 metadevices, a minimum of three slices is necessary to enable striping of the parity information across slices.

`-i` *interlace* Specifies the interlace size. This value tells Volume Manager how much data to place on a slice of a striped or RAID level 5 metadevice before moving on to the next slice. *interlace* is a specified value, followed by either '`k`' for kilobytes, '`m`' for megabytes, or '`b`' for blocks. The characters can be either uppercase or lowercase. The *interlace* specified cannot be less than 16 blocks, or greater than 100 megabytes. If *interlace* is not specified, it defaults to 16 kilobytes.

`-h` *hot_spare_pool* Specifies the *hot_spare_pool* to be associated with the metadevice. If you use the command line, the hot spare pool must have been previously created by the `metainit` command before it can be associated with a metadevice. The *hot_spare_pool* must be of the form `hsp`*nnn*, where *nnn* is a number in the range 000-999. Use `/-h hsp`*nnn* when the concat/stripe being created is to be used as a submirror.

MIRROR OPTIONS

mirror `-m` *submirror*

Specifies the metadevice name of the mirror. The `-m` indicates that the configuration is a mirror. *submirror* is a metadevice (stripe or concatentation) that makes up the initial one-way mirror. Volume Manager supports a maximum of four-way mirroring. When defining mirrors, first create the mirror with the `metainit` command as a one-way mirror. Then attach subsequent submirrors using the `metattach` command. This method ensures that Volume Manager properly syncs the mirrors. (The second and any subsequent submirrors are first created using the `metainit` command.)

read_options

The following read options for mirrors are available:

`-g` Enables the geometric read option, which results in faster performance on sequential reads.

`-r` Directs all reads to the first submirror. This should only be used when the devices comprising the first submirror are substantially faster than those of the second mirror. This flag cannot be used with the `-g` flag.

If neither the `-g` nor `-r` flags are specified, reads are made in a round-robin order from all submirrors in the mirror. This enables load balancing across the submirrors.

write_options

The following write options for mirrors are available:

`-S` Performs serial writes to mirrors. The first submirror write completes before the second is started. This may be useful if hardware is susceptible to partial sector failures. If `-S` is not specified, writes are replicated and dispatched to all mirrors simultaneously.

pass_num

A number in the range 0-9 at the end of an entry defining a mirror that determines the order in which that mirror is resynced during a reboot. The default is 1. Smaller pass numbers are resynced first. Equal pass numbers are run concurrently. If 0 is used, the resync is skipped. 0 should be used only for mirrors mounted as read-only, or as `swap`.

RAID Level 5 OPTIONS

RAID `-r`
: Specifies the name of the RAID level 5 metadevice. The `-r` specifies that the configuration is RAID level 5.

`-k`
: For RAID level 5 metadevices, informs the driver that it is not to initialize (zero the disk blocks) due to existing data. Only use this option to recreate a previously created RAID level 5 device.

`-o` *original_column_count*
: For RAID level 5 metadevices, used with the `-k` option to define the number of original slices in the event the originally defined metadevice was grown. This is necessary since the parity segments are not striped across concatenated devices.

Warning for `-k` and `-o`: Use extreme caution when using the `-k` and `-o` options. When used, these options set the disk blocks to the `OK` state. If any errors exist on disk blocks within the metadevice, Volume Manager might begin fabricating data. Instead of using these options, you might want to initialize the device and restore data from tape.

TRANS OPTIONS

trans `-t` *master* `[` *log* `]`
: *trans* specifies the name of the trans metadevice, which consists of *master* and *log* devices, or just a *master* device. The `-t` specifies that the configuration is a trans metadevice. If *log* is not specified when you create the trans metadevice, no logging can take place until a logging device is provided by using the `metattach` command. *master* and *log* can be simple, mirror, or RAID level 5 metadevices. They cannot be trans metadevices. *master* should be a UFS file system. You can configure an existing file system for logging by creating a trans metadevice as follows: make the existing file system into the master trans device, then create the log device on a separate, unused slice. The minimum log size is 1 Mbyte of disk space. Under heavy sustained loads, small logs will detract from performance because old data must be copied from the log to the file system before new data can be logged. The maximum log size is 1 Gbyte. Large logs might increase performance. However, logs larger than 64 Mbytes can have negligible performance benefits.

SOFT PARTITION OPTIONS

softpart `-p` [`-e`] *component size*
: The *softpart* argument specifies the name of the soft partition. The `-p` specifies that the configuration is a soft partition.

  The `-e` specifies that the entire disk specified by *component* as `c*t*d*` should be repartitioned and reserved for soft partitions. The specified component will be repartitioned such that slice 7 reserves space for system (state database replica)

usage and slice 0 contains all remaining space on the disk. Slice 7 will be a minimum of 4MB, but can be larger, depending on the disk geometry. The newly created soft partition will be placed on slice 0 of the device.

The *component* argument specifies the disk (`c*t*d*`), slice (`c*t*d*s*`), or meta device (`d*`) from which to create the soft partition. The *size* argument determines the space to use for the soft partition and can be specified in `K` or `k` for kilobytes, `M` or `m` for megabytes, `G` or `g` for gigabytes, `T` or `t` for terabyte (one terabyte is the maximum size), and `B` or `b` for blocks (sectors).

HOT SPARE POOL OPTIONS

hot_spare_pool `[` *hotspare...* `]`

When used as arguments to the `metainit` command, *hot_spare_pool* defines the name for a hot spare pool, and *hotspare...* is the logical name for the physical slice(s) for availability in that pool. *hot_spare_pool* is a number of the form `hsp`*nnn*, where *nnn* is a number in the range 000-999.

`md.tab` FILE OPTIONS

metadevice-name

When the `metainit` command is run with a *metadevice-name* as its only argument, it searches the `/etc/lvm/md.tab` file to find that name and its corresponding entry. The order in which entries appear in the `md.tab` file is unimportant. For example, consider the following `md.tab` entry:

```
d0 2 1 c1t0d0s0 1 c2t1d0s0
```

When you run the command `metainit d0`, it configures metadevice `d0` based on the configuration information found in the `md.tab` file.

`-a`

Activates all metadevices defined in the `md.tab` file.

EXAMPLES

EXAMPLE 1 Creating a One-on-One Concatenation

The following command creates a one-on-one concatenation for the root slice. Such a command is the first step you take when setting up a mirror for the root slice (and any other slice that cannot be unmounted).

```
# metainit d1 1 1 c0t0d0s0
```

The preceding command makes `d1` a one-on-one concatenation, using the root slice. You can then enter:

```
# metainit d0 -m d1
```

...to make a one-way mirror of the root slice.

EXAMPLE 2 Concatenation

All drives in the following examples have the same size of 525 Mbytes.

EXAMPLE 2 Concatenation *(Continued)*

This example shows a metadevice, `/dev/md/dsk/d7`, consisting of a concatenation of four slices.

```
# metainit d7 4 1 c0t1d0s0 1 c0t2d0s0 1 c0t3d0s0 1 /dev/dsk/c0t4d0s0
```

The number 4 indicates there are four individual stripes in the concatenation. Each stripe is made of one slice, hence the number 1 appears in front of each slice. Note: The first disk sector in all of the above devices contains a disk label. To preserve the labels on devices `/dev/dsk/c0t2d0s0`, `/dev/dsk/c0t3d0s0`, and `/dev/dsk/c0t4d0s0`, the metadisk driver must skip at least the first sector of those disks when mapping accesses across the concatenation boundaries. Because skipping only the first sector would create an irregular disk geometry, the entire first cylinder of these disks will be skipped. This allows higher level file system software to optimize block allocations correctly.

EXAMPLE 3 Stripe

This example shows a metadevice, `/dev/md/dsk/d15`, consisting of two slices.

```
# metainit d15 1 2 c0t1d0s2 c0t2d0s2 -i 32k
```

The number 1 indicates that one stripe is being created. Because the stripe is made of two slices, the number 2 follows next. The optional `-i` followed by `32k` specifies the interlace size will be 32 Kbytes. If the interlace size were not specified, the stripe would use the default value of 16 Kbytes.

EXAMPLE 4 Concatentation of Stripes

This example shows a metadevice, `/dev/md/dsk/d75`, consisting of a concatenation of two stripes of three disks.

```
# metainit d75 2 3 c0t1d0s2 c0t2d0s2 \
        c0t3d0s2 -i 16k \
      3 c1t1d0s2 c1t2d0s2 c1t3d0s2 -i 32k
```

On the first line, the `-i` followed by 16k specifies that the stripe interlace size is 16 Kbytes. The second set specifies the stripe interlace size will be 32 Kbytes. If the second set did not specify 32 Kbytes, the set would use the default interlace value of 16 Kbytes. The blocks of each set of three disks are interlaced across three disks.

EXAMPLE 5 Mirroring

This example shows a two-way mirror, `/dev/md/dsk/d50`, consisting of two submirrors. This mirror does not contain any existing data.

```
# metainit d51 1 1 c0t1d0s2
# metainit d52 1 1 c0t2d0s2
```

EXAMPLE 5 Mirroring *(Continued)*

```
# metainit d50 -m d51
# metattach d50 d52
```

In this example, two submirrors, `d51` and `d52`, are created with the `metainit` command. These two submirrors are simple concatenations. Next, a one-way mirror, `d50`, is created using the `-m` option wtih `d51`. The second submirror is attached later using the `metattach` command. When creating a mirror, any combination of stripes and concatenations can be used. The default read and write options in this example are a round-robin read algorithm and parallel writes to all submirrors.

EXAMPLE 6 Logging (trans)

This example shows trans metadevice, `/dev/md/dsk/d1`, with mirrors for the master and logging devices. This trans does not contain any existing data.

```
# metainit d11 1 1 c0t1d0s2
# metainit d12 1 1 c0t2d0s2
# metainit d21 1 1 c1t1d0s3
# metainit d22 1 1 c1t2d0s3
# metainit d10 -m d11
# metattach d10 d12
# metainit d20 -m d21
# metattach d20 d22
# metainit d1 -t d10 d20
```

This example begins by defining four concatenations, `d11`, `d12`, `d21`, and `d22`. Next, mirror `d10` is defined, followed by mirror `d20`. The mirrors are initially defined as one-way mirrors, then the second submirrors are attached later with the `metattach` command. Finally, the trans metadevice `d1` is defined, with `d10` as the master device and `d20` as the logging device by using the `-t` option.

EXAMPLE 7 RAID Level 5

This example shows a RAID level 5 device, `d80`, consisting of three slices:

```
# metainit d80 -r c1t0d0s2 c1t1d0s2 c1t3d0s2 -i 20k
```

In this example, a RAID level 5 metadevice is defined using the `-r` option with an interlace size of 20 Kbytes. The data and parity segments will be striped across the slices, `c1t0d0s2`, `c1t2d0s2`, and `c1t3d0s2`.

EXAMPLE 8 Soft Partition

The following example shows a soft partition device, `d1`, built on metadevice `d100` and 100 Mbytes (indicated by `100M`) in size:

```
# metainit d1 -p d100 100M
```

EXAMPLE 8 Soft Partition *(Continued)*

The preceding command creates a 100–Mbyte soft partition on the `d100` metadevice. This metadevice could be a RAID level 5, stripe, concatenation, or mirror.

EXAMPLE 9 Soft Partition on Full Disk

The following example shows a soft partition device, `d1`, built on disk `c3t4d0`:

```
# metainit d1 -p -e c3t4d0 9Gb
```

In this example, the disk is repartitioned and a soft partition is defined to occupy all 9 Gbytes of disk `c3t4d0s0`.

EXAMPLE 10 Hot Spare

This example shows a two-way mirror, `/dev/md/dsk/d10`, and a hot spare pool with three hot spare components. The mirror does not contain any existing data.

```
# metainit hsp001 c2t2d0s2 c3t2d0s2 c1t2d0s2
# metainit d41 1 1 c1t0d0s2 -h hsp001
# metainit d42 1 1 c3t0d0s2 -h hsp001
# metainit d40 -m d41
# metattach d40 d42
```

In the above example, a hot spare pool, `hsp001`, is created with three disks used as hot spares. Next, two submirrors are created, `d41` and `d42`. These are simple concatenations. The `metainit` command uses the `-h` option to associate the hot spare pool `hsp001` with each submirror. A one-way mirror is then defined using the `-m` option. The second submirror is attached using the `metattach` command.

FILES

`/etc/lvm/md.tab`
Contains list of metadevice and hot spare configurations for batch-like creation.

Multi-Way Mirror

Do not use the `metainit` command to create a multi-way mirror. Rather, create a one-way mirror with `metainit` then attach additional submirrors with `metattach`. When the `metattach` command is not used, no resync operations occur and data could become corrupted.

If you use `metainit` to create a mirror with multiple submirrors, the following message is displayed:

```
WARNING: This form of metainit is not recommended.
The submirrors may not have the same data.
Please see ERRORS in metainit(1M) for additional information.
```

Write-On-Write Problem

When mirroring data in Solaris Volume Manager, transfers from memory to the disks do not all occur at exactly the same time for all sides of the mirror. If the contents of buffers are changed while the data is in-flight to the disk (called write-on-write), then different data can end up being stored on each side of a mirror.

This problem can be addressed by making a private copy of the data for mirror writes, however, doing this copy is expensive. Another approach is to detect when memory has been modified across a write by looking at the dirty-bit associated with the memory page. Volume Manager uses this dirty-bit technique when it can. Unfortunately, this technique does not work for raw I/O or direct I/O. By default, Volume Manager is tuned for performance with the liability that mirrored data might be out of sync if an application does a "write-on-write" to buffers associated with raw I/O or direct I/O.

Note that without mirroring, you were not guaranteed what data would actually end up on media, but multiple reads would return the same data. With mirroring, multiple reads may return different data. The following line can be added to `/etc/system` to cause a stable copy of the buffers to be used for all raw I/O and direct I/O write operations.

```
set md_mirror:md_mirror_wow_flg=0x20
```

Setting this flag will degrade performance.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdr

SEE ALSO

`metaclear`(1M), `metadb`(1M), `metadetach`(1M), `metahs`(1M), `metaoffline`(1M), `metaonline`(1M), `metaparam`(1M), `metarecover`(1M), `metareplace`(1M), `metaroot`(1M), `metaset`(1M), `metastat`(1M), `metasync`(1M), `metattach`(1M), `md.cf`(4), `md.tab`(4), `mddb.cf`(4), `attributes`(5)

Solaris Volume Manager Administration Guide

LIMITATIONS

Recursive mirroring is not allowed; that is, a mirror cannot appear in the definition of another mirror.

Recursive logging is not allowed; that is, a trans metadevice cannot appear in the definition of another metadevice.

Stripes, concatenations, and RAID level 5 metadevices must consist of slices only.

Mirroring of RAID level 5 metadevices is not allowed.

Soft partitions can be built on raw devices, or on stripes, RAID level 5, or mirrors.

RAID level 5 or stripe metadevices can be built directly on soft partitions.

NAME

metaoffline, metaonline – place submirrors offline and online

SYNOPSIS

/usr/sbin/metaoffline -h

/usr/sbin/metaoffline [-s *setname*] [-f] *mirror submirror*

/usr/sbin/metaonline -h

/usr/sbin/metaonline [-s *setname*] *mirror submirror*

DESCRIPTION

The `metaoffline` command prevents Solaris Volume Manager from reading and writing to the submirror that has been taken offline. While the submirror is offline, all writes to the mirror will be kept track of (by region) and will be written when the submirror is brought back online. The `metaoffline` command can also be used to perform online backups: one submirror is taken offline and backed up while the mirror remains accessible. (However, if this is a two-way mirror, data redundancy is lost while one submirror is offline.) The `metaoffline` command differs from the `metadetach` command because it does not sever the logical association between the submirror and the mirror. To completely remove a submirror from a mirror, use the `metadetach` command.

A submirror that has been taken offline will only remain offline until the `metaonline` command is invoked or the system is rebooted.

When the `metaonline` command is used, reading from and writing to the submirror resumes. A resync is automatically invoked to resync the regions written while the submirror was offline. Writes are directed to the submirror during resync. Reads, however, will come from a different submirror. Once the resync operation completes, reads and writes are performed on that submirror. The `metaonline` command is only effective on a submirror of a mirror that has been taken offline.

Note: A submirror that has been taken offline with the `metaoffline` command can only be mounted as read-only.

OPTIONS

Root privileges are required for all of the following options except `-h`.

`-f`	Forces offlining of submirrors that have slices requiring maintenance.
`-h`	Displays usage message.
`-s` *setname*	Specifies the name of the diskset on which `metaoffline` and `metaonline` will work. Using the `-s` option will cause the command to perform its administrative function within the specified diskset. Without this option, the command will perform its function on local metadevices.
mirror	Specifies the metadevice name of the mirror from which the submirror will be either taken offline or put online.
submirror	Specifies the metadevice name of the submirror to be either taken offline or put online.

EXAMPLES

EXAMPLE 1 Taking submirror offline

This example takes one submirror, `d9`, offline from mirror `d10`.

```
# metaoffline d10 d9
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdu

SEE ALSO

`metaclear`(1M), `metadb`(1M), `metadetach`(1M), `metahs`(1M), `metainit`(1M), `metaparam`(1M), `metareplace`(1M), `metaroot`(1M), `metaset`(1M), `metastat`(1M), `metasync`(1M), `metattach`(1M), `md.tab`(4), `md.cf`(4), `mddb.cf`(4), `attributes`(5)

Solaris Volume Manager Administration Guide

NAME

metaparam – modify parameters of metadevices

SYNOPSIS

/usr/sbin/metaparam -h

/usr/sbin/metaparam [-s *setname*] [*concat/stripe or RAID5 options*] *concat/stripe RAID*

/usr/sbin/metaparam [-s *setname*] [*mirror options*] *mirror*

DESCRIPTION

The `metaparam` command is used to display or modify current parameters of metadevices.

If just the metadevice is specified as an argument to the `metaparam` command, the current settings are displayed.

The `metaparam` command enables most metadevice (volume) parameters to be changed. Only the interlace value cannot be changed by `metaparam`, because it is established when the metadevice is created and cannot be changed thereafter.

OPTIONS

Root privileges are required for all of the following options except `-h`.

`-h` Displays usage message.

`-s` *setname* Specifies the name of the diskset on which `metaparam` will work. Using the `-s` option will cause the command to perform its administrative function within the specified diskset. Without this option, the command will perform its function on local metadevices.

CONCAT/STRIPE OR RAID5 OPTIONS

`–h` *hot_spare_pool* | `none` Specifies the hot spare pool to be used by a metadevice. If `none` is specified, the metadevice is disassociated with the hot spare pool assigned to it. If the metadevice is currently using a hot spare, then `metaparam` cannot replace the hot spare pool.

concat/stripe | *RAID* Specifies the metadevice name of the concatenation, stripe, or concatenation of stripes, or of the RAID5 metadevice.

MIRROR OPTIONS

`–r roundrobin | geometric | first`
Modifies the read option for a mirror. The `-r` option must be followed by either `roundrobin`, `geometric`, or `first`. `roundrobin`, which is the default action under the `metainit` command, specifies reading the disks in a round-robin (load balancing) method. `geometric` allows for faster performance on sequential reads. `first` specifies reading only from the first submirror.

`–w parallel | serial`
Modifies the write option for a mirror. The `-w` option must be followed by either `parallel` or `serial`. `parallel`, the default action under the `metainit` command, specifies that all writes are parallel. `serial` specifies that all writes are serial.

`-p` *pass_number*
A number from 0-to-9 that specifies the order in which a mirror is resynced during reboot. The default is 1. Smaller pass numbers are resynced first. Equal pass numbers are run concurrently. If 0 is used, the mirror resync is skipped. 0 should only be used for mirrors mounted as read-only, or as `swap`.

mirror
Specifies the metadevice name of the mirror.

EXAMPLES

EXAMPLE 1 Associating Hot Spare Pool with RAID5 Metadevice

This example associates a hot spare pool, `hsp005`, with a RAID5 metadevice, `d80`.

```
# metaparam -h hsp005 d80
```

EXAMPLE 2 Changing Read Option to Geometric

This example changes the read option on a mirror `d50` from the default of roundrobin to geometric.

```
# metaparam -r geometric d50
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdu

SEE ALSO

`metaclear`(1M), `metadb`(1M), `metadetach`(1M), `metahs`(1M), `metainit`(1M), `metaoffline`(1M), `metaonline`(1M), `metareplace`(1M), `metaroot`(1M), `metaset`(1M), `metastat`(1M), `metasync`(1M), `metattach`(1M), `md.tab`(4), `md.cf`(4), `mddb.cf`(4)

Solaris Volume Manager Administration Guide

NAME

metarecover – recover soft partition information

SYNOPSIS

/sbin/metarecover [-n] [-v] [-s *setname*] *component* -p

/sbin/metarecover [-n] [-v] [-s *setname*] *component* -p {-d}

/sbin/metarecover [-n] [-v] [-s *setname*] *component* -p {-m}

DESCRIPTION

The `metarecover` command scans a specified component to look for soft partition configuration information and to regenerate the configuration.

OPTIONS

The `metarecover` command line options are as follows:

`-d`	Recover soft partitions in the metadevice state database from the extent headers on the device. Options `-d` and `-m` are mutually exclusive.
`-m`	Regenerates the extent headers and reapplies them to the underlying device based on the soft partitions listed in the metadevice state database. Options `-d` and `-m` are mutually exclusive.
`-n`	Does not actually perform the operation, but shows the output or errors that would have resulted from the operation, had it been run.
`-p`	Regenerates soft partitions based on the metadevice state database or extent headers on the underlying device. If neither `-d` nor `-m` are specified, this option compares the soft partition information in the metadevice state database to the extent headers.
`-s` *setname*	Specifies the name of the diskset on which `metarecover` will work. Using the `s` option causes the command to perform its function within the specified diskset. Without the `-s` option, the `metarecover` command operates on the metadevices and/or hot spare pools in the local diskset.
`-v`	Verbose mode, displaying the changes being made.

OPERANDS

The following operand is supported:

component	Specifies the `c*t*d*s*` number of the disk or slice containing the partitions, or the device name (for example, `d10`) of the metadevice containing the partitions.

EXAMPLES

EXAMPLE 1 Update Metadevice State Database Based on Disk Extent Headers

A disk containing soft partitions is moved from one system to another. The system administrator would like to use the existing soft partitions. `metarecover` will update the metadevice state database based on the extent headers on the disk.

```
# metarecover -v c0t3d0s2 -p -d
```

EXAMPLE 2 Update Metadevice State Database Based on Incomplete Soft Partition Creation

A system crashes in the middle of creating a new soft partition. The soft partition is in the creating state and the driver will not let that device be opened. `metarecover` will rewrite the extent headers for the partially created soft partition and mark it as `Okay`.

```
# metarecover -v c0t3d0s2 -p -m
```

EXAMPLE 3 Update Extent Headers Based on Metadevice State Database

Someone accidentally overwrote a portion of a disk leaving extent headers destroyed. `metarecover` will rewrite the extent headers to ensure a valid soft partition configuration, though user data will not be recovered.

```
# metarecover -v d5 -m
```

EXAMPLE 4 Validate Soft Partition Configuration

To validate the existing soft partition configuration, use `metarecover` with only the `-p` flag.

```
# metarecover c0t3d0s2 -p
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdr

SEE ALSO

`metaclear`(1M), `metadb`(1M), `metadetach`(1M), `metahs`(1M), `metaoffline`(1M), `metaonline`(1M), `metaparam`(1M), `metareplace`(1M), `metaroot`(1M), `metaset`(1M), `metastat`(1M), `metasync`(1M), `metattach`(1M), `md.cf`(4), `md.tab`(4), `mddb.cf`(4), `attributes`(5)

Solaris Volume Manager Administration Guide

NAME

metarename – rename metadevice or switch layered metadevice names

SYNOPSIS

/usr/sbin/metarename [-s *setname*] *metadevice1 metadevice2*

/usr/sbin/metarename [-s *setname*] [-f] -x *metadevice1 metadevice2*

/usr/sbin/metarename -h

DESCRIPTION

There are two ways to use `metarename`, one with and one without the `-x` option. The first method (without `-x`) renames an existing metadevice to a new name. This makes managing the metadevice namespace easier. The metadevice being renamed cannot be mounted or open, nor can the new name already exist. For example, to rename a metadevice that contains a mounted file system, you would first need to unmount the file system.

With the second way to use `metarename`, using the `-x` option, `metarename` switches (exchanges) the names of an existing layered metadevice and one of its subdevices. (In Solaris Volume Manager terms, a layered metadevice can be either a mirror or a trans metadevice.) The `-x` option enables you to switch the metadevice names of a mirror and one of its submirrors, or a trans metadevice and its master device.

`metarename -x` makes it easier to mirror or unmirror an existing stripe or concatenation, and to create or remove a trans of an existing metadevice.

When used to mirror an existing stripe or concatenatation, you must stop access to the device. For example, if the device contains a mounted file system, you must first unmount the file system before doing the rename.

The `metarename -x` command can also be used to create a trans metadevice from an existing metadevice, or to untrans the device. This applies only to the master device. A logging device cannot be created or removed with `metarename`. Before you can rename a trans device, you must detach the logging device. Then you must stop access to the trans metadevice itself.

You cannot rename or switch metadevices that are in an error state or that have subcomponents in an error state, or metadevices actively using a hot spare replacement.

You can only switch metadevices that have a direct child/parent relationship. You could not, for example, directly exchange a stripe in a mirror that is a master device with the trans metadevice.

You must use the `-f` flag when switching members of a trans metadevice.

Only metadevices can be switched, not slices.

OPTIONS

`-f`
Force the switching of trans metadevice members.

`-h`
Display a help message.

`-s` *setname*
Specifies the name of the diskset on which `metarename` will work. Using the `-s` option will cause the command to perform its administrative function within the specified diskset. Without this option, the command will perform its function on the local metadevices.

`-x`
Exchange the metadevice names *metadevice1* and *metadevice2*.

metadevice1
Specifies the metadevice to be renamed or switched.

metadevice2
Specifies the target metadevice name for the rename or switch operation.

EXAMPLES

EXAMPLE 1 Renaming a Metadevice

This example renames a metadevice named `d10` to `d100`. Note that `d100` must not exist for the rename to succeed.

```
# metarename d10 d100
```

EXAMPLE 2 Creating a Two-Way Mirror

This example creates a two-way mirror from an existing stripe named `d1` with a mounted file system, `/home2`.

```
# metainit d2 1 1 c13d0s1
# metainit -f d20 -m d1
# umount /home2
# metarename -x d20 d1
# metattach d1 d2
# mount /home2
```

First, a second concatenation `d2`, is created. (`d1` already exists.) The `metainit` command creates a one-way mirror, `d20`, from `d1`. Next, you umount the file system and switch `d1` for `d20`, making `d1` the top-level device (mirror). You attach the second submirror, `d2`, to create a two-way mirror. Lastly, you remount the file system.

EXAMPLE 3 Mounting Mirrored File System on Stripe

This example takes an existing mirror named `d1` with a mounted file system, and ends up with the file system mounted on a stripe `d1`.

```
# umount /fs2
# metarename -x d1 d20
# metadetach d20 d1
# metaclear -r d20
# mount /fs2
```

First, you unmount the file system, then switch the mirror `d1` and its submirror `d20`. This makes the mirror into `d20`. Next, you detach `d1` from `d20`, then delete the mirror `d20` and its other submirror. You then remount the file system.

EXAMPLE 4 Creating Trans Metadevice from Existing RAID Level 5 Metadevice

This example creates a trans metadevice from an existing RAID level 5 metadevice named `d1` which contains the file system `/myhome`.

```
# umount /myhome
# metainit d21 -t d1
# metarename -f -x d21 d1
# metattach d1 d0
# mount /myhome
```

You umount the file system before using the `metainit` command to create the trans metadevice `d21`, with `d1` as the master device. You then switch `d21` and `d1`, making `d1` the top-level metadevice (trans metadevice). A logging device `d0` is attached with the `metattach` command. You then remount the file system.

EXAMPLE 5 Deleting Trans Metadevice

This example deletes a trans metadevice named `d10` while its mount point is `/myhome`. The master device, which is a stripe, is named `d2`. The logging device, also a stripe, is named `d5`.

```
# umount /myhome
# metadetach d10
# metarename -f -x d10 d2
# metaclear d2
# metaclear d5
# fsck /dev/md/dsk/d10
# mount /myhome
```

You umount the file system first, then detach the trans metadevice's logging device. The trans metadevice is switched with the master device, making the trans metadevice `d2` and the underlying stripe `d10`. You clear the trans metadevice `d2` and the logging device `d5`. `d10` must be `fsck`'d, and then the file system is remounted.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdu

SEE ALSO

`metaclear`(1M), `metainit`(1M), `metastat`(1M), `attributes`(5)

Solaris Volume Manager Administration Guide

LIMITATIONS

Renaming and exchanging metadevice names can only be used for metadevices. A physical slice cannot be renamed to a metadevice, nor can a metadevice be exchanged with a physical slice name.

Metadevice names are strings of the pattern *d<xyz>* where *xyz* is a value between `0` and `8192`. You cannot use logical names for metadevices.

NAME

metareplace – enable or replace components of submirrors or RAID5 metadevices

SYNOPSIS

/usr/sbin/metareplace `-h`

/usr/sbin/metareplace `[-s` *setname*`]` `-e` *mirror component*

/usr/sbin/metareplace `[-s` *setname*`]` *mirror component-old component-new*

/usr/sbin/metareplace `[-s` *setname*`]` `-e` *RAID component*

/usr/sbin/metareplace `[-s` *setname*`]` `[-f]` *RAID component-old component-new*

DESCRIPTION

The `metareplace` command is used to enable or replace components (slices) within a submirror or a RAID5 metadevice.

When you replace a component, the `metareplace` command automatically starts resyncing the new component with the rest of the metadevice. When the resync completes, the replaced component becomes readable and writable. If the failed component has been hot spare replaced, the hot spare is placed in the available state and made available for other hot spare replacements.

Note that the new component must be large enough to replace the old component.

A component may be in one of several states. The `Last Erred` and the `Maintenance` states require action. Always replace components in the `Maintenance` state first, followed by a resync and validation of data. After components requiring maintenance are fixed, validated, and resynced, components in the `Last Erred` state should be replaced. To avoid data loss, it is always best to back up all data before replacing `Last Erred` devices.

OPTIONS

Root privileges are required for all of the following options except `-h`.

`-e` Transitions the state of *component* to the available state and resyncs the failed component. If the failed component has been hot spare replaced, the hot spare is placed in the available state and made available for other hot spare replacements. This command is useful when a component fails due to human error (for example, accidentally turning off a disk), or because the component was physically replaced. In this case, the replacement component must be partitioned to match the disk being replaced before running the `metareplace` command.

`-f` Forces the replacement of an errored component of a metadevice in which multiple components are in error. The component determined by the `metastat` display to be in the ''Maintenance'' state must be replaced first. This option may cause data to be fabricated since multiple components are in error.

`-h` Display help message.

`-s` *setname* Specifies the name of the diskset on which `metareplace` will work. Using the `-s` option will cause the command to perform its

	administrative function within the specified diskset. Without this option, the command will perform its function on local metadevices.
mirror	The metadevice name of the mirror.
component	The logical name for the physical slice (partition) on a disk drive, such as `/dev/dsk/c0t0d0s2`.
component-old	The physical slice that is being replaced.
component-new	The physical slice that is replacing *component-old*.
RAID	The metadevice name of the RAID5 device.

EXAMPLES

EXAMPLE 1 Recovering from Error Condition in RAID5 Metadevice

This example shows how to recover when a single component in a RAID5 metadevice is errored.

```
# metareplace d10 c3t0d0s2 c5t0d0s2
```

In this example, a RAID5 metadevice `d10` has an errored component, `c3t0d0s2`, replaced by a new component, `c5t0d0s2`.

EXAMPLE 2 Use of `-e` After Physical Disk Replacement

This example shows the use of the `-e` option after a physical disk in a submirror (a submirror of mirror `d11`, in this case) has been replaced.

```
# metareplace -e d11 c1t4d0s2
```

Note: The replacement disk must be partitioned to match the disk it is replacing before running the `metareplace` command.

EXIT STATUS

The following exit values are returned:

`0`	Successful completion.
`>0`	An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdu

SEE ALSO

`metaclear`(1M), `metadb`(1M), `metadetach`(1M), `metahs`(1M), `metainit`(1M), `metaoffline`(1M), `metaonline`(1M), `metaparam`(1M), `metaroot`(1M), `metaset`(1M), `metastat`(1M), `metasync`(1M), `metattach`(1M), `md.tab`(4), `md.cf`(4), `mddb.cf`(4), `attributes`(5)

Solaris Volume Manager Administration Guide

NAME

metaroot – setup system files for root (/) metadevice

SYNOPSIS

```
/usr/sbin/metaroot -h

/usr/sbin/metaroot [-n] [-k system-name] [-v vfstab-name]
     [-c mddb.cf-name] device
```

DESCRIPTION

The `metaroot` command edits the `/etc/vfstab` and `/etc/system` files so that the system may be booted with the root file system (/) on an appropriate metadevice. The only metadevices that support the root file system are a stripe with only a single disk or a mirror on a single-disk stripe.

If necessary, the `metaroot` command can reset a system that has been configured to boot the root file system (/) on a metadevice so that it uses a physical slice.

OPTIONS

Root privileges are required for all of the following options except `-h`.

The following options are supported:

`-c` *mddb.cf-name*	Uses *mddb.cf-name* instead of the default `/etc/lvm/mddb.cf` file as a source of metadevice database locations.
`-h`	Displays a usage message.
`-k` *system-name*	Edits a user-supplied *system-name* instead of the default `/etc/system` system configuration information file.
`-n`	Print what would be done without actually doing it.
`-v` *vfstab-name*	Edits *vfstab-name* instead of the default `/etc/vfstab` table of file system defaults.

OPERANDS

The following operands are supported:

device	Specifies either the metadevice or the conventional disk device (slice) used for the root file system (/).

EXAMPLES

EXAMPLE 1 Specifying Root File System on Metadevice

The following command edits `/etc/system` and `/etc/vfstab` to specify that the root file system is now on metadevice `d0`.

```
# metaroot d0
```

EXAMPLE 2 Specifying Root File System on SCSI Disk

The following command edits `/etc/system` and `/etc/vfstab` to specify that the root file system is now on the SCSI disk device `/dev/dsk/c0t3d0s0`.

```
# metaroot /dev/dsk/c0t3d0s0
```

FILES

`/etc/system`	System configuration information file. See `system`(4).

`/etc/vfstab`	File system defaults.
`/etc/lvm/mddb.cf`	Metadevice state database locations.

EXIT STATUS

The following exit values are returned:

`0`	Successful completion.
`>0`	An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdu

SEE ALSO

`metadb`(1M), `metainit`(1M), `metastat`(1M), `mddb.cf`(4), `system`(4), `vfstab`(4), `attributes`(5)

Solaris Volume Manager Administration Guide

NOTES

Soft partitions cannot be boot devices.

You can safely ignore the following messages. Their display is an artifact of the way drivers are loaded during the boot process.

`WARNING: forceload of misc/md_hotspares failed` might appear during boot if root is on a metadevice and no hot spares are specified. This can be eliminated by defining an empty hot spare pool.

`WARNING: forceload of misc/md_trans failed` might appear if no trans devices have been configured.

`WARNING: forceload of misc/md_raid failed` might appear if no RAID5 devices have been configured.

NAME

metaset – configure shared disksets

SYNOPSIS

/usr/sbin/metaset `-s` *setname* `-a -h` *hostname...*

/usr/sbin/metaset `-s` *setname* `-a [-l` *length*`]` *drivename...*

/usr/sbin/metaset `-s` *setname* `-d [-f] -h` *hostname...*

/usr/sbin/metaset `-s` *setname* `-d [-f]` *drivename...*

/usr/sbin/metaset `-s` *setname* `-r`

/usr/sbin/metaset `-s` *setname* `-t [-f]`

/usr/sbin/metaset `-s` *setname* `-b`

/usr/sbin/metaset `-s` *setname* `-o [-h` *hostname*`]`

/usr/sbin/metaset `[-s` *setname*`]`

/usr/sbin/metaset `[-s` *setname*`] -a | -d [ [m]` *mediator_host_list*`]`

DESCRIPTION

In a diskset configuration, two hosts are physically connected to the same set of disks. When one host fails, the other host has exclusive access to the disks. The `metaset` command administers sets of disks shared for exclusive (but not concurrent) access between such hosts. While disksets enable a high-availability configuration, Solaris Volume Manager itself does not actually provide a high-availability environment.

Shared metadevices/hot spare pools can be created only from drives which are in the diskset created by `metaset`. To create a set, one or more hosts must be added to the set. To create metadevices within the set, one or more devices must be added to the set. The *drivename* specified must be in the form c*x*t*x*d*x* with no slice specified.

Drives are repartitioned when they are added to a diskset only if Slice 7 is not set up correctly. A small portion of each drive is reserved in Slice 7 for use by Volume Manager. The remainder of the space on each drive is placed into Slice 0. Any existing data on the disks is lost after repartitioning. After adding a drive to a diskset, you can repartition the drive as necessary. However, Slice 7 should not be moved, removed, or overlapped with any other partition.

After a diskset is created and metadevices are set up within the set, the metadevice name will be in the following form:

`/dev/md/`*setname*`/{dsk,rdsk}/d`*number*

where *setname* is the name of the diskset, and *number* is the number of the metadevice (0-127).

Hot spare pools within local disksets use standard Volume Manager naming conventions. Hot spare pools with shared disksets use the following convention:

setname`/hsp`*number*

where *setname* is the name of the diskset, and *number* is the number of the hot spare pool (0-999).

Mediator Configuration

SVM provides support for a low-end HA solution consisting of two hosts that share only two strings of drives. The hosts in this type of configuration, referred to as *mediators* or mediator hosts, run a special daemon, `rpc.metamedd`(1M). The mediator hosts take on additional responsibilities to ensure that data is available in the case of host or drive failures.

A mediator configuration can survive the failure of a single host or a single string of drives, without administrative intervention. If both a host and a string of drives fail (multiple failures), the integrity of the data cannot be guaranteed. At this point, administrative intervention is required to make the data accessible. See `mediator`(7D) for further details.

Use the `-m` option, described below, to add or delete a mediator host.

OPTIONS

`-a` Adds drives or hosts to the named set. For a drive to be accepted into a set, the drive must not be in use within another metadevice or diskset, mounted on, or swapped on. When the drive is accepted into the set, it is repartitioned and the metadevice state database replica (for the set) may be placed on it. However, if a Slice 7 starts at cylinder 0, and is large enough to hold a state database replica, then the disk is not repartioned. Also, a drive is not accepted if it cannot be found on all hosts specified as part of the set. This means that if a host within the specified set is unreachable due to network problems, or is administratively down, the add will fail.

`-b` Insures that the replicas are distributed according to the replica layout algorithm. This can be invoked at any time, and will do nothing if the replicas are correctly distributed. In cases where the user has used the `metadb` command to manually remove or add replicas, this command can be used to insure that the distribution of replicas matches the replica layout algorithm.

`-d` Deletes drives or hosts from the named diskset. For a drive to be deleted, it must not be in use within the set. The last host cannot be deleted unless all of the drives within the set are deleted. Note that deleting the last host in a diskset destroys the diskset.

`-f` Forces one of three actions to occur: takes ownership of a diskset when used with `-t`; deletes the last disk drive from the diskset; or deletes the last host from the diskset. (Deleting the last drive or host from a diskset requires the `-d` option.) When used to forcibly take ownership of the diskset, this causes the diskset to be grabbed whether or not another host owns the set. All of the disks within the set are taken over (reserved) and fail fast is enabled, causing the other host to panic if it had diskset ownership. The metadevice state database will be read in by the host performing the take, and

the shared metadevices contained in the set will be accessible. The `-f` option is also used to delete the last drive in the diskset, because this drive would implicitly contain the last state database replica. The `-f` option is also used for deleting hosts from a set. When specified with a partial list of hosts, it can be used for one-host administration. One-host administration could be useful when a host is known to be non-functional, thus avoiding timeouts and failed commands. When specified with a complete list of hosts, the set is completely deleted. It is generally specified with a complete list of hosts to clean up after one-host administration has been performed.

`-h` *hostname...* — Specifies one or more host names to be added to or deleted from a diskset. Adding the first host creates the set. The last host cannot be deleted unless all of the drives within the set have been deleted. The host name is not accepted if all of the drives within the set cannot be found on the specified host. The host name is the same name found in `/etc/nodename`.

`-l` *length* — Sets the size (in blocks) for the metadevice state database replica. The length can only be set when adding a new drive; it cannot be changed on an existing drive. The default (and maximum) size is 8192 blocks, which should be appropriate for most configurations. The minimum size of the length is 64 blocks.

`-a | -d -m` *mediator_host_list* — Adds (`-a`) or deletes (`-d`) mediator hosts to the specified diskset. A *mediator_host_list* is the `nodename`(4) of the mediator host to be added and (for adding) up to two other aliases for the mediator host. The nodename and aliases for each mediator host are separated only by commas. Up to two mediator hosts can be specified for the named diskset. For deleting a mediator host, specify only the nodename of that host as the argument to `-m`.

In a single `metaset` command you can add or delete two mediator hosts. See EXAMPLES.

`-o` — Returns an exit status of 0 if the local host or the host specified with the `-h` option is the owner of the diskset.

`-r` — Releases ownership of a diskset. All of the disks within the set are released. The metadevices set up within the set are no longer accessible.

`-s` *setname* — Specifies the name of a diskset on which `metaset` will work. If no *setname* is specified, all disksets are returned.

`-t` — Takes ownership of a diskset safely. If `metaset` finds that another host owns the set, this host will not be allowed to take ownership of the set. If the set is not owned by any other host, all the disks within the set will be owned by the host on which `metaset` was

executed. The metadevice state database is read in, and the shared metadevices contained in the set become accessible. The `-t` option will take a diskset that has stale databases. When the databases are stale, `metaset` will exit code 66, and a message will be printed. At that point, the only operations permitted are the addition and deletion of replicas. Once the addition or deletion of the replicas has been completed, the diskset should be released and retaken to gain full access to the data.

EXAMPLES

EXAMPLE 1 Defining a Diskset

This example defines a diskset.

```
# metaset -s relo-red -a -h red blue
```

The name of the diskset is `relo-red`. The names of the first and second hosts added to the set are `red` and `blue`, respectively. (The hostname is found in `/etc/nodename`.) Adding the first host creates the diskset. A diskset can be created with just one host, with the second added later. The last host cannot be deleted until all of the drives within the set have been deleted.

EXAMPLE 2 Adding Drives to a Diskset

This example adds drives to a diskset.

```
# metaset -s relo-red -a c2t0d0 c2t1d0 c2t2d0 c2t3d0 c2t4d0 c2t5d0
```

The name of the previously created diskset is `relo-red`. The names of the drives are `c2t0d0`, `c2t1d0`, `c2t2d0`, `c2t3d0`, `c2t4d0`, and `c2t5d0`. Note that there is no slice identifier ("s*x*") at the end of the drive names.

EXAMPLE 3 Adding Multiple Mediator Hosts

The following command adds two mediator hosts to the specified diskset.

```
# metaset -s mydiskset -a -m myhost1,alias1 myhost2,alias2
```

FILES

`/etc/lvm/md.tab` Contains list of metadevice configurations.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdu

SEE ALSO

`metaclear`(1M), `metadb`(1M), `metadetach`(1M), `metahs`(1M), `metainit`(1M), `metaoffline`(1M), `metaonline`(1M), `metaparam`(1M), `metareplace`(1M), `metaroot`(1M), `metastat`(1M), `metasync`(1M), `metattach`(1M), `md.cf`(4), `md.tab`(4), `mddb.cf`(4), `attributes`(5)

Solaris Volume Manager Administration Guide

NOTES

Diskset administration, including the addition and deletion of hosts and drives, requires all hosts in the set to be accessible from the network.

NAME

metastat – display status for metadevice or hot spare pool

SYNOPSIS

```
/sbin/metastat -h

/sbin/metastat [-i] [-s setname] [-p] [-t] [metadevice...]
     [hot_spare_pool . . .]

/sbin/metastat [-i] [-s setname] [-p] component... [-v]
```

DESCRIPTION

The `metastat` command displays the current status for each metadevice (including stripes, concatenations, concatenations of stripes, mirrors, RAID5, soft partitions, and trans devices) or hot spare pool, or of specified metadevices, components, or hot spare pools.

It is helpful to run the `metastat` command after using the `metattach` command to view the status of the metadevice.

OPTIONS

`-h`	Displays usage message.
`-i`	Checks the status of all active metadevices and hot spares. The inquiry causes all components of each metadevice to be checked for accessibility, starting at the top level metadevice. When problems are discovered, the metadevice state databases are updated as if an error had occurred.
`-p`	Displays the list of active metadevices and hot spare pools in the same format as `md.tab`.
`-r`	Displays whether sub-devices are relocatable. At the end of the output, displays the devices and their associated device IDs.
`-s` *setname*	Specifies the name of the diskset on which `metastat` will work. Using the `-s` option causes the command to perform its administrative function within the specified diskset. Without this option, the command performs its function on metadevices and/or hot spare pools in the local diskset.
`-t`	Prints the current status and timestamp for the specified metadevices and hot spare pools. The timestamp provides the date and time of the last state change.

OPERANDS

component...	Displays the status of the component hosting a soft partition, including extents, starting blocks, and block count.
hot_spare_pool...	Displays the status of the specified hot spare pool(s).
metadevice...	Displays the status of the specified metadevice(s). If a trans metadevice is specified, the status of the master and log devices is also displayed.

EXAMPLES

EXAMPLE 1 Output Showing Mirror with Two Submirrors

The following example shows the partial output of the `metastat` command after creating a mirror, `d0`, consisting of two submirrors, `d70` and `d80`.

```
# metastat d0
d0: Mirror
    Submirror 0: d80
      State: Okay
    Submirror 1: d70
      State: Resyncing
    Resync in progress: 15 % done
    Pass: 1
    Read option: roundrobin (default)
    Write option: parallel (default)
    Size: 2006130 blocks
    .
    .
    .
```

EXAMPLE 2 Soft Partition on Mirror with Submirror

The following example shows the partial output of the `metastat` command after creating a soft partition, `d3`, on concat `d2`, which is built on a soft partition.

```
# metastat
d2: Concat/Stripe
    Size: 204800 blocks
    Stripe 0:
        Device              Start Block  Dbase State        Hot Spare
        d0                         0     No    Okay

d0: Soft Partition
    Component: c0t3d0s0
    Status: Okay
    Size: 204800 blocks
        Extent              Start Block  Block count
             0                     129       204800

d3: Soft Partition
    Component: d2
    Status: Okay
    Size: 202752 blocks
         Extent              Start Block  Block count
              0                     129       202752
```

WARNING

`metastat` displays states as of the time the command is entered. It is unwise to use the output of the `metastat -p` command to create a `md.tab`(4) file for a number of reasons:

- The output of `metastat -p` might show hot spares being used.
- It might show mirrors with multiple submirrors. See `metainit`(1M) for instructions for creating multi-way mirrors using `metainit` and `metattach`.

- A slice may go into an error state after `metastat -p` is issued.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdr

SEE ALSO

`mdmonitord`(1M), `metaclear`(1M), `metadb`(1M), `metadetach`(1M), `metahs`(1M), `metainit`(1M), `metaoffline`(1M), `metaonline`(1M), `metaparam`(1M), `metarecover`(1M), `metareplace`(1M), `metaroot`(1M), `metaset`(1M), `metasync`(1M), `metattach`(1M), `md.tab`(4), `md.cf`(4), `mddb.cf`(4), `attributes`(5)

Solaris Volume Manager Administration Guide

NAME

metasync – handle metadevice resync during reboot

SYNOPSIS

/usr/sbin/metasync -h

/usr/sbin/metasync [-s *setname*] [*buffer_size*] *metadevice*

/usr/sbin/metasync [-s *setname*] -r [*buffer_size*]

/usr/sbin/metasync -p *metadevice*

DESCRIPTION

The `metasync` command starts a resync operation on the specified *metadevice*. All components that need to be resynced are resynced. If the system crashes during a RAID5 initialization, or during a RAID5 resync, either an initialization or resync restarts when the system reboots.

Applications are free to access a metadevice at the same time that it is being resynced by `metasync`. Also, `metasync` performs the copy operations from inside the kernel, which makes the utility more efficient.

Use the `-r` option in boot scripts to resync all possible submirrors.

OPTIONS

`-h` Displays usage message.

`-r` Specifies that the `metasync` command handle special resync requirements during a system reboot. `metasync -r` should only be invoked from `/etc/rc2.d/S95svm.sync`. The `metasync` command only resyncs those metadevices that need to be resynced. `metasync` schedules all the mirror resyncs according to their pass numbers.

As root, you can edit `S95svm.sync` to specify the `-r 2048` option to `metasync`, so that resync following a reboot occurs as quickly as possible. See the description of *buffer_size*, below.

`-p` *metadevice* Regenerates parity information for RAID5 metadevices.

`-s` *setname* Specifies the name of the diskset on which `metasync` will work. Using the `-s` option will cause the command to perform its administrative function within the specified diskset. Without this option, the command will perform its function on local metadevices.

buffer_size Specifies the size (number of 512-byte disk blocks) of the internal copy buffer for the mirror resync. The size defaults to 128 512-byte disk blocks (64 Kbytes). It can be no more than 2048 blocks. For best performance (quickest completion of the resync), 2048 blocks is the recommended size.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdu

SEE ALSO

`metaclear`(1M), `metadb`(1M), `metadetach`(1M), `metahs`(1M), `metainit`(1M), `metaoffline`(1M), `metaonline`(1M), `metaparam`(1M), `metareplace`(1M), `metaroot`(1M), `metaset`(1M), `metastat`(1M), `metattach`(1M), `md.tab`(4), `md.cf`(4), `mddb.cf`(4), `attributes`(5)

Solaris Volume Manager Administration Guide

NAME

metattach, metadetach – attach or detach metadevice to or from a mirror or trans device, or attach space (blocks) to a soft partition to grow the soft partition

SYNOPSIS

/usr/sbin/metattach [-h]

/usr/sbin/metattach [-s *setname*] *mirror* [*metadevice*]

/usr/sbin/metattach [-s *setname*] [-i *size*] *concat/stripe component...*

/usr/sbin/metattach [-s *setname*] *RAID component...*

/usr/sbin/metattach [-s *setname*] *softpart size*

/usr/sbin/metattach [-s *setname*] *trans log*

/usr/sbin/metadetach [-s *setname*] [-f] *mirror submirror*

/usr/sbin/metadetach [-s *setname*] [-f] *trans*

DESCRIPTION

`metattach` is used to add submirrors to a mirror, add logging devices to trans devices, grow metadevices, or grow soft partitions. Growing metadevices can be done without interrupting service. To grow the size of a mirror or trans, the slices must be added to the submirrors or to the master devices.

Solaris Volume Manager supports one-to-four-way mirrors. Thus, you can only attach a metadevice to a mirror if there are three or fewer submirrors beneath the mirror. Once a new metadevice is attached to a mirror, `metattach` will automatically start a resync operation to the new submirror.

Attaching a new logging device to a busy trans metadevice is allowed, although a trans metadevice will start using its new logging device only after the trans is idle (after it is unmounted, for example). The busy trans will be in an *Attaching* state (`metastat`) until the logging device is actually attached. Attaching a logging device in the *Hard Error* or *Error* state (`metastat`) is not allowed.

`metadetach` is used to detach submirrors from mirrors and to detach logging devices from trans metadevices.

When a submirror is detached from a mirror, it is no longer part of the mirror, thus reads and writes to and from that metadevice via the mirror are no longer performed through the mirror. Detaching the only existing submirror is not allowed. Detaching a submirror that has slices reported as needing maintenance (by `metastat`) is not allowed unless the `-f` (force) flag is used.

`metadetach` also detaches the logging device from a trans. Once detached, the logging device is no longer part of the trans, thus the trans is no longer logging and all benefits of logging are lost. Any information on the logging device that pertains to the master device is written to the master device before the logging device is detached.

Detaching the logging device from a busy trans device is not allowed unless the `-f` (force) flag is used. Even so, the logging device is not actually detached until the trans is idle. The trans is in the *Detaching* state (`metastat`) until the logging device is detached.

OPTIONS

Root privileges are required for all of the following options except `-h`.

`-f` Force the detaching of metadevices that have components that need maintenance or are busy. You can use this option only when a mirror is in a maintenance state that can be fixed with `metareplace`(1M). If the mirror is in a maintenance state that can only be fixed with `metasync`(1M) (as shown by the output of `metastat`(1M)), `metadetach -f` will have no effect, because the mirrors must be resynchronized before one of them can be detached.

`-h` Displays a usage message.

`-i` *size* Specifies the interlace value for stripes, where `size` is a specified value followed by either '`k`' for kilobytes, '`m`' for megabytes, or '`b`' for blocks. The units can be either uppercase or lowercase. If *size* is not specified, the size defaults to the interlace size of the last stripe of the metadevice. When an interlace size change is made on a stripe, it will be carried forward on all stripes that follow.

`-s` *setname* Specifies the name of the diskset on which the `metattach` command or the `metadetach` command will work. Using the `-s` option will cause the command to perform its administrative function within the specified diskset. Without this option, the command will perform its function on local metadevices.

OPERANDS

component... The logical name for the physical slice (partition) on a disk drive, such as `/dev/dsk/c0t0d0s2`, being added to the concatenation, stripe, concatenation of stripes, or RAID5 metadevice.

concat/stripe Specifies the metadevice name of the concatenation, stripe, or concatenation of stripes.

log Specifies the metadevice name of the logging device to be attached to the trans metadevice.

metadevice Specifies the name of the metadevice to be attached to the mirror as a submirror. This metadevice must have been previously created by the `metainit` command.

mirror Specifies the mirror.

RAID Specifies the metadevice name of the RAID5 metadevice.

size Specifies the amount of space to add to the soft partition in `K` or `k` for kilobytes, `M` or `m` for megabytes, `G` or `g` for gigabytes, `T` or `t` for terabytes, and `B` or `b` for blocks (sectors).

softpart Specifies the metadevice name of the existing soft partition.

submirror	The metadevice name of the submirror to be detached from the mirror.
trans	Specifies the metadevice name of the trans metadevice (not the master or logging device).

EXAMPLES

EXAMPLE 1 Concatenating a New Slice to a Metadevice

This example concatenates a single new slice to an existing metadevice, `d8`. Afterwards, you would use the `growfs`(1M) command to expand the file system.

```
# metattach d8 /dev/dsk/c0t1d0s2
```

EXAMPLE 2 Detaching Logging Device from Trans Metadevice

This example detaches the logging device from a trans metadevice `d9`. Notice that you do not have to specify the logging device itself, as there can only be one.

```
# metadetach d9
```

EXAMPLE 3 Expanding a RAID5 Metadevice

This example expands a RAID5 metadevice, `d45`, by attaching another slice.

```
# metattach d45 /dev/dsk/c3t0d0s2
```

When you add additional slices to a RAID5 metadevice, the additional space is devoted to data. No new parity blocks are allocated. The data on the added slices is, however, included in the overall parity calculations, so it is protected against single-device failure.

EXAMPLE 4 Expanding a Soft Partition

The following example expands a soft partition, `d42`, by attaching 150 Mbytes.

```
# metattach d42 150M
```

When you add additional space to a soft partition, the additional space is taken from any available space on the slice and might not be contiguous with the existing soft partition.

EXAMPLE 5 Adding Space to Two-Way Mirror

This example adds space to a two-way mirror by adding a slice to each submirror. Afterwards, you would use the `growfs`(1M) command to expand the file system.

```
# metattach d9 /dev/dsk/c0t2d0s5
# metattach d10 /dev/dsk/c0t3d0s5
```

EXAMPLE 5 Adding Space to Two-Way Mirror *(Continued)*

EXAMPLE 6 Detaching a Submirror from a Mirror

This example detaches a submirror, `d2`, from a mirror, `d4`.

```
# metadetach d4 d2
```

EXAMPLE 7 Adding Four Slices to Metadevice

This example adds four slices to an existing metadevice, `d9`. Afterwards, you would use the `growfs`(1M) command to expand the file system.

```
# metattach d9 /dev/dsk/c0t1d0s2 /dev/dsk/c0t2d0s2 \
     /dev/dsk/c0t3d0s2 /dev/dsk/c0t4d0s2
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdu

SEE ALSO

`metaclear`(1M), `metadb`(1M), `metadetach`(1M), `metahs`(1M), `metainit`(1M), `metaoffline`(1M), `metaonline`(1M), `metaparam`(1M), `metarecover`(1M), `metareplace`(1M), `metaroot`(1M), `metaset`(1M), `metastat`(1M), `metasync`(1M), `md.tab`(4), `md.cf`(4), `mddb.cf`(4), `attributes`(5)

Solaris Volume Manager Administration Guide

WARNING

When a submirror is detached from its mirror, the data on the metadevice may not be the same as the data that existed on the mirror prior to running `metadetach`. In particular, if the `-f` option was needed, the metadevice and mirror probably do not contain the same data.

NAME

mib2mof – generate MOF file(s) from input SNMP MIB file(s)

SYNOPSIS

/usr/sadm/bin/mib2mof [-n] [-d *directory*] [-q] [-c] [-a] [-h] *files*

DESCRIPTION

The mib2mof utility reads input Management Information Base (MIB) files and produces one or more Managed Object Format (MOF) files. MOF files contain a Common Information Model (CIM) class declaration that represents the MIB for the Solaris Simple Network Management Protocol (SNMP) provider. The SNMP provider allows Web-Based Enterprise Management (WBEM) applications to access SNMP device information.

SNMP scalar variables map to properties in the CIM class. Qualifiers on each property convey the following MIB information for each scalar variable:

- syntax
- read/write access
- OID (Object IDentifier)
- description (optional)
- index (if the variable is within a group [sequence] that defines a row)

The syntax of an SNMP scalar variable is represented in a CIM class by the property's CIM datatype. All properties are marked with write access (true or false).

The following table shows how a Solaris SNMP datatype in a MIB maps to a Web-Based Enterprise Management (WBEM) CIM datatype and then to an SNMP datatype used by the WBEM SNMP API:

SNMP SMI Datatype	SNMP Ver.	CIM Datatype	SNMP API Object type
INTEGER	v1	sint32	SnmpInt
OCTET STRING	v1	string	SnmpString
OBJECT IDENTIFIER	v1	string	SnmpOid
IpAddress	v1	string	SnmpIpAddress
Counter	v1	uint32	SnmpCounter
Gauge	v1	uint32	SnmpGauge
TimeTicks	v1	uint32	SnmpTimeticks
Opaque	v1	sint8[]	SnmpOpaque
DisplayString - see OCTET STRING	v1		
NetworkAddress - see IpAddress	v1		
Counter32 - see Counter	v2		
Counter64	v2	uint64	SnmpCounter64
Integer32	v2	sint32	SnmpInt
Gauge32 - see Gauge	v2		
Unsigned32	v2	uint32	SnmpGauge
TruthValue	v2	sint32	SnmpInt
BITS - see OCTET STRING	v2		

The mib2mof utility includes its required Solaris_SNMPmib_core.txt file (containing core MIB definitions), installed in /usr/sadm/mof. The mib2mof utility looks first for mib core file in local directory. If this file is not found in the local directory, mib2mof looks in /usr/sadm/mof.

A MOF file is generated for each SNMP group and table row sequence (that is, the columns in one row) found in the supplied MIBs. (This does not include the core MIB definitions contained in the `Solaris_SNMPmib_core.txt file`.)

There is no MOF file or property for an SNMP table - all table access is through the rows and columns of the table, and the SNMP variable for the table is marked as inaccessible in the MIB.

The MOF file created contains a CIM class that represents an SNMP group or row and a CIM class to represent a CIM association. The output file name (and CIM class) is of the format <SNMP_><MIB name><Group name>.mof.

OPTIONS

The following options are supported:

`-a` Generate MOF files for all of the input MIB files. If `-a` is not given, a MOF file is generated only for the last file of the input list.

`-c` Do not use the default `Solaris_SNMPmib_core.txt` definitions file shipped with the Solaris SNMP Provider for WBEM. If this option is specified, you must specify another `MIB_CORE` definitions file as one of the input files.

`-d` *directory* Generate output MOF files in the specified directory.

`-h` Show how to invoke `mib2mof` and list its arguments.

`-n` Parse the input MIB files without generating any output.

`-q` Include the `DESCRIPTION` clause of `SNMP OBJECT-TYPE` as a qualifier in the generated MOF file.

OPERANDS

The following operands are supported:

files List of SNMP MIB files to be converted.

EXIT STATUS

The `mib2mof` utility terminates with exit status `0`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWwbcou

SEE ALSO

`init.wbem`(1M), `mofcomp`(1M), `wbemadmin`(1M), `attributes`(5)

NAME

mibiisa – Sun SNMP Agent

SYNOPSIS

mibiisa [-ar] [-c *config-dir*] [-d *debug-level*] [-p *port*] [-t *cache-timer*]

DESCRIPTION

The mibiisa utility is an RFC 1157-compliant SNMP agent. It supports MIB-II as defined in *RFC 1213*, with Sun extensions under Sun's enterprise number. The MIB (Management Information Base) is both readable and writable. The mibiisa utility supports all SNMP protocol operations including GET-REQUEST, GETNEXT-REQUEST, SET-REQUEST, GET-REPLY, and TRAP.

The mibiisa utility supports the coldStart, linkUp, linkDown, and authentication traps. The authentication trap may be disabled by a command-line switch, which itself may be overridden by a management station writing to a MIB variable in the standard SNMP MIB group.

The mibiisa utility supports four distinct views of the MIB. The view used for any request is determined by the community string contained in that request.

To enhance security, mibiisa supports an option to block all writes to the MIB. You can also limit the set of management stations from which the agent will accept requests in the configuration file used when starting the mibiisa. See the SECURITY section for more information.

Unless overridden, mibiisa uses UDP port 161, the standard SNMP port. The mibiisa utility issues traps through the same port on which it receives SNMP requests.

The mibiisa utility must run with super-user privileges and is typically started at system startup via /etc/rc3.d. mibiisa may not be started using inetd(1M). When started, mibiisa detaches itself from the keyboard, disables all signals except SIGKILL, SIGILL, SIGUSR1, and SIGUSR2, and places itself in the background.

OPTIONS

The following options are supported:

-a
: Disable the generation of authentication traps. However, an SNMP manager may write a value into snmpEnableAuthenTraps to enable or disable authentication traps.

-c *config-dir*
: Specify a directory where it expects snmpd.conf file, on startup. The default directory is /etc/snmp/conf.

-d *debug-level*
: Debug. A value of 0 disables all debug and is the default. Levels 1 through 3 represent increasing levels of debug output. When mibiisa receives the signal SIGUSR1, it resets the debug-level to 0. When mibiisa receives the signal SIGUSR2, it increments the debug-level by one.

  Debug output is sent to the standard output in effect at the time mibiisa is started. No matter what debug level is in effect, certain significant events are logged in the system log.

`-p` *port*	Define an alternative UDP port on which `mibiisa` listens for incoming requests. The default is UDP port 161.
`-r`	Place the MIB into read-only mode.
`-t` *cache-timer*	By default, information fetched from the kernel is considered to be valid for 45 seconds from the time it is retrieved. This cache lifetime may be altered with this parameter. You cannot set *cache-timer* to any value less than 1.

CONFIGURATION FILE

The `snmpd.conf` file is used for configuration information. Each entry in the file consists of a keyword followed by a parameter string. The keyword must begin in the first position. Parameters are separated from the keyword and from one another by white space. Case in keywords is ignored. Each entry must be contained on a single line. All text following (and including) a pound sign (#) is ignored. Keywords currently supported are:

`sysdescr`	The value to be used to answer queries for sysDescr.
`syscontact`	The value to be used to answer queries for sysContact.
`syslocation`	The value to be used to answer queries for sysLocation.
`trap`	The parameter names one or more hosts to receive traps. Only five hosts may be listed.
`system-group-read-community`	The community name to get read access to the system group and Sun's extended system group.
`system-group-write-community`	The community name to get write access to the system group and Sun's extended system group.
`read-community`	The community name to get read access to the entire MIB.
`write-community`	The community name to get write access to the entire MIB (implies read access).
`trap-community`	The community name to be used in traps.
`kernel-file`	The name of the file to use for kernel symbols.
`managers`	The names of hosts that may send SNMP queries. Only five hosts may be listed on any one line. This keyword may be repeated for a total of 32 hosts.

`newdevice`	The additional devices which are not built in SNMPD. The format is as follows: *newdevice* `type` *speed name* where *newdevice* is the keyword, `type` is an integer which has to match your schema file, *speed* is the new device's speed, and *name* is this new device's name.

An example `snmpd.conf` file is shown below:

```
sysdescr        Sun SNMP Agent, SPARCstation 10, Company
                   Property Number 123456
syscontact      Cliff Claven
sysLocation     Stool next to Norms at Cheers
#
system-group-read-community     public
system-group-write-community    private
#
read-community  all_public
write-community all_private
#
trap            localhost
trap-community  SNMP-trap
#
#kernel-file    /vmunix
#
managers        lvs golden
managers        swap
```

INSTALLATION

The `mibiisa` utility and its configuration file, `snmpd.conf`, may be placed in any directory. However for Solaris 2.4 and subseqent releases, use `/usr/lib/snmp` for `mibiisa` itself and `/etc/snmp/conf` for the configuration file. You can modify the configuration file as appropriate. If you make any changes to `snmpd.conf` file keyword values, you must kill and restart `mibiisa` for the changes to take effect.

Your `/etc/services` file (or NIS equivalent) should contain the following entries:

```
snmp            161/udp                         # Simple Network
                                                Mgmt Protocol

snmp-trap       162/udp         snmptrap        # SNMP trap
                                                (event) messages
```

The following is an example for Solaris 2.*x*:

```
#
# Start the SNMP agent
#
if [ -f /etc/snmp/conf/snmpd.conf -a -x
   /usr/lib/snmp/mibiisa ];
then
```

```
/opt/SUNWconn/snm/agents/snmpd
echo 'Starting SNMP-agent.'
```

SECURITY

SNMP, as presently defined, offers relatively little security. The `mibiisa` utility accepts requests from other machines, which can have the effect of disabling the network capabilities of your computer. To limit the risk, the configuration file lets you specify a list of up to 32 manager stations from which `mibiisa` will accept requests. If you do not specify any such manager stations, `mibiisa` accepts requests from anywhere.

The `mibiisa` utility also allows you to mark the MIB as "read-only" by using the `-r` option.

Finally, `mibiisa` supports four different community strings. These strings, however, are visible in the configuration file and within the SNMP packets as they flow on the network.

The configuration file should be owned by, and readable only by super-user. In other words the mode should be:

```
-rw-------  1 root         2090 Oct 17 15:04 /etc/snmp/conf/snmpd.conf
```

MIB

This section discusses some of the differences between the `mibiisa` MIB and the standard MIB-II (as defined in RFC 1213).

The following variables are read-only in the `mibiisa` MIB:

```
sysName
atIfIndex
ipDefaultTTL
```

These variables are read-write in the standard MIB-II.

The `mibiisa` MIB Address Translation tables support limited write access: only atPhysAddress may be written, either to change the physical address of an existing entry or to delete an entire ARP table entry.

The `mibiisa` MIB IP Net to Media table supports limited write access: only ipNetToMediaPhysAddress and ipNetToMediaType may be written, either to change the physical address of an existing entry or to delete an entire ARP table entry.

The following variables are read-write in the `mibiisa` MIB; however, these variables have fixed values. Any new values "set" to them are accepted, but have no effect:

```
ipRoutIfIndex
ipRouteMetric1
ipRouteMetric2
ipRouteMetric3
ipRouteMetric4
ipRouteType
ipRouteAge
```

```
ipRouteMask
ipRouteMetric5
```

The following `mibiisa` MIB variable reflects the actual state of the related table entry. "Sets" are accepted but have no effect:

```
tcpConnState
```

The following `mibiisa` MIB variables are readable, but return a fixed value:

`icmpInDestUnreachs`	Returns 1
`icmpInTimeExcds`	Returns 1
`icmpInParmProbs`	Returns 1
`icmpInSrcQuenchs`	Returns 1
`icmpInRedirects`	Returns 1
`icmpInEchos`	Returns 1
`icmpInEchoReps`	Returns 1
`icmpInTimestamps`	Returns 1
`icmpInTimestampReps`	Returns 1
`icmpInAddrMasks`	Returns 1
`icmpInAddrMaskReps`	Returns 1
`icmpOutDestUnreachs`	Returns 1
`icmpOutTimeExcds`	Returns 1
`icmpOutParmProbs`	Returns 1
`icmpOutSrcQuenchs`	Returns 1
`icmpOutRedirects`	Returns 1
`icmpOutEchos`	Returns 1
`icmpOutEchoReps`	Returns 1
`icmpOutTimestamps`	Returns 1
`icmpOutTimestampReps`	Returns 1
`icmpOutAddrMasks`	Returns 1
`icmpOutAddrMaskReps`	Returns 1
`ifInUnknownProtos`	Returns 0

ipAdEntBcastAddr	Returns 1
ipAdEntReasmMaxSiz	Returns 65535
ipRouteMetric1	Returns –1
ipRouteMetric2	Returns –1
ipRouteMetric3	Returns –1
ipRouteMetric4	Returns –1
ipRouteAge	Returns 0
ipRouteMetric5	Returns –1
ipNetToMediaType	Returns (3) dynamic
ipRoutingDiscards	Returns 0

The following variables return a fixed value of `0` for drivers not conforming to the GLD framework (see `gld`(7D)), including the old LAN drivers on SPARC machines:

ifInOctets	Returns 0
ifInNUcastPkts	Returns 0
ifInDiscards	Returns 0
ifOutOctets	Returns 0
ifOutNUcastPkts	Returns 0
ifOutDiscards	Returns 0

SCHEMA ATTRIBUTES

The following describes the attributes in the group and table definitions in the `/var/snmp/mib/sun.mib` file.

system

The `system` group reports statistics about a particular system (for example, a workstation or a printer).

`sysDescr` – A textual description of the entity. This value should include the full name and version identification of the system's hardware type, software operating-system, and networking software. This value must only contain printable ASCII characters. (string[255])

`sysObjectID` – The vendor's authoritative identification of the network management subsystem contained in the entity. This value is allocated within the SMI enterprises subtree (1.3.6.1.4.1) and provides an easy and unambiguous means for determining what type of equipment is being managed. For example, if vendor "Flintstones, Inc." was assigned the subtree 1.3.6.1.4.1.4242, it could assign the identifier 1.3.6.1.4.1.4242.1.1 to its "Fred Router." (objectid)

`sysUpTime` – Time (in hundredths of a second) since the network management portion of the system was last reinitialized. (timeticks)

`sysContact` – The textual identification of the contact person for this managed node, together with information on how to contact this person. (string[255])

`sysName` – An administratively-assigned name for this managed node. By convention, this is the node's fully-qualified domain name. (string[255])

`sysLocation` – The physical location of this node (for example, "telephone closet, 3rd floor" (string[255]))

`sysServices` – A value indicating the set of services that this entity primarily offers. (int) The value is a sum. This sum initially takes the value zero. Then, for each layer L in the range 1 through 7 for which this node performs transactions, 2 raised to (L - 1) is added to the sum. For example, a node that performs primarily routing functions would have a value of 4 (2**(3-1)). In contrast, a node that is a host offering application services would have a value of 72 (2**(4-1) + 2**(7-1)). Note that in the context of the Internet suite of protocols, values should be calculated accordingly:

`layer`	`functionality`
1	physical (such as repeaters)
2	datalink/subnetwork (such as bridges)
3	internet (such as IP gateways)
4	end-to-end (such as IP hosts)
7	applications (such as mail relays)

For systems including OSI protocols, Layers 5 and 6 may also be counted.

interfaces The `interfaces` group reports the number of interfaces handled by the agent.

`ifNumber` – The number of network interfaces, regardless of their current state, present on this system. (int)

ifTable The `ifTable` is a table of interface entries. The number of entries is given by the value of ifNumber.

`ifIndex` – A unique value for each interface. Its value ranges between 1 and the value of ifNumber. The value for each interface must remain constant at least from one reinitialization of the entity's network management system to the next reinitialization. (int)

`ifDescr` – A textual string containing information about the interface. This string should include the name of the manufacturer, the product name, and the version of the hardware interface. (string[255])

`ifType` – The type of interface, distinguished according to the physical/link protocol(s) immediately below the network layer in the protocol stack. (enum)

`ifMtu` – The size of the largest datagram that can be sent/received on the interface, specified in octets. For interfaces used for transmitting network datagrams, this is the size of the largest network datagram that can be sent on the interface. (int)

`ifSpeed` – An estimate of the interface's current bandwidth in bits-per-second. For interfaces that do not vary in bandwidth, or for those where no accurate estimation can be made, this object should contain the nominal bandwidth. (gauge)

`if1hysAddress` – The interface's address at the protocol layer immediately below the network layer in the protocol stack. For interfaces without such an address (for example, a serial line), this object should contain an octet string of zero length. (octet[128])

`ifAdminStatus` – The desired state of the interface. The testing(3) state indicates that no operational packets can be passed. (enum)

`if OperStatus` – The current operational state of the interface. The testing(3) state indicates that no operational packets can be passed. (enum)

`ifLastChange` – The value of sysUpTime at the time the interface entered its current operational state. If the current state was entered prior to the last reinitialization of the local network management subsystem, then this object contains a zero value. (timeticks)

`ifInOctets` – The total number of octets received on the interface, including framing characters. (counter) Returns a fixed value of 0.

`ifInUcastPkts` – The number of subnetwork-unicast packets delivered to a higher-layer protocol. (counter)

`ifInNUcastPkts` – The number of non-unicast (that is, subnetwork- broadcast or subnetwork-multicast) packets delivered to a higher-layer protocol. (counter) Returns a fixed value of 0.

`ifInDiscards` – The number of inbound packets chosen to be discarded, even though no errors had been detected to prevent their being deliverable to a higher-layer protocol. One possible reason for discarding such a packet could be to free up buffer space. (counter) Returns a fixed value of 0.

`ifInErrors` – The number of inbound packets that contained errors preventing them from being deliverable to a higher-layer protocol. (counter)

`ifInUnknownProtos` – The number of packets received via the interface that were discarded because of an unknown or unsupported protocol. (counter) Returns a fixed value of 0.

`ifOutOctets` – The total number of octets transmitted out of the interface, including framing characters. (counter) Returns a fixed value of 0.

`ifOutUcastPkts` – The total number of packets that higher-level protocols requested be transmitted to a subnetwork-unicast address, including those that were discarded or not sent. (counter)

`ifOutNUcastPkts` – The total number of packets that higher-level protocols requested be transmitted to a non- unicast (that is, a subnetwork-broadcast or subnetwork-multicast) address, including those that were discarded or not sent. (counter) Returns a fixed value of 0.

`ifOutDiscards` – The number of outbound packets that were chosen to be discarded even though no errors had been detected to prevent their being transmitted. One possible reason for discarding such a packet could be to free up buffer space. (counter) Returns a fixed value of 0.

`ifOutErrors` – The number of outbound packets that could not be transmitted because of errors. (counter)

`ifOutQLen` – The length of the output packet queue (in packets). (gauge)

`ifSpecific` – A reference to MIB definitions specific to the particular media being used to realize the interface. For example, if the interface is realized by an Ethernet, then the value of this object refers to a document defining objects specific to Ethernet. If this information is not present, its value should be set to the OBJECT IDENTIFIER { 0 0 }, which is a syntactically valid object identifier. Any conformant implementation of ASN.1 and BER must be able to generate and recognize this value. (objectid)

atTable `atTable` Address Translation tables contain the NetworkAddress to physical address equivalences. Some interfaces do not use translation tables for determining address equivalences (for example, DDN-X.25 has an algorithmic method). If all interfaces are of this type, then the Address Translation table is empty, that is, has zero entries.

`atIfIndex` – The interface on which this entry's equivalence is effective. The interface identified by a particular value of this index is the same interface as identified by the same value of ifIndex. (int)

`atPhysAddress` – The media-dependent physical address. (octet[128]) Setting this object to a null string (one of zero length) has the effect of invaliding the corresponding entry in the atTable object. That is, it effectively dissociates the interface identified with said entry from the mapping identified with said entry. It is an implementation-specific matter as to whether the agent removes an invalidated entry from the table. Accordingly, management stations must be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such entries requires examination of the relevant atPhysAddress object.

`atNetAddress` – The NetworkAddress (that is, the IP address) corresponding to the media-dependent physical address. (netaddress)

ip The `ip` group reports statistics about the Internet Protocol (IP) group.

`ipForwarding` – The indication of whether this entity is acting as an IP gateway in respect to the forwarding of datagrams received by, but not addressed to, this entity. IP gateways forward datagrams. IP hosts do not— except those source-routed via the host. (enum)

Note that for some managed nodes, this object may take on only a subset of the values possible. Accordingly, it is appropriate for an agent to return a "badValue" response if a management station attempts to change this object to an inappropriate value.

`ipDefaultTTL` – The default value inserted into the Time-To-Live field of the IP header of datagrams originated at this entity, whenever a TTL value is not supplied by the transport layer protocol. (int)

`ipInReceives` – The total number of input datagrams received from interfaces, including those received in error. (counter)

`ipInHdrErrors` – The number of input datagrams discarded due to errors in their IP headers, including bad checksums, version number mismatch, other format errors, time-to-live exceeded, errors discovered in processing their IP options, and so on. (counter)

`ipInAddrErrors` – The number of input datagrams discarded because the IP address in their IP header's destination field was not a valid address to be received at this entity. This count includes invalid addresses (for example, 0.0.0.0) and addresses of unsupported Classes (for example, Class E). For entities that are not IP Gateways and therefore do not forward datagrams, this counter includes datagrams discarded because the destination address was not a local address. (counter)

`ipForwDatagrams` – The number of input datagrams for which this entity was not their final IP destination, as a result of which an attempt was made to find a route to forward them to that final destination. In entities that do not act as IP Gateways, this counter will include only those packets that were Source-Routed via this entity, and the Source- Route option processing was successful. (counter)

`ipInUnknownProtos` – The number of locally-addressed datagrams received successfully but discarded because of an unknown or unsupported protocol. (counter)

`ipInDiscards` – The number of input IP datagrams for which no problems were encountered to prevent their continued processing, but which were discarded, for example, for lack of buffer space. Note that this counter does not include any datagrams discarded while awaiting reassembly. (counter)

`ipInDelivers` – The total number of input datagrams successfully delivered to IP user-protocols (including ICMP). (counter)

`ipOutRequests` – The total number of IP datagrams that local IP user-protocols (including ICMP) supplied to IP in requests for transmission. Note that this counter does not include any datagrams counted in ipForwDatagrams. (counter)

`ipOutDiscards` – The number of output IP datagrams for which no problem was encountered to prevent their transmission to their destination, but which were discarded (for example, for lack of buffer space). Note that this counter would include datagrams counted in ipForwDatagrams if any such packets met this (discretionary) discard criterion. (counter)

`ipOutNoRoutes` – The number of IP datagrams discarded because no route could be found to transmit them to their destination. Note that this counter includes any packets counted in ipForwDatagrams which meet this "no-route" criterion. Note that this includes any datagrams that a host cannot route because all its default gateways are down. (counter)

`ipReasmTimeout` – The maximum number of seconds that received fragments are held while they are awaiting reassembly at this entity. (int)

`ipReasmReqds` – The number of IP fragments received that needed to be reassembled at this entity. (counter)

`ipReasmOKs` – The number of IP datagrams successfully reassembled. (counter)

`ipReasmFails` – The number of failures detected by the IP reassembly algorithm, for whatever reason: timed out, errors, and the like. Note that this is not necessarily a count of discarded IP fragments since some algorithms (notably the algorithm in RFC 815) can lose track of the number of fragments by combining them as they are received. (counter)

`ipFragOKs` – The number of IP datagrams that have been successfully fragmented at this entity. (counter)

`ipFragFails` – The number of IP datagrams that have been discarded because they needed to be fragmented at this entity but could not be, for example, because their "Don't Fragment" flag was set. (counter)

`ipFragCreates` – The number of IP datagram fragments that have been generated as a result of fragmentation at this entity. (counter)

`ipRoutingDiscards` – The number of routing entries that were chosen to be discarded even though they were valid. One possible reason for discarding such an entry could be to free-up buffer space for other routing entries. (counter) Returns a fixed value of 0.

ipAddrTable

`ipAddrTable` is a table of addressing information relevant to this entity's IP addresses.

`ipAdEntAddr` – The IP address to which this entry's addressing information pertains. (netaddress)

`ipAdEntIfIndex` – The index value that uniquely identifies the interface to which this entry is applicable. The interface identified by a particular value of this index is the same interface as identified by the same value of ifIndex. (int)

`ipAdEntNetMask` – The subnet mask associated with the IP address of this entry. The value of the mask is an IP address with all the network bits set to 1, and all the hosts bits set to 0. (netaddress)

`ipAdEntBcastAddr` – The value of the least-significant bit in the IP broadcast address used for sending datagrams on the (logical) interface associated with the IP address of this entry. For example, when the Internet standard all-ones broadcast address is used, the value will be 1. This value applies to both the subnet and network broadcasts addresses used by the entity on this (logical) interface. (int) Returns a fixed value of 1.

`ipAdEntReasmMaxSize` – The size of the largest IP datagram that this entity can reassemble from incoming IP fragmented datagrams received on this interface. (int) Returns a fixed value of 65535.

ipRouteTable The `ipRouteTable` is this entity's IP Routing table.

`ipRouteDest` – The destination IP address of this route. An entry with a value of 0.0.0.0 is considered a default route. Multiple routes to a single destination can appear in the table, but access to such multiple entries is dependent on the table- access mechanisms defined by the network management protocol in use. (netaddress)

`ipRouteIfIndex` – The index value that uniquely identifies the local interface through which the next hop of this route should be reached. The interface identified by a particular value of this index is the same interface as identified by the same value of ifIndex. (int)

`ipRouteMetric1` – The primary routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's ipRouteProto value. If this metric is not used, its value should be set to –1. (int) Returns a fixed value of –1.

`ipRouteMetric2` – An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's ipRouteProto value. If this metric is not used, its value should be set to –1. (int) Returns a fixed value of –1.

`ipRouteMetric3` – An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's ipRouteProto value. If this metric is not used, its value should be set to –1. (int) Returns a fixed value of –1.

`ipRouteMetric4` – An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's ipRouteProto value. If this metric is not used, its value should be set to –1. (int) Returns a fixed value of –1.

`ipRouteNextHop` – The IP address of the next hop of this route. (In the case of a route bound to an interface that is realized via a broadcast media, the value of this field is the agent's IP address on that interface.) (netaddress)

`ipRouteType` – The type of route. Note that the values direct (3) and indirect (4) refer to the notion of direct and indirect routing in the IP architecture. (enum)

Setting this object to the value invalid (2) has the effect of invalidating the corresponding entry in the ipRouteTable object. That is, it effectively dissociates the destination identified with said entry from the route identified with said entry. It is an implementation-specific matter as to whether the agent removes an invalidated entry from the table. Accordingly, management stations must be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such entries requires examination of the relevant ipRouteType object.

`ipRouteProto` – The routing mechanism through which this route was learned. Inclusion of values for gateway routing protocols is not intended to imply that hosts should support those protocols. (enum)

`ipRouteAge` – The number of seconds since this route was last updated or otherwise determined to be correct. Note that no semantics of "too old" can be implied except through knowledge of the routing protocol by which the route was learned. (int) Returns a fixed value of 0.

`ipRouteMask` – Indicate the mask to be logical-ANDed with the destination address before being compared to the value in the ipRouteDest field. For those systems that do not support arbitrary subnet masks, an agent constructs the value of the ipRouteMask by determining whether the value of the correspondent ipRouteDest field belongs to a class-A, B, or C network, and then using one of:

`mask`	`network`
255.0.0.0	class-A
255.255.0.0	class-B
255.255.255.0	class-C

If the value of the ipRouteDest is 0.0.0.0 (a default route), then the mask value is also 0.0.0.0. It should be noted that all IP routing subsystems implicitly use this mechanism. (netaddress)

`ipRouteMetric5` – An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's ipRouteProto value. If this metric is not used, its value should be set to –1. (int) Returns a fixed value of –1.

`ipRouteInfo` – A reference to MIB definitions specific to the particular routing protocol responsible for this route, as determined by the value specified in the route's ipRouteProto value. If this information is not present, its value should be set to the OBJECT IDENTIFIER { 0 0 }, which is a syntactically valid object identifier. Any conformant implementation of ASN.1 and BER must be able to generate and recognize this value. (objectid)

ipNetToMediaTable The `ipNetToMediaTable` is the IP Address Translation table used for mapping from IP addresses to physical addresses.

`ipNetToMediaIfIndex` – The interface on which this entry's equivalence is effective. The interface identified by a particular value of this index is the same interface as identified by the same value of ifIndex. (int)

`ipNetToMediaPhysAddress` – The media-dependent physical address. (octet[128])

`ipNetToMediaNetAddress` – The IpAddress corresponding to the media-dependent physical address. (netaddress)

`ipNetToMediaType` – The type of mapping. (enum) Returns a fixed value of (3)dynamic. Setting this object to the value invalid(2) has the effect of invalidating the corresponding entry in the ipNetToMediaTable. That is, it effectively dissociates the interface identified with said entry from the mapping identified with said entry. It is an implementation-specific matter as to whether the agent removes an invalidated entry from the table. Accordingly, management stations must be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such entries requires examination of the relevant ipNetToMediaType object.

icmp The `icmp` group reports statistics about the ICMP group.

`icmpInMsgs` – The total number of ICMP messages that the entity received. Note that this counter includes all those counted by icmpInErrors. (counter)

`icmpInErrors` – The number of ICMP messages that the entity received but determined as having ICMP-specific errors (bad ICMP checksums, bad length, and the like.). (counter)

`icmpInDestUnreachs` – The number of ICMP Destination Unreachable messages received. (counter)

`icmpInTimeExcds` – The number of ICMP Time Exceeded messages received. (counter)

`icmpInParmProbs` – The number of ICMP Parameter Problem messages received. (counter)

`icmpInSrcQuenchs` – The number of ICMP Source Quench messages received. (counter)

`icmpInRedirects` – The number of ICMP Redirect messages received. (counter)

`icmpInEchos` – The number of ICMP Echo (request) messages received. (counter)

`icmpInEchoReps` – The number of ICMP Echo Reply messages received. (counter)

`icmpInTimestamps` – The number of ICMP Timestamp (request) messages received. (counter)

`icmpInTimestampReps` – The number of ICMP Timestamp Reply messages received. (counter)

`icmpInAddrMasks` – The number of ICMP Address Mask Request messages received. (counter)

`icmpInAddrMaskReps` – The number of ICMP Address Mask Reply messages received. (counter)

`icmpOutMsgs` – The total number of ICMP messages that this entity attempted to send. Note that this counter includes all those counted by icmpOutErrors. (counter)

`icmpOutErrors` – The number of ICMP messages that this entity did not send due to problems discovered within ICMP, such as a lack of buffers. This value should not include errors discovered outside the ICMP layer, such as the inability of IP to route the resultant datagram. In some implementations there may be no types of errors that contribute to this counter's value. (counter)

`icmpOutDestUnreachs` – The number of ICMP Destination Unreachable messages sent. (counter)

`icmpOutTimeExcds` – The number of ICMP Time Exceeded messages sent. (counter)

`icmpOutParmProbs` – The number of ICMP Parameter Problem messages sent. (counter)

`icmpOutSrcQuenchs` – The number of ICMP Source Quench messages sent. (counter)

`icmpOutRedirects` – The number of ICMP Redirect messages sent. For a host, this object will always be zero, since hosts do not send redirects. (counter)

`icmpOutEchos` – The number of ICMP Echo (request) messages sent. (counter)

`icmpOutEchoReps` – The number of ICMP Echo Reply messages sent. (counter)

`icmpOutTimestamps` – The number of ICMP Timestamp (request) messages sent. (counter)

`icmpOutTimestampReps` – The number of ICMP Timestamp Reply messages sent. (counter)

`icmpOutAddrMasks` – The number of ICMP Address Mask Request messages sent. (counter)

`icmpOutAddrMaskReps` – The number of ICMP Address Mask Reply messages sent. (counter)

tcp The `tcp` group reports statistics about the TCP group.

`tcpRtoAlgorithm` – The algorithm used to determine the timeout value used for retransmitting unacknowledged octets. (enum)

`tcpRtoMin` – The minimum value permitted by a TCP implementation for the retransmission timeout, measured in milliseconds. More refined semantics for objects of this type depend upon the algorithm used to determine the retransmission timeout. In particular, when the timeout algorithm is rsre(3), an object of this type has the semantics of the LBOUND quantity described in RFC 793. (int)

`tcpRtoMax` – The maximum value permitted by a TCP implementation for the retransmission timeout, measured in milliseconds. More refined semantics for objects of this type depend upon the algorithm used to determine the retransmission timeout. In particular, when the timeout algorithm is rsre(3), an object of this type has the semantics of the UBOUND quantity described in RFC 793. (int)

`tcpMaxConn` – The limit on the total number of TCP connections that the entity can support. In entities where the maximum number of connections is dynamic, this object should contain the value –1. (int)

`tcpActiveOpens` – The number of times that TCP connections have made a direct transition to the SYN-SENT state from the CLOSED state. (counter)

`tcpPassiveOpens` – The number of times that TCP connections have made a direct transition to the SYN-RCVD state from the LISTEN state. (counter)

`tcpAttemptFails` – The number of times that TCP connections have made a direct transition to the CLOSED state from either the SYN-SENT state or the SYN-RCVD state, plus the number of times TCP connections have made a direct transition to the LISTEN state from the SYN-RCVD state. (counter)

`tcpEstabResets` – The number of times TCP connections have made a direct transition to the CLOSED state from either the ESTABLISHED state or the CLOSE-WAIT state. (counter)

`tcpCurrEstab` – The number of TCP connections for which the current state is either ESTABLISHED or CLOSE-WAIT. (gauge)

`tcpInSegs` – The total number of segments received, including those received in error. This count includes segments received on currently established connections. (counter)

`tcpOutSegs` – The total number of segments sent, including those on current connections but excluding those containing only retransmitted octets. (counter)

`tcpRetransSegs` – The total number of segments retransmitted - that is, the number of TCP segments transmitted containing one or more previously transmitted octets. (counter)

`tcpInErrs` – The total number of segments received in error (for example, bad TCP checksums). (counter)

`tcpOutRsts` – The number of TCP segments sent containing the RST flag. (counter)

tcpConnTable The `tcpConnTable` is a table containing TCP connection-specific information.

`tcpConnState` – The state of this TCP connection. (enum)

The only value that may be set by a management station is deleteTCB(12). Accordingly, it is appropriate for an agent to return a "badValue" response if a management station attempts to set this object to any other value.

If a management station sets this object to the value deleteTCB(12), then this has the effect of deleting the TCB (as defined in RFC 793) of the corresponding connection on the managed node. This results in immediate termination of the connection.

As an implementation-specific option, an RST segment may be sent from the managed node to the other TCP endpoint. (Note, however, that RST segments are not sent reliably.)

`tcpConnLocalAddress` – The local IP address for this TCP connection. For a connection in the listen state that is willing to accept connections for any IP interface associated with the node, the value 0.0.0.0 is used. (netaddress)

`tcpConnLocalPort` – The local port number for this TCP connection. (int)

`tcpConnRemAddress` – The remote IP address for this TCP connection. (netaddress)

`tcpConnRemPort` – The remote port number for this TCP connection. (int)

upd The `udp` group reports statistics about the UDP group.

`udpInDatagrams` – The total number of UDP datagrams delivered to UDP users. (counter) Returns a fixed value of 0.

`udpNoPorts` – The total number of received UDP datagrams for which there was no application at the destination port. (counter) Returns a fixed value of 0.

`udpInErrors` – The number of received UDP datagrams that could not be delivered for reasons other than the lack of an application at the destination port. (counter)

`udpOutDatagrams` – The total number of UDP datagrams sent from this entity. (counter) Returns a fixed value of 0.

udpTable The `udpTable` is a table containing UDP listener information.

`udpLocalAddress` – The local IP address for this UDP listener. For a UDP listener that is willing to accept datagrams for any IP interface associated with the node, the value 0.0.0.0 is used. (netaddress)

`udpLocalPort` – The local port number for this UDP listener. (int)

snmp The `snmp` group reports statistics about the SNMP group.

`snmpInPkts` – The total number of Messages delivered to the SNMP entity from the transport service. (counter)

`snmpOutPkts` – The total number of SNMP Messages passed from the SNMP protocol entity to the transport service. (counter)

`snmpInBadVersions` – The total number of SNMP Messages delivered to the SNMP protocol entity that were for an unsupported SNMP version. (counter)

`snmpInBadCommunityNames` – The total number of SNMP Messages delivered to the SNMP protocol entity that used a SNMP community name not known to said entity. (counter)

`snmpInBadCommunityUses` – The total number of SNMP Messages delivered to the SNMP protocol entity, which represented an SNMP operation not allowed by the SNMP community named in the Message. (counter)

`snmpInASNParseErrs` – The total number of ASN.1 or BER errors encountered by the SNMP protocol entity when decoding received SNMP Messages. (counter)

`snmpInTooBigs` – The total number of SNMP PDUs delivered to the SNMP protocol entity for which the value of the error-status field is "tooBig." (counter)

`snmpInNoSuchNames` – The total number of SNMP PDUs delivered to the SNMP protocol entity for which the value of the error-status field is "noSuchName." (counter)

`snmpInBadValues` – The total number of SNMP PDUs delivered to the SNMP protocol entity for which the value of the error-status field is "badValue." (counter)

`snmpInReadOnlys` – The total number valid SNMP PDUs delivered to the SNMP protocol entity for which the value of the error-status field is "readOnly." It should be noted that it is a protocol error to generate an SNMP PDU that contains the value "readOnly" in the error-status field. This object is provided as a means of detecting incorrect implementations of the SNMP. (counter)

`snmpInGenErrs` – The total number of SNMP PDUs delivered to the SNMP protocol entity for which the value of the error-status field is "genErr." (counter)

`snmpInTotalReqVars` – The total number of MIB objects successfully retrieved by the SNMP protocol entity as the result of receiving valid SNMP Get-Request and Get-Next PDUs. (counter)

`snmpInTotalSetVars` – The total number of MIB objects successfully altered by the SNMP protocol entity as the result of receiving valid SNMP Set-Request PDUs. (counter)

`snmpInGetRequests` – The total number of SNMP Get-Request PDUs accepted and processed by the SNMP protocol entity. (counter)

`snmpInGetNexts` – The total number of SNMP Get-Next PDUs accepted and processed by the SNMP protocol entity. (counter)

`snmpInSetRequests` – The total number of SNMP Set-Request PDUs accepted and processed by the SNMP protocol entity. (counter)

`snmpInGetResponses` – The total number of SNMP Get-Response PDUs accepted and processed by the SNMP protocol entity. (counter)

`snmpInTraps` – The total number of SNMP Trap PDUs accepted and processed by the SNMP protocol entity. (counter)

`snmpOutTooBigs` – The total number of SNMP PDUs generated by the SNMP protocol entity for which the value of the error-status field is "tooBig." (counter)

`snmpOutNoSuchNames` – The total number of SNMP PDUs generated by the SNMP protocol entity for which the value of the error-status is "noSuchName." (counter)

`snmpOutBadValues` – The total number of SNMP PDUs generated by the SNMP protocol entity for which the value of the error-status field is "badValue." (counter)

`snmpOutGenErrs` – The total number of SNMP PDUs generated by the SNMP protocol entity for which the value of the error-status field is "genErr." (counter)

`snmpOutGetRequests` – The total number of SNMP Get-Request PDUs which have been generated by the SNMP protocol entity. (counter)

`snmpOutGetNexts` – The total number of SNMP Get-Next PDUs generated by the SNMP protocol entity. (counter)

`snmpOutSetRequests` – The total number of SNMP Set-Request PDUs generated by the SNMP protocol entity. (counter)

`snmpOutGetResponses` – The total number of SNMP Get-Response PDUs generated by the SNMP protocol entity. (counter)

`snmpOutTraps` – The total number of SNMP Trap PDUs generated by the SNMP protocol entity. (counter)

`snmpEnableAuthenTraps` – Indicates whether the SNMP agent process is permitted to generate authentication-failure traps. The value of this object overrides any configuration information. As such, it provides a means whereby all authentication-failure traps may be disabled. (enum)

Note that this object must be stored in non-volatile memory, so that it remains constant between reinitializations of the network management system.

The following are Sun-specific group and table definitions.

sunSystem The `sunSystem` group reports general system information.

`agentDescr` – The SNMP agent's description of itself. (string[255])

`hostID` – The unique Sun hardware identifier. The value returned is four byte binary string. (octet[4])

`motd` – The first line of `/etc/motd`. (string[255])

`unixTime` – The UNIX system time. Measured in seconds since January 1, 1970 GMT. (counter)

sunProcessTable The `sunProcessTable` table reports UNIX process table information.

`psProcessID` – The process identifier for this process. (int)

`psParentProcessID` – The process identifier of this process's parent. (int)

`psProcessSize` – The combined size of the data and stack segments (in kilobytes.) (int)

`psProcessCpuTime` – The CPU time (including both user and system time) consumed so far. (int)

`psProcessState` – The run-state of the process. (octet[4])

R	Runnable
T	Stopped
P	In page wait
D	Non-interruptable wait
S	Sleeping (less than 20 seconds)
I	Idle (more than 20 seconds)
Z	Zombie

`psProcessWaitChannel` – Reason process is waiting. (octet[16])

`psProcessTTY` – Terminal, if any, controlling this process. (octet[16])

`psProcessUserName` – Name of the user associated with this process. (octet[16])

`psProcessUserID` – Numeric form of the name of the user associated with this process. (int)

`psProcessName` – Command name used to invoke this process. (octet[64])

`psProcessStatus` – Setting this variable will cause a signal of the set value to be sent to the process. (int)

sunHostPerf The `sunHostPerf` group reports hostperf information.

`rsUserProcessTime` – Total number of timeticks used by user processes since the last system boot. (counter)

`rsNiceModeTime` – Total number of timeticks used by "nice" mode since the last system boot. (counter)

`rsSystemProcessTime` – Total number of timeticks used by system processes since the last system boot. (counter)

`rsIdleModeTime` – Total number of timeticks in idle mode since the last system boot. (counter)

`rsDiskXfer1` – Total number of disk transfers since the last boot for the first of four configured disks. (counter)

`rsDiskXfer2` – Total number of disk transfers since the last boot for the second of four configured disks. (counter)

`rsDiskXfer3` – Total number of disk transfers since the last boot for the third of four configured disks. (counter)

`rsDiskXfer4` – Total number of disk transfers since the last boot for the fourth of four configured disks. (counter)

`rsVPagesIn` – Number of pages read in from disk. (counter)

`rsVPagesOut` – Number of pages written to disk. (counter)

`rsVSwapIn` – Number of pages swapped in. (counter)

`rsVSwapOut` – Number of pages swapped out. (counter)

`rsVIntr` – Number of device interrupts. (counter)

`rsIfInPackets` – Number of input packets. (counter)

`rsIfOutPackets` – Number of output packets. (counter)

`rsIfInErrors` – Number of input errors. (counter)

`rsIfOutErrors` – Number of output errors. (counter)

`rsIfCollisions` – Number of output collisions. (counter)

FILES

`/etc/snmp/conf/snmpd.conf`	configuration information
`/var/snmp/mib/sun.mib`	standard SNMP MIBII file

ATTRIBUTES See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmibii
Interface Stability	Obsolete

SEE ALSO

`inetd`(1M), `select`(3C), `recvfrom`(3SOCKET), `sendto`(3SOCKET), `attributes`(5), `gld`(7D),

DIAGNOSTICS

`cannot dispatch request`
The proxy cannot dispatch the request. The rest of the message indicates the cause of the failure.

`select(3C) failed`
A `select`(3C) call failed. The rest of the message indicates the cause of the failure.

`sendto(3SOCKET) failed`
A `sendto`(3SOCKET) call failed. The rest of the message indicates the cause of the failure.

`recvfrom(3SOCKET) failed`
A `recvfrom`(3SOCKET) call failed. The rest of the message indicates the cause of the failure.

`no response from system`
The SNMP agent on the target system does not respond to SNMP requests. This error might indicate that the SNMP agent is not running on the target system, the target system is down, or the network containing the target system is unreachable.

`response too big`
The agent could not fit the results of an operation into a single SNMP message. Split large groups or tables into smaller entities.

`missing attribute`
An attribute is missing from the requested group.

`bad attribute type`
An object attribute type received from the SNMP agent that does not match the attribute type specified by the proxy agent schema. The rest of the message indicates the expected type and received type.

`cannot get sysUpTime`
The proxy agent cannot get the variable *sysUpTime* from the SNMP agent.

`sysUpTime type bad`
The variable *sysUpTime* received from the SNMP agent has the wrong data type.

`unknown SNMP error`
An unknown SNMP error was received.

`bad variable value`
The requested specified an incorrect syntax or value for a set operation.

`variable is read only`
The SNMP agent did not perform the set request because a variable to set may not be written.

`general error`
A general error was received.

`cannot make request PDU`
An error occurred building a request PDU.

`cannot make request varbind list`
An error occurred building a request variable binding list.

`cannot parse response PDU`
An error occurred parsing a response PDU.

`request ID - response ID mismatch`
The response ID does not match the request ID.

`string contains non-displayable characters`
A displayable string contains non-displayable characters.

`cannot open schema file`
An error occurred opening the proxy agent schema file.

`cannot parse schema file`
The proxy agent couldn't parse the proxy agent schema file.

`cannot open host file`
An error occurred opening the file associated with the *na.snmp.hostfile* keyword in `/etc/snmp/conf/snmpd.conf`

`cannot parse host file`
The proxy agent was unable to parse the file associated with the *na.snmp.hostfile* keyword in `/etc/snmp/conf/snm.conf`.

`attribute unavailable for set operations`
The set could not be completed because the attribute was not available for set operations.

BUGS

The `mibiisa` utility returns the wrong interface speed for the SBUS FDDI interface (for example, "bf0").

The `mibiisa` utility does not return a MAC address for the SBUS FDDI interface (for example, "bf0").

Process names retrieved from `mibiisa` contain a leading blank space.

When you change attribute values in the system group with an SNMP set request, the change is effective only as long as `mibiisa` is running. `mibiisa` does not save the changes to `/etc/snmp/conf/snmpd.conf`.

NAME

mipagent – Mobile IP agent

SYNOPSIS

/usr/lib/inet/mipagent

DESCRIPTION

The `mipagent` utility implements the Mobile IP home agent and foreign agent functionality described in *RFC 2002, IP Mobility Support*. The term "mobility agent" is used to refer to the home agent and foreign agent functionality collectively. `mipagent` responds to Mobile IP registration and deregistration requests and router discovery solicitation messages from a mobile node. Besides responding to external messages, the `mipagent` utility also tasks on a periodic basis, such as aging the mobility bindings and visitor entries and sending agent advertisements. The mobility agent can also handle direct delivery style reverse tunneling as specified in *RFC 2344, Reverse Tunneling for Mobile IP*. Limited private address support for mobile nodes is also available. In addition, separate IPsec policies for registration requests, replies, and tunnel traffic can be configured to protect the datagrams associated with these between two mobility agents.

Run the `mipagent` daemon as root using the start-up script, which has the following syntax:

```
example# /etc/init.d/mipagent [start|stop]
```

`/etc/inet/mipagent.conf` must be present before you start-up the `mipagent` daemon. See `mipagent.conf`(4). At start up, `mipagent` reads the configuration information from `/etc/inet/mipagent.conf`. The `mipagent` daemon records a continuous log of its activities by means of `syslog()`. See `syslog`(3C). You can use the *LogVerbosity* parameter in `/etc/inet/mipagent.conf` to control the verbosity level of the log.

The `mipagent` daemon can be terminated either by the script:

```
example# /etc/init.d/mipagent stop
```

or by the `kill` command.

Periodically while running, or if terminated or shutdown, the `mipagent` daemon stores the following internal state information in `/var/inet/mipagent_state`:

- a list of the mobile nodes supported as home agents;
- their current care-of addresses; and
- the remaining registration lifetimes.

If the `mipagent` utility is terminated for maintenance and restarted, `mipagent_state` is used to recreate as much of the mobility agent's internal state as possible. This minimizes service disruption for mobile nodes that may be visiting other networks. If `mipagent_state` exists, it is read immediately after `mipagent.conf` when `mipagent` is restarted. The format of `mipagent_state` is undocumented since it is likely to change and programs other than `mipagent` should not use it for any purpose. A separate utility program `mipagentstat` is provided for monitoring `mipagent`.

EXIT STATUS

The following exit values are returned:

`0` The daemon started successfully.

`1` The daemon failed to start.

FILES

`/etc/inet/mipagent.conf`
Configuration file for Mobile IP mobility agent.

`/var/inet/mipagent_state`
File where private state information from `mipagent` is stored.

`/etc/init.d/mipagent [start|stop]`
`mipagent` start-up script.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmipu

SEE ALSO

`mipagentstat`(1M), `mipagentconfig`(1M), `syslog`(3C), `mipagent.conf`(4), `attributes`(5)

Montenegro, G., editor.*RFC 2344, Reverse Tunneling for Mobile IP*. Network Working Group. May 1998.

Perkins, C. *RFC 2002, IP Mobility Support*. Network Working Group. October 1996.

DIAGNOSTICS

The `mipagent` utility exits with an error if the configuration file, `mipagent.conf`, cannot be read successfully. Upon receiving a `SIGTERM` or `SIGINT` signal, `mipagent` cleans its internal state, including any changes to the routing and ARP tables, and exits.

NOTES

The foreign agent adds host– specific local routes to its routing table for visiting mobile nodes after they are successfully registered. If a visiting mobile node departs without sending a de-registration message through the foreign agent, these routing entries persist until the mobile node's previous registration expires. Any packets that arrive at the foreign agent for the departed mobile node during this time, for example because the foreign agent is also a router for the foreign network, will be lost. System administrators can configure foreign agents to accept only short registration lifetimes. This will automatically restrict the maximum duration for which a departed mobile node will be temporarily unreachable.

Home and foreign agents dynamically add and delete IPsec policies configured with a mobility agent peer. Those pertaining to the tunnel are only added when the tunnel is plumbed. At this time, IPsec tunnel policies must be identical in the forward and reverse direction. IPsec policies pertaining to permiting registration requests on the home agent are added to the IPsec policy file at init time as it must be ready to receive

these at any time. Otherwise, IPsec policies pertaining to registration request and reply messages with a mobility agent peer are added as soon as they are needed, and are not removed until all mobile nodes are no longer registered with the mobility agent peer, at which point the tunnels are torn down.

NAME

mipagentconfig – configure Mobility IP Agent

SYNOPSIS

/sbin/mipagentconfig [-f *configfile*] *command dest* [*parameters*...]

DESCRIPTION

The `mipagentconfig` utility is used to configure the Mobility IP Agent. It allows the user to change settings and to add and delete mobility clients, Pools, and SPIs in the mobility agent configuration file.

OPTIONS

The following options are supported:

`-f` *configfile* — Use the specified configuration file instead of the system default, `/etc/inet/mipagent.conf`.

OPERANDS

The *command* operand, as well as the parameters for each command are described below. See `mipagent.conf`(4) for the default values of the configuration operands described here.

`add` — Depending on the destination *dest*, this command will add advertisement parameters, security parameters, SPIs, or addresses to the configuration file.

`add Address` *ipAddress attr_value*
Add the specified *ipAddress* with the specified `SPI`. To add an NAI address, you must specify the `Pool`.

`add adv` *device*
Enable home and foreign agent functionality on the specified interface.

`add adv` *device* `AdvLifetime` *seconds*
Add `AdvLifetime` to the specified device.

`add adv` *device* `RegLifetime` *seconds*
Add `RegLifetime` to the specified device.

`add adv` *device* `AdvFrequency` *seconds*
Add `AdvFrequency` to the specified device.

`add adv` *device* `AdvInitCount` *count*
Add initial unsolicited advertisement count. *count* should be a small integer.

`add adv` *device* `AdvLimitUnsolicited yes | no`
Enable limited or unlimited unsolicited advertisements for foreign agent. Accepted values are:

`yes` — Limit unsolicited advertisement to `AdvInitCount` initial advertisements.

`no` — Do not limit unsolicited advertisement. The advertisement should take place periodically at the frquency specified by `AdvFrequency`.

`add adv` *device* `HomeAgent yes | no`
Add the `HomeAgent` flag to the specified device.

`add adv` *device* `ForeignAgent yes | no`
Add the `ForeignAgent` flag to the specified device.

`add adv` *device* `PrefixLengthExt yes | no`
Add the `PrefixLengthExt` flag to the specified device.

`add adv` *device* `NAIExt yes | no`
Add the `NAIExt` flag to the specified device.

`add adv` *device* `Challenge yes | no`
Add the `Challenge` flag to the specified device.

`add adv` *device* `ReverseTunnel no | neither fa ha yes | both`
Add the level of `ReverseTunnel` support indicated to the specified device. Possible values include:

`no`	Do not support `ReverseTunnel` as either a foreign agent or a home agent on this device. Does not advertise reverse tunneling nor accept a registration requesting reverse tunnel support on this device.
`neither`	Do not support `ReverseTunnel` as either a foreign agent or a home agent on this device. Do not advertise reverse tunneling nor accept a registration requesting reverse tunnel support on this device.
`fa`	When the foreign agent processes a registration request received on this device, check to see if the mobile node requests that a reverse tunnel be set up to its home agent. If so, perform the necessary encapsulation of datagrams to the mobile node's home agent as described in *RFC 2344*. This means that a mobile node must see the agent advertising reverse tunnel support, so the reverse tunnel bit is advertised in the agent advertisement on this device.
`ha`	When the home agent processes a registration request received on this device, check to see if the mobile node requests that a reverse tunnel be set up from its care-of address. If so, perform the necessary decapsulation as described in *RFC 2344*. This does not mean the home agent is advertising support of reverse tunneling on

this device. Mobile nodes are only interested in the advertisement flags if they are going to use foreign agent services. Moreover, reverse tunnels by definition originate at the care-of address, and HA support is therefore only of interest to the owner of the care-of address.

`yes` Whenever the mobility agent is processing a registration request received on this device, check to see if the mobile node is requesting a reverse tunnel be set up. If so, apply *RFC 2344* as appropriate, either as an encapsulating foreign agent, or a decapsulating home agent, depending on how this mobility agent is servicing the specific mobile node. As a result, the mobility agent will be advertising reverse tunnel support on this device.

`both` Whenever the mobility agent is processing a registration request received on this device, check to see if the mobile node is requesting a reverse tunnel be set up. If so, apply *RFC 2344* as appropriate, either as an encapsulating foreign agent, or a decapsulating home agent, depending on how this mobility agent is servicing the specific mobile node. As a result, the mobility agent will be advertising reverse tunnel support on this device.

`add adv` *device* `ReverseTunnelRequired no | neither fa ha yes | both yes | both`

Add the requirement that the `ReverseTunnel` flag be set in any registration request received on the indicated device. Possible values include:

`no` Reverse tunneling is not required by the `mipagent` on this device.

`neither` Reverse tunneling is not required by the `mipagent` on this device.

`fa` The `ReverseTunnel` flag is required to be set in registration requests received by the foreign agent on this device.

	`ha`	The `ReverseTunnel` flag is required to be set in registration requests received by the home agent on this device.
	`yes`	The `ReverseTunnel` flag is required to be set in all registration requests received by either home and or foreign agents on this device.
	`both`	The `ReverseTunnel` flag is required to be set in all registration requests received by either home and or foreign agents on this device.

`add Pool` *number startAddr length*
Add the specified `Pool` with the specified start addresses and length.

`add SPI` *number replay Key*
Add the specified `SPI` with the given replay type and key. The replay type can have a value of `none` or `timestamps`.

`add HA-FAAuth yes | no`
Add the `HA-FAAuth` flag.

`add MN-FAAuth yes | no`
Add the `MN-FAAuth` flag.

`add MaxClockSkew` *seconds*
Add the `MaxClockSkew`.

`add KeyDistribution` *type*
Add the `KeyDistribution` type. The only value for `KeyDistribution` that is supported at this time is `file`.

`change`	Depending on the destination *dest*, this command will change advertisement parameters, security parameters, SPIs, or addresses in the configuration file. Any of the above destinations are valid.
`delete`	Depending on the destination *dest*, this command will delete advertisement parameters, security parameters, SPIs, or addresses from the configuration file. Any destination discussed above is valid.
`get`	Display all of the parameters associated with *dest*. Any destination discussed above is valid.

EXAMPLES

EXAMPLE 1 Adding an SPI, a Pool, and a Mobile Node and Requiring Reverse Tunneling on a Device to the *configfile*

The following example adds an SPI, a `Pool`, a `mobile node`, and requires reverse tunneling for the foreign agent in the *configfile*. First, the SPI of 250 is added. Then, a Pool of 200 addresses starting at `192.168.168.1` is added. `joe@mobile.com` is added with an SPI of 250 and using Pool 1. Finally, reverse tunneling is required for the foreign agent on device `le0`.

```
example# mipagentconfig add SPI 250 ReplayMethod none
example# mipagentconfig add SPI 250 Key 00ff00ff00ff
example# mipagentconfig add Pool 1 192.168.168.1 200
example# mipagentconfig add Address joe@mobile.com 250 1
example# mipagentconfig add le0 reversetunnel fa
example# mipagentconfig add le0 reversetunnelrequired fa
```

EXAMPLE 2 Adding Dynamic Interface Mobility Support on PPP Interfaces

The following example adds dynamic interface mobility support on PPP interfaces. The backslash (\) escape character is used to type in device number `*`. The example also indicates that all the new ppp interfaces offer reverse tunnel service.

```
example# mipagentconfig add ppp\* reversetunnel yes
example# mipagentconfig add ppp\* AdvLimitUnsolicited yes
example# mipagentconfig add ppp\* AdvInitCount 3
example# mipagentconfig add ppp\* AdvFrequency 1
```

EXAMPLE 3 Adding IPsec Policies to an Agent-Peer Entry

The following example adds IPsec policies to an existing mobility agent entry, then displays the configuration for the mobility agent peer. The backslash (\) character denotes a line continuation for the formatting of this example.

```
example# mipagentconfig add Address 192.168.10.1 \
           IPsecRequest apply {auth_algs md5 sa shared}
example# mipagentconfig add Address 192.168.10.1 \
           IPsecReply permit {auth_algs md5}
example# mipagentconfig add Address 192.168.10.1 \
           IPsecTunnel permit {encr_auth_algs md5 encr_algs 3des}
example# mipagentconfig get Address 192.168.10.1

[Address 192.168.10.1]
     Type = agent
     SPI = 137
     IPsecRequest = apply {auth_algs md5 sa shared}
     IPsecReply = permit {auth_algs md5}
     IPsecTunnel = \
     permit {encr_auth_algs md5 encr_algs 3des}
```

EXAMPLE 4 Modifying an SPI

To modify the SPI associated with joe, first, use the command `get` to verify the existing settings, then change the SPI from 250 to 257.

```
example# mipagentconfig get Address joe@mobile.com
Address: joe@mobile.com
SPI:  250
Pool: 1
example# mipagentconfig change Address joe@mobile.com 257 1
```

EXAMPLE 5 Deleting a Pool

Use the following example to delete Pool `3`:

```
example# mipagentconfig delete Pool 3
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`non-zero` An error occurred.

FILES

`/etc/inet/mipagent.conf`
Configuration file for Mobile IP mobility agent.

`/etc/inet/mipagent.conf-sample`
Sample configuration file for mobility agents.

`/etc/inet/mipagent.conf.ha-sample`
Sample configuration file for home agent functionality.

`/etc/inet/mipagent.conf.fa-sample`
Sample configuration file for foreign agent functionality.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmipu

SEE ALSO

`mipagent`(1M), `mipagent.conf`(4), `attributes`(5)

Montenegro, G., editor. *RFC 2334, Reverse Tunneling for Mobile IP*. Network Working Group. May, 1998.

Perkins, C. *RFC 2002, IP Mobility Support*. Network Working Group. October 1996.

NAME

mipagentstat – show Mobile IP Mobility Agent status

SYNOPSIS

mipagentstat [-fhp]

DESCRIPTION

Use the `mipagentstat` utility to display the content of various Mobile-IP related data structures.

Visitor Table (First Form)

The visitor table display lists information for all mobile nodes registered with the foreign agent, one mobile node per line. This list consists of the mobile node's home address or Network Access Identifier (NAI), home agent address, total registration lifetime and the number of seconds remaining before the registration expires.

The following command line shows the output from a foreign agent with two mobile nodes registered:

```
example# mipagentstat -f
Mobile Node     Foreign Agent  Time Granted    Time Remaining     Flags
                               (in secs)        (in secs)
foobar@xyz.com  fa1@tuv.com     600             125
10.1.5.23       123.2.5.12     1000            10                 R
```

An "R" in the flags column indicates a reverse tunnel is present. No reverse tunnel is configured for the mobile node `foobar@xyz.com`. A reverse tunnel is configured from mobile node `10.1.5.23`.

Binding Table (Second Form)

The binding table display lists information for all mobile nodes registered with the home agent, one mobile node per line. This list consists of the mobile node's home address or NAI, foreign agent address, total registration lifetime and the number of seconds remaining before the registration expires.

Use the following command line to show the output from a home agent with two active mobile nodes:

```
example# mipagentstat -h
Mobile Node     Home Agent     Time Granted    Time Remaining     Flags
                               (in secs)       (in secs)
foobar@xyz.com  ha1@xyz.com    600             125
10.1.5.23       10.1.5.1       1000            10                 R
```

An "R" in the flags column indicates a reverse tunnel is present. No reverse tunnel is configured for the mobile node `foobar@xyz.com`. A reverse tunnel is configured for mobile node `10.1.5.23`.

Agent Table (Third Form)

The agent table display lists information for all current mobility agent-peers, that is all mobility agents with which mobile-nodes we are servicing are trying to obtain service.

Provided in this display are the IPsec protection mechanisms being used with registration requests, replies, and tunnels.

Use the following command line to show the output from a home agent with two (foreign) mobility agent peers:

```
example# mipagentstat -hp
```

```
Foreign                    ..... Security Association(s).....
Agent                      Requests Replies  FTunnel  RTunnel
------------------------   -------- -------- -------- --------
fa.eng.sun.com             AH,ESP   AH,ESP   AH,ESP   AH,ESP
fa.central.sun.com         AH       AH       ESP      ESP
```

Use the following command line to show the output from a home agent with two (foreign) mobility agent peers:

```
example# mipagentstat -fp
Home                       ..... Security Association(s) .....
Agent                      Requests Replies  FTunnel  RTunnel
------------------------   -------- -------- -------- --------
ha.eng.sun.com             AH,ESP   AH,ESP   AH,ESP   AH,ESP
ha.central.sun.com
```

Use of the `-p` option without specifying the agent results in both displays described above, that is one display for each agent.

An `AH` in any column indicates the IPsec `AH` mechanism is in place for those datagrams.

An `ESP` in any column indicates the IPsec `ESP` mechanism is in place for those datagrams.

OPTIONS

The following options are supported:

`-f` Display the list of active mobile nodes in the foreign agent's visitor's list.

`-h` Display the list of active mobile nodes in the home agent's binding table.

`-p` Display the list of mobility agent peers, and the IPsec protection mechanisms currently in use for registration and tunnel traffic.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`non-zero` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmipu

SEE ALSO

`mipagent`(1M), `mipagentconfig`(1M), `mipagent.conf`(4), `attributes`(5)

Aboda, B., and Beadles, M. *RFC 2486, The Network Access Identifier*. The Internet Society, 1999.

NAME

mkdevalloc – Make device_allocate entries

SYNOPSIS

/usr/sbin/mkdevalloc

DESCRIPTION

The `mkdevalloc` command writes to standard out a set of `device_allocate`(4) entries describing the system's frame buffer, audio and removable media devices.

The `mkdevalloc` command is used by the `init.d`(4) scripts to create or update the `/etc/security/device_allocate` file.

Entries are generated based on the device special files found in `/dev`. For the different categories of devices, the `mkdevalloc` command checks for the following files under `/dev`:

```
audio           /dev/audio, /dev/audioctl, /dev/sound/...
tape            /dev/rst*, /dev/nrst*, /dev/rmt/...
floppy          /dev/diskette, /dev/fd*, /dev/rdiskette, /dev/rfd*
removable disk  /dev/sr*, /dev/nsr*, /dev/dsk/c0t?d0s?, /dev/rdsk/c0t?d0s?
frame buffer    /dev/fb
```

All entries set the *device-minimum* and *device-maximum* fields to the hex representations of `ADMIN_LOW` and `ADMIN_HIGH`, respectively. The *device-authorization* field is set to `solaris.device.allocate`, except for the `framebuffer` entry, where it is set to `*`. The *device-name*, *device-type* and *device-clean* fields are set to the following values:

```
                device-name      device-type   device-clean

audio           audio            audio         audio_clean_wrapper
tape            mag_tape_0,1,... st            st_clean
floppy          floppy_0,1,...   fd            disk_clean
removable disk  cdrom_0,1,...    sr            disk_clean
frame buffer    framebuffer      fb            /bin/true
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`allocate`(1), `bsmconv`(1M), `attributes`(5)

NAME

mkdevmaps – make device_maps entries

SYNOPSIS

/usr/sbin/mkdevmaps

DESCRIPTION

The `mkdevmaps` command writes to standard out a set of `device_maps`(4) entries describing the system's frame buffer, audio, and removable media devices.

The `mkdevmaps` command is used by the `init.d`(4) scripts to create or update the `/etc/security/device_maps` file.

Entries are generated based on the device special files found in `/dev`. For the different categories of devices, the `mkdevmaps` command checks for the following files under `/dev`:

```
audio           /dev/audio, /dev/audioctl, /dev/sound/...
tape            /dev/rst*, /dev/nrst*, /dev/rmt/...
floppy          /dev/diskette, /dev/fd*, /dev/rdiskette, /dev/rfd*
removable disk  /dev/dsk/c0t?d0s?, /dev/rdsk/c0t?d0s?
frame buffer    /dev/fb
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`allocate`(1), `bsmconv`(1M), `attributes`(5)

NAME

mkfifo – make FIFO special file

SYNOPSIS

/usr/bin/mkfifo [-m *mode*] *path*...

DESCRIPTION

The mkfifo command creates the FIFO special files named by its argument list. The arguments are taken sequentially, in the order specified; and each FIFO special file is either created completely or, in the case of an error or signal, not created at all.

If errors are encountered in creating one of the special files, mkfifo writes a diagnostic message to the standard error and continues with the remaining arguments, if any.

The mkfifo command calls the library routine mkfifo(3C), with the *path* argument is passed as the *path* argument from the command line, and *mode* is set to the equivalent of a=rw, modified by the current value of the file mode creation mask umask(1).

OPTIONS

The following option is supported:

-m *mode* Set the file permission bits of the newly-created FIFO to the specified *mode* value. The *mode* option-argument will be the same as the *mode* operand defined for the chmod(1) command. In <*symbolic*mode> strings, the *op* characters + and – will be interpreted relative to an assumed initial mode of a=rw.

OPERANDS

The following operand is supported:

file A path name of the FIFO special file to be created.

USAGE

See largefile(5) for the description of the behavior of mkfifo when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

ENVIRONMENT VARIABLES

See environ(5) for descriptions of the following environment variables that affect the execution of mkfifo: LC_CTYPE, LC_MESSAGES, and NLSPATH.

EXIT STATUS

The following exit values are returned:

0 All the specified FIFO special files were created successfully.

>0 An error occurred.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWesu

SEE ALSO

mkfifo(3C), attributes(5), environ(5), largefile(5)

NAME

mkfile – create a file

SYNOPSIS

mkfile [-nv] *size* [g | k | b | m] *filename...*

DESCRIPTION

mkfile creates one or more files that are suitable for use as NFS-mounted swap areas, or as local swap areas. When a root user executes mkfile(), the sticky bit is set and the file is padded with zeros by default. When non-root users execute mkfile(), they must manually set the sticky bit using chmod(1). The default size is in bytes, but it can be flagged as gigabytes, kilobytes, blocks, or megabytes, with the g, k, b, or m suffixes, respectively.

OPTIONS

-n Create an empty *filename*. The size is noted, but disk blocks are not allocated until data is written to them. Files created with this option cannot be swapped over local UFS mounts.

-v Verbose. Report the names and sizes of created files.

USAGE

See largefile(5) for the description of the behavior of mkfile when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

chmod(1), swap(1M), attributes(5), largefile(5)

NAME

mkfs – construct a file system

SYNOPSIS

mkfs [-F *FSType*] [*generic_options*] [-o *FSType-specific_options*] *raw_device_file* [*operands*]

DESCRIPTION

The mkfs utility constructs a file system on the *raw_device_file* by calling the specific mkfs module indicated by -F *FSType*.

Note: ufs file systems are normally created with the newfs(1M) command.

generic_options are independent of file system type. *FSType-specific_options* is a comma-separated list of *keyword=value* pairs (with no intervening spaces), which are *FSType*-specific. *raw_device_file* specifies the disk partition on which to write the file system. It is required and must be the first argument following the *specific_options* (if any). *operands* are *FSType*-specific. See the *FSType*-specific manual page of mkfs (for example, mkfs_ufs (1M)) for a detailed description.

OPTIONS

The following are the generic options for mkfs:

-F Specify the *FSType* to be constructed. If -F is not specified, the *FSType* is determined from /etc/vfstab by matching the *raw_device_file* with a vfstab entry, or by consulting the /etc/default/fs file.

-V Echo the complete command line, but do not execute the command. The command line is generated by using the options and arguments provided and adding to them information derived from /etc/vfstab or /etc/default/fs. This option may be used to verify and validate the command line.

-m Return the command line which was used to create the file system. The file system must already exist. This option provides a means of determining the command used in constructing the file system.

-o Specify *FSType*-specific options. See the manual page for the mkfs module specific to the file system type.

USAGE

See largefile(5) for the description of the behavior of mkfs when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

FILES

/etc/default/fs Default file system type. Default values can be set for the following flags in /etc/default/fs. For example: LOCAL=ufs

LOCAL The default partition for a command if no *FSType* is specified.

/etc/vfstab List of default parameters for each file system

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`mkfs_ufs`(1M), `newfs`(1M), `vfstab`(4), `attributes`(5), `largefile`(5)

Manual pages for the *FSType*-specific modules of `mkfs`.

NOTES

This command might not be supported for all *FSType*s.

You can use `lofiadm` to create a file that appears to a `mkfs` command as a raw device. You can then use a `mkfs` command to create a file system on that device. See `lofiadm`(1M) for examples of creating a UFS and a PC (FAT) file system (using `mkfs_ufs`(1M) and `mkfs_pcfs`(1M)) on a device created by `lofiadm`.

NAME

mkfs_pcfs – construct a FAT file system

SYNOPSIS

mkfs -F pcfs [*generic_options*] [-o *FSType_specific_options*] *raw_device_file*

DESCRIPTION

The pcfs-specific module of mkfs constructs a File Allocation Table (FAT) on removable media (diskette, JAZ disk, ZIP disk, PCMCIA card), a hard disk, or a file (see NOTES). FATs are the standard MS-DOS and Windows file system format. Note that you can use fdformat(1) to construct a FAT file system only on a diskette or PCMCIA card.

mkfs for pcfs determines an appropriate FAT size for the medium, then it installs an initial boot sector and an empty FAT. A sector size of 512 bytes is used. mkfs for pcfs can also install the initial file in the file system (see the pcfs-specific -o i option). This first file can optionally be marked as read-only, system, and/or hidden.

If you want to construct a FAT with mkfs for pcfs on a medium that is not formatted, you must first perform a low-level format on the medium with fdformat(1) or format(1M). Non-diskette media must also be partitioned with the fdisk(1M) utility. Note that all existing data on the diskette or disk partition, if any, is destroyed when a new FAT is constructed.

generic_options are supported by the generic mkfs command. See mkfs(1M) for a description of these options.

raw_device_file indicates the device on which to write unless the -o N option has been specified, or if the -V or -m generic options are passed from the generic mkfs module.

OPTIONS

See mkfs(1M) for the list of supported generic options.

The following options are supported:

-o *FSType_specific_options*
Specify pcfs file system-specific options in a comma-separated list with no intervening spaces. If invalid options are specified, a warning message is printed and the invalid options are ignored.

b=*label*	Label the media with volume label. The volume label is restricted to 11 uppercase characters.
B=*filename*	Install *filename* as the boot loader in the file system's boot sector. If you don't specify a boot loader, an MS-DOS boot loader is installed. The MS-DOS boot loader requires specific MS-DOS system files to make the diskette bootable. See NOTES for more information.
fat=*n*	The size of a FAT entry. Currently, only 12 and 16 are valid values. The default is 12 for diskettes, 16 for larger media.
h	Mark the first file installed as a hidden file. The -i option must also be specified.

hidden=*n*	Set the number of hidden sectors to *n*. This is the number of sectors on the physical disk preceding the start of the volume (which is the boot sector itself). This defaults to 0 for diskettes or a computed valued (based on the fdisk table) for disks. This option may be used only in conjunction with the `nofdisk` option.
i=*filename*	Install *filename* as the initial file in the new file system. The initial file's contents are guaranteed to occupy consecutive clusters at the start of the files area. When creating bootable media, a boot program should be specified as the initial file.
nofdisk	Do not attempt to find an `fdisk` table on the medium. Instead rely on the `size` option for determining the partition size. By default, the created FAT is 16 bits and begins at the first sector of the device. This origination sector can be modified with the hidden option (`-h`).
nsect=*n*	The number of sectors per track on the disk. If not specified, the value is determined by using a `dkio`(7I) ioctl to get the disk geometry, or (for diskette) from the results of an `FDIOGCHAR` ioctl.
ntrack=*n*	The number of tracks per cylinder on the disk. If not specified, the value is determined by using a `dkio`(7I) ioctl to get the disk geometry, or (for diskette) from the results of an `FDIOGCHAR` ioctl.
N	No execution mode. Print normal output, but do not actually write the file system to the medium. This is most useful when used in conjunction with the verbose option.
r	Mark the first file installed as read-only. The `-i` option must also be specified.
reserve=*n*	Set the number of reserved sectors to *n*. This is the number of sectors in the volume, preceding the start of the first FAT, including the boot sector. The value should always be at least 1, and the default value is exactly 1.
s	Mark the first file installed as a system file. The `-i` option must also be specified.
size=*n*	The number of sectors in the file system. If not specified, the value is determined from the size of the partition given in the fdisk table or (for diskette) by way of computation using the `FDIOGCHAR` ioctl.
spc=*n*	The size of the allocation unit for space within the file system, expressed as a number of sectors. The default value depends on the FAT entry size and the size of the file system.

v — Verbose output. Describe, in detail, operations being performed.

FILES

raw_device_file — The device on which to build the FAT. The device name for a diskette must be specified as `/dev/rdiskette0` for the first diskette drive, or `/dev/rdiskette1` for a second diskette drive. For non-diskette media, a disk device name must be qualified with a suffix to indicate the proper partition. For example, in the name `/dev/rdsk/c0t0d0p0:c`, the `:c` suffix indicates that the first partition on the disk should receive the new FAT.

For a file, *raw_device_file* is the block device name returned by `lofiadm`(1M).

EXAMPLES

The media in these examples must be formatted before running `mkfs` for `pcfs`. See `DESCRIPTION` for more details.

EXAMPLE 1 Creating a FAT File System on a Diskette

The following command creates a FAT file system on a diskette:

```
mkfs -F pcfs /dev/rdiskette
```

EXAMPLE 2 Creating a FAT File System on a Disk

The following command creates a FAT file system on the second fdisk partition of a disk attached to an IA based system:

```
mkfs -F pcfs /dev/rdsk/c0d0p0:d
```

EXAMPLE 3 Creating a FAT File System on a ZIP Disk

The following command creates a FAT file system on a ZIP disk located on a SPARC based system:

```
mkfs -F pcfs /dev/rdsk/c0t4d0s2:c
```

EXAMPLE 4 Creating a FAT File System on a JAZ Disk

The following command creates a FAT file system on a JAZ disk located on a SPARC based system and overrides the sectors/track and tracks/cylinder values obtained from the device's controller:

```
mkfs -F pcfs -o nsect=32,ntrack=64 /dev/rdsk/c0t3d0s2:c
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWesu
Interface Stability	Stable

SEE ALSO `fdformat`(1), `fdisk`(1M), `format`(1M), `lofiadm`(1M), `mkfs`(1M), `attributes`(5), `fd`(7D), `dkio`(7I), `fdio`(7I)

NOTES The default MS-DOS boot loader, which is installed by default if `-o B` is not specified, requires specific MS-DOS system files to make the diskette bootable. These MS-DOS files are not installed when you format a diskette with `mkfs` for `pcfs`, which makes a diskette formatted this way not bootable. Trying to boot from it on an IA based system will result in the following message:

```
Non-System disk or disk error
Replace and strike any key when ready
```

You must format a diskette with the DOS `format` command to install the specific MS-DOS system files required by the default boot loader.

You can use `lofiadm` to create a file that appears to a `mkfs` command (for example, `mkfs_pcfs` or `mkfs_ufs`) as a raw device. You can then use a `mkfs` command to create a file system on that device. See `lofiadm`(1M) for examples of creating a UFS and a PC (FAT) file system on a device created by `lofiadm`.

NAME

mkfs_udfs – construct a udfs file system

SYNOPSIS

mkfs -F udfs [*generic_options*] [-o *specific_options*] *raw_device_file* [*size*]

DESCRIPTION

This is the universal disk format file system (udfs) -specific module of the `mkfs` command. `mkfs` constructs a `udfs` file system with a root directory.

OPTIONS

See `mkfs`(1M) for the list of supported *generic_options*.

The following options are supported:

`-o` *specific_options* Specify a `udfs`-specific option. Specify `udfs` file system specific options in a comma-separated list with no intervening spaces. If invalid options are specified, a warning message is printed and the invalid options are ignored.

The following *specific_options* are available:

N
Print the file system parameters without actually creating the file system.

label=*string*
Specify the label to be written into the volume header structures. Specify *string* as the name of the label. If *string* is not specified, a default *string* is generated in the form of `*NoLabel*`.

OPERANDS

The following operands are supported:

raw_device_file Specify the disk partition on which to write.

size Specify the number of 512-byte blocks in the file system.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWudf

SEE ALSO

`fsck`(1M),`mkfs`(1M), `attributes`(5)

DIAGNOSTICS

`not currently a valid file system`

The specified device does not contain a valid `udfs` file system.

`Invalid size: larger than the partition size`

Number of blocks given as parameter to create the file system is larger than the size of the device specified.

`is mounted can't mkfs`

Device is in use, cannot create file system when the device is in use.

```
preposterous size
```

Negative size parameter provided is invalid.

```
sector size must be between 512, 8192 bytes
```

Sector size given is not in the valid range.

```
Volume integrity sequence descriptors too long
File set descriptor too long.
```

Not enough space to create volume integrity sequence or file set descriptor.

```
mkfs: argument out of range
```

One of the arguments is out of range.

```
mkfs: bad numeric arg
```

One of the arguments is potentially a bad numeric.

NOTES You can use `lofiadm` to create a file that appears to a `mkfs` command (for example, `mkfs_pcfs` or `mkfs_ufs`) as a raw device. You can then use a `mkfs` command to create a file system on that device. See `lofiadm`(1M) for examples of creating a UFS and a PC (FAT) file system on a device created by `lofiadm`.

NAME

mkfs_ufs – construct a ufs file system

SYNOPSIS

mkfs -F ufs [*generic_options*] [-o *FSType_specific_options*] *raw_device_file* [*size*]

DESCRIPTION

The ufs-specific module of mkfs builds a ufs file system with a root directory and a lost+found directory (see fsck(1M))

The ufs-specific mkfs is rarely run directly; use the newfs(1M) command instead.

raw_device_file indicates the disk partition to write on unless the -o N option has been specified, or either the -V or -m generic options are passed from the generic mkfs module. size specifies the number of sectors in the file system. This argument must follow the *raw_device_file* argument and is required (even with -o N), unless the -V or -m generic options are specified.

generic_options are supported by the generic mkfs command. See mkfs(1M) for a description of these options.

OPTIONS

The following options are supported:

-o Use one or more of the following values separated by commas (with no intervening spaces) to specify ufs-specific options:

apc=*n* The number of alternate sectors per cylinder to reserve for bad block replacement (SCSI devices only). The default is 0.

bsize=*n* Logical block size, either 4096 or 8192. The default is 8192. The sun4u architecture does not support the 4096 block size.

cgsize=*n* The number of cylinders per cylinder group (ranging from 16 to 256). The default is calculated by dividing the number of sectors in the file system by the number of sectors in a gigabyte, and then multiplying the result by 32. The default value will always be between 16 and 256. The per-cylinder-group meta data must fit in a space no larger than that available in one logical file system block. If too large a cgsize is requested, it is decreased by the minimum amount necessary.

fragsize=*n* The smallest amount of disk space in bytes to allocate to a file. The smallest amout of disk space in bytes to allocate to a file. If the logical block size is 4096, legal values are 512, 1024, 2048, and 4096. When the logical block size is 8192, legal values are 1024, 2048, 4096, and 8192. The default value is 1024.

free=*n* The minimum percentage of free space to maintain in the file system. This space is off-limits to normal users. Once the file system is filled to this threshold, only the superuser can continue writing to the file system. This

parameter can be subsequently changed using the `tunefs`(1M) command. The default is `10%`.

`gap=`*n* — Rotational delay. The expected time (in milliseconds) to service a transfer completion interrupt and initiate a new transfer on the same disk. The value is used to decide how much rotational spacing to place between successive blocks in a file. This parameter can be subsequently changed using the `tunefs`(1M) command. The default is disk-type dependent.

`maxcontig=`*n* — The maximum number of logical blocks, belonging to one file, that will be allocated contiguously before inserting a rotational delay. For a `4K` file system, the default is `14`; for an `8K` file system it is `7`. This parameter can be subsequently changed using the `tunefs`(1M) command. Where *maxphys* is the maximum size of physical I/O requests supported by UFS (`1048576` bytes), the value for `maxcontig` must obey the relationship:

maxphys `>= maxcontig * logical bsize`

This parameter also controls clustering. Regardless of the value of *gap*, clustering is enabled only when `maxcontig` is greater than 1. Clustering allows higher I/O rates for sequential I/O and is described in `tunefs`(1M).

`N` — Print out the file system parameters without actually creating the file system.

`nbpi=`*n* — The number of bytes per inode, which specifies the density of inodes in the file system. The number is divided into the total size of the file system to determine the fixed number of inodes to create. It should reflect the expected average size of files in the file system. If fewer inodes are desired, a larger number should be used; to create more inodes, a smaller number should be given. The default is `2048`.

`nrpos=`*n* — The number of different rotational positions in which to divide a cylinder group. The default is `8`.

`nsect=`*n* — The number of sectors per track on the disk. The default is `32`.

`ntrack=`*n* — The number of tracks per cylinder on the disk. The default is `16`.

`opt=`*a* — Space or time optimization preference; `s` specifies optimization for space, `t` specifies optimization for

time. The default is `t`. This parameter may be subsequently changed with the `tunefs`(1M) command.

`rps=`*n* The rotational speed of the disk, in revolutions per second. The default is `60`.

Alternatively, parameters can be entered as a list of space-separated values (without keywords) whose meaning is positional. In this case, the `-o` option is omitted and the list follows the size operand. This is the way `newfs` passes the parameters to `mkfs`.

OPERANDS

The following operands are supported:

raw_device_file The disk partition on which to write.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`fsck`(1M), `mkfs`(1M), `newfs`(1M), `tunefs`(1M), `dir_ufs`(4), `fs_ufs`(4), `attributes`(5)

DIAGNOSTICS

The following error message occurs typically on very high density disks. On such disks, the file system structure cannot encode the proper disk layout information, resulting in suboptimal performance.

```
Warning: insufficient space in super block for
rotational layout tables with nsect sblock.fs_nsect
and ntrak sblock.fs_ntrak. (File system performance may be impaired.)
```

The following error message occurs if a user request for inodes or bytes (with the `nbpi` keyword) and the disk geometry results in a situation in which the last truncated cylinder group cannot contain the correct number of data blocks; some disk space is wasted.

```
Warning: inode blocks/cyl group (grp) >= data blocks (num) in last cylinder
```

The following error message occurs if the user parameters and disk geometry conflict; some disk space is lost. A possible cause is the specified size being smaller than the partition size.

```
Warning: num sector(s) in last cylinder group unallocated
```

NOTES

You can use `lofiadm` to create a file that appears to a `mkfs` command (for example, `mkfs_pcfs` or `mkfs_ufs`) as a raw device. You can then use a `mkfs` command to create a file system on that device. See `lofiadm`(1M) for examples of creating a UFS and a PC (FAT) file system on a device created by `lofiadm`.

NAME

mknod – make a special file

SYNOPSIS

mknod *name* b *major minor*

mknod *name* c *major minor*

mknod *name* p

DESCRIPTION

mknod makes a directory entry for a special file.

OPTIONS

The following options are supported:

b Create a block-type special file.

c Create a character-type special file.

p Create a FIFO (named pipe).

OPERANDS

The following operands are supported:

major The *major* device number.

minor The *minor* device number; can be either decimal or octal. The assignment of major device numbers is specific to each system. You must be the super-user to use this form of the command.

name A special file to be created.

USAGE

See largefile(5) for the description of the behavior of mknod when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

ftp(1), in.ftpd(1M), mknod(2), symlink(2), attributes(5), largefile(5)

NOTES

If mknod(2) is used to create a device, the major and minor device numbers are always interpreted by the kernel running on that machine.

With the advent of physical device naming, it would be preferable to create a symbolic link to the physical name of the device (in the /devices subtree) rather than using mknod.

NAME

modinfo – display information about loaded kernel modules

SYNOPSIS

/usr/sbin/modinfo [-c] [-w] [-i *module-id*]

DESCRIPTION

The modinfo utility displays information about the loaded modules. The format of the information is as follows:

*Id Loadaddr Size Info Rev Module Name*where *Id* is the module ID, *Loadaddr* is the starting text address in hexadecimal, *Size* is the size of text, data, and bss in hexadecimal bytes, *Info* is module specific information, *Rev* is the revision of the loadable modules system, and *Module Name* is the filename and description of the module.

The module specific information is the block and character major numbers for drivers, the system call number for system calls, or, for other module types, the index into the appropriate kernel table:

fmodsw	for STREAMS modules
vfssw	for filesystems
class	for scheduling classes
execsw	for exec modules

OPTIONS

The following options are supported:

-c	Displays the number of instances of the module loaded and the module's current state.
-i *module-id*	Displays information about this module only.
-w	Does not truncate module information at 80 characters.

EXAMPLES

EXAMPLE 1 Using the modinfo command.

The following example displays the status of module 3:

```
example% modinfo -i 3Id  Loadaddr  Size  Info  Rev  Module Name
3   f5a7a000  3bc0  1     1    specfs (filesystem for specfs)
```

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

modload(1M), modunload(1M), attributes(5)

NAME

modload – load a kernel module

SYNOPSIS

modload [-p] [-e *exec_file*] *filename*

DESCRIPTION

The `modload` command loads the loadable module *filename* into the running system.

filename is an object file produced by `ld -r`. If *filename* is an absolute pathname then the file specified by that absolute path is loaded. If *filename* does not begin with a slash (/), then the path to load *filename* is relative to the current directory unless the `-p` option is specified.

The kernel's `modpath` variable can be set using the `/etc/system` file. The default value of the kernel's `modpath` variable is set to the path where the operating system was loaded. Typically this is `/kernel /usr/kernel`.

For example, the following command looks for `./drv/foo`:

```
example# modload drv/foo
```

The following command looks for `/kernel/drv/foo` and then `/usr/kernel/drv/foo`:

```
example# modload -p drv/foo
```

OPTIONS

The following options are supported:

`-e` *exec_file* — Specify the name of a shell script or executable image file that is executed after the module is successfully loaded. The first argument passed is the module ID (in decimal). The other argument is module specific. The module specific information is: the block and character major numbers for drivers, the system call number for system calls, or, for other module types, the index into the appropriate kernel table. See `modinfo`(1M)

`-p` — Use the kernel's internal modpath variable as the search path for the module.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`ld`(1), `add_drv`(1M), `kernel`(1M), `modinfo`(1M), `modunload`(1M), `system`(4), `attributes`(5), `modldrv`(9S), `modlinkage`(9S), `modlstrmod`(9S), `module_info`(9S)

Writing Device Drivers

NOTES

Use `add_drv`(1M) to add device drivers, not `modload`. See *Writing Device Drivers* for procedures on adding device drivers.

NAME

modunload – unload a module

SYNOPSIS

modunload `-i` *module_id* `[-e` *exec_file*`]`

DESCRIPTION

`modunload` unloads a loadable module from the running system. The *module_id* is the ID of the module as shown by `modinfo`(1M). If `ID` is `0`, all modules that were autoloaded which are unloadable, are unloaded. Modules loaded by `modload`(1M) are not affected.

OPTIONS

The following options are supported:

`-e` *exec_file* — Specify the name of a shell script or executable image file to be executed before the module is unloaded. The first argument passed is the module id (in decimal). There are two additional arguments that are module specific. For loadable drivers, the second argument is the driver major number. For loadable system calls, the second argument is the system call number. For loadable exec classes, the second argument is the index into the `execsw` table. For loadable filesystems, the second argument is the index into the `vfssw` table. For loadable streams modules, the second argument is the index into the `fmodsw` table. For loadable scheduling classes, the second argument is the index into the class array. Minus one is passed for an argument that does not apply.

`-i` *module_id* — Specify the module to be unloaded.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`modinfo`(1M), `modload`(1M), `update_drv`(1M), `attributes`(5)

NOTES

The `modunload` command has often been used on driver modules to force the system to reread the associated driver configuration file. While this works in Solaris 9, this behavior might break in future releases. The supported way for rereading driver configuration file is through the `update_drv`(1M) command.

NAME

mofcomp – compile MOF files into CIM classes

SYNOPSIS

```
/usr/sadm/bin/mofcomp [-c cimom_hostname ] [-h] [-j filename] [-n
     namespace] [-o dirname] [-p password ] [-CIQ] [-u username] [-v ]
     [-version] [-x] file
```

DESCRIPTION

The `mofcomp` utility is executed during installation to compile managed object format (MOF) files that describe the Common Information Model (CIM) and Solaris Schemas into the CIM Object Manager Repository, a central storage area for management data. The CIM Schema is a collection of class definitions used to represent managed objects that occur in every management environment. The Solaris Schema is a collection of class definitions that extend the CIM Schema and represent managed objects in a typical Solaris operating environment.

The `mofcomp` utility must be run as root or as a user with write access to the namespace in which you are compiling.

MOF is a language for defining CIM classes and instances. MOF files are ASCII text files that use the MOF language to describe CIM objects. A CIM object is a computer representation or model of a managed resource, such as a printer, disk drive, or CPU.

Many sites store information about managed resources in MOF files. Because MOF can be converted to Java, Java applications that can run on any system with a Java Virtual Machine can interpret and exchange this information. You can also use the `mofcomp` utility to compile MOF files at any time after installation.

OPTIONS

The following options are supported:

`-c` *cimom_hostname*	Specify a remote system running the CIM Object Manager.
`-C`	Run the compiler set with the class option, which updates a class if it exists, and returns an error if the class does not exist. If you do not specify this option, the compiler adds a CIM class to the connected namespace, and returns an error if the class already exists.
`-h`	List the arguments to the `mofcomp` utility.
`-I`	Run the compiler set with the instance option, which updates an instance if it exists, and returns an error if the instance does not exist. If you do not specify this option, the compiler adds a CIM instance to the connected namespace, and returns an error if the instance already exists.
`-j` *filename*	Generate Java Beans and Java Interfaces to manage the CIM instances related to the CIM classes in the MOF being compiled. The contents of *filename* are:

PACKAGE=*Java package name*
IMPORTS=*import1*:...:*importN*
<EXCEPTIONS=*exception1*:...:*exceptionN*

PACKAGE is a valid Java package name to include in all generated Java source. IMPORTS is an optional colon separated list of valid Java classes to be imported in all generated Java source. EXCEPTIONS is an optional colon separated list of valid Java exceptions to be thrown by the methods in all generated Java source.

-n *namespace*

Requests that the compiler load the MOF file into the namespace specified as *namespace*. The default namespace (root\cimv2) is used unless this switch is used or a #pragma *namespace* ("namespace") statement appears in the MOF file. If both the -n *namespace* switch and the #pragma *namespace* construct are used, all namespaces are created, but the objects are created only in the #pragma namespaces.

-o *dirname*

Run compiler in standalone mode, without the CIM Object Manager. Specify dirname as the directory in which the compiler output is to be stored. In this mode, the CIM Object Manager need not be running.

-p *password*

Specify a password for connecting to the CIM Object Manager. Use this option for compilations that require privileged access to the CIM Object Manager. If you specify both -p and -u, you must type the password on the command line, which can pose a security risk. A more secure way to specify a password is to specify -u but not -p, so that the compiler will prompt for the password.

-Q

Run the compiler set with the qualifier types option, which updates a qualifier type if it exists, and returns an error if the qualifier type does not exist. If you do not specify this option, the compiler adds a CIM qualifier type to the connected namespace, and returns an error if the qualifier type already exists.

-u *username*

Specify user name for connecting to the CIM Object Manager. Use this option for compilations that require privileged access to the CIM Object Manager. If you specify both -p and -u, you must type the password on the command line, which can pose a security risk. A more secure way to specify a password is to specify -u but not -p, so that the compiler will prompt for the password.

`-v`	Run the compiler in verbose mode, which displays compiler messages.
`-version`	Display the version of the MOF compiler.
`-x`	Generate XML documents for the CIM classes defined in the input MOF file.

OPERANDS

The following operands are supported:

file The pathname of the file to be compiled.

EXIT STATUS

The `mofcomp` utility exits with `0` upon success and a positive integer upon failure.

FILES

MOF files are installed in `/usr/sadm/mof`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWwbcor

SEE ALSO

`init.wbem`(1M), `mofreg`(1M), `wbemadmin`(1M), `wbemlogviewer`(1M), `attributes`(5), `wbem`(5),

NAME

mofreg – register MOF classes with WBEM services

SYNOPSIS

/usr/sadm/bin/mofreg `-r` *tag file*

/usr/sadm/bin/mofreg `-s`

/usr/sadm/bin/mofreg `-u` *tag* [*file*]

DESCRIPTION

The `mofreg` command is used by package and patch install scripts, or by any applications that wish to register managed object format (MOF) classes with Sun The Web-Based Enterprise Management (WBEM) services.

The WBEM services daemon (Common Information Model or CIM object manager) processes at start up the files that are specified by `mofreg` commands. Files are processed in the order that the individual `mofreg` commands are executed.

As an alternative to using the `mofreg` command, MOFs can be registered or unregistered by manipulating directories in `/var/sadm/wbem/logr`. Instead of running the `mofreg -r tag file` version fo the command you can create a directory named `tag` under `/var/sadm/wbem/logr/preReg` and copy `file` to the `tag` directory.

Similarly, instead of running the `mofreg -u tag [file]` command, you can create a directory named `tag` under `/var/sadm/wbem/logr/preUnreg` and copy the optional `file` to the `tag` directory.

The entries are processed in increasing order of last modification time of the `tag` directories. As with the `mofreg` command, processing is done at next restart or by using the `-s` option.

This alternative mechanism is typically used in package install scripts which do not have access to `/usr`, and therefore do not have access to the `mofreg` command. This case arises when packages are installed for diskless clients.

OPTIONS

The following options are supported:

`-r` *tag file*

The *file* argument is the actual MOF registration file. Its form is identical to the MOF syntax as defined by the Distributed Management Task Force (DMTF). The only difference is the addition of the following 3 new pseudo-pragmas, which are variations of the namespace pragma. The name of file cannot end in `.unreg`.

```
#pragma namespace("__create")
#pragma namespace("__delete")
#pragma namespace("__modify")
```

These three pragmas are used specify if the elements following the pragmas should be created, deleted, or modified by the CIM object manager. The `__delete` pragma can currently only be applied for a `mofreg -u` command.

The *tag* argument is a unique string that specifies the identity of the registry action. This tag can be set to the package name or the patch number if the `mofreg` script is being invoked through packages/patches, though any tag can be specified.

Errors and warnings that are encountered when the CIM object manager handles the `mofreg` script are logged. Processing of the `mofreg` script stops at the first error. Specific warnings include:

```
Element already defined - the element already exists and
cannot be created.

Element not found - the element does not exist and
cannot be modified.
```

The error conditions are:

```
Key modification - A class cannot be modified if its keys
are being changed.

Other mod compilation errors.
```

`-s` Forces the CIM object manager to immediately process outstanding registry requests, instead of at the next restart. This currently requires Java.

`-u` *tag* [*file*] Undoes the operations performed during `mof` registry.

The *tag* argument must correspond to the value set during the original `mofreg` invocation. If no `mofreg` was done with the original *tag*, the command does not succeed.

If required, an *unreg* file can be specified. If no *unreg* file is specified, the CIM object manager automatically undoes the actions of the registry. Any class created by the registry process is removed and any classes modified by the registry revert to the old state.

The `mofreg` command does not take care of cases where packages and patches make conflicting changes to classes. This should be taken care of by the standard patch and package conflict resolution.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` An error occurred. The reason for error is displayed.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWwbcou

SEE ALSO `init.wbem`(1M), `mofcomp`(1M), `wbemadmin`(1M), `wbemlogviewer`(1M), `attributes`(5), `wbem`(5)

NAME

monitor – SPARC system PROM monitor

SYNOPSIS

```
L1–A

BREAK

initial system power-on

exit from a client program, e.g., the Operating System
```

DESCRIPTION

The CPU board of a workstation contains one or more EPROMs or EEPROMs. The program which executes from the PROMs is referred to as "the monitor". Among other things, the monitor performs system initialization at power-on and provides a user interface.

Monitor Prompt

The monitor of earlier workstations was known as the `SunMON` monitor and displayed the `>` for its prompt. See the `SunMON MONITOR USAGE` section for further details.

Existing workstations use a monitor which is known as the OpenBoot monitor. The OpenBoot monitor typically displays `ok` as its prompt, but it may also display the `>` prompt under certain circumstances.

If the `'auto-boot?'` NVRAM parameter is set to `'false'` when the workstation is powered on then the system will not attempt to boot and the monitor will issue its prompt. If `'auto-boot'` is set to `'true'` then the system will initiate the boot sequence. The boot sequence can be aborted by simultaneously pressing two keys on the system's keyboard: L1 and A (on older keyboards), or Stop and A (on newer keyboards). Note that either a lower case `'a'` or an upper case `'A'` will work for the keyboard abort sequence. If a console has been attached via one of the system's serial ports then the abort sequence can be accomplished by sending a BREAK – see the `tip`(1) manpage.

When the NVRAM `'security-mode'` parameter has been turned on, or when the value of the `'sunmon-compat?'` parameter is true, then the OpenBoot monitor will display the message:

Type b (boot), c (continue), or n (new command mode)

and the `>` prompt will appear.

OPENBOOT PROM USAGE

Some of the more useful commands that can be issued from OpenBoot's `ok` prompt are described here. Refer to the *OpenBoot 2.x Command Reference Manual* book for a complete list of commands.

Help

Help for various functional areas of the OpenBoot monitor can be obtained by typing `help`. The help listing will provide a number of other key words which can then be used in the help command to provide further details.

NVRAM Parameters

Each workstation contains one or more NVRAM devices which contains unique system ID information, as well as a set of user-configurable parameters. The NVRAM parameters allow the user a certain level of flexibility in configuring the system to act in a given manner under a specific set of circumstances.

See the `eeprom`(1M) manpage for a description of the parameters. This manpage also describes a way of setting the parameters from the OS level.

The following commands can be used at the OpenBoot monitor to access the NVRAM parameters.

`printenv`	Used to list the NVRAM parameters, along with their default values and current values.
`setenv` *pn pv*	Used to set or modify a parameter. The *pn* represents the parameter name, and *pv* represents the parameter value.
`set-default`*pn*	Used to set an individual parameter back to its default value.
`set-defaults`	Used to reset all parameters to their default values. (Note that `'set-defaults'` only affects parameters that have assigned default values.)

Hardware Checks and Diagnostics

The following commands are available for testing or checking the system's hardware. If the `'diag-switch?'` NVRAM parameter is set to true when the system is powered on, then a Power-On Self Test (POST) diagnostic will be run, if present, sending its results messages to the system's serial port A. Not all of the commands shown are available on all workstations.

`test-all`	Run the diagnostic tests on each device which has provided a self-test.
`test floppy`	Run diagnostics on the system's floppy device.
`test /memory`	Run the main memory tests. If the NVRAM parameter `'diag-switch?'` is set to true, then all of main memory is tested. If the parameter is false then only the amount of memory specified in the `'selftest-#megs'` NVRAM parameter will be tested.
`test net`	Test the network connection for the on-board network controller.
`watch-net`	Monitor the network attached to the on-board net controller.
`watch-net-all`	Monitor the network attached to the on-board net controller, as well as the network controllers installed in SBus slots.
`watch-clock`	Test the system's clock function.

System Information

The following commands are available for displaying information about the system. Not all commands are available on all workstations.

`banner`	Display the power-on banner.
`.enet-addr`	Display the system's Ethernet address.

`.idprom`	Display the formatted contents of the IDPROM.
`module-info`	Display information about the system's processor(s).
`probe-scsi`	Identify the devices attached to the on-board SCSI controller.
`probe-scsi-all`	Identify the devices attached to the on-board SCSI controller as well as those devices which are attached to SBus SCSI controllers.
`show-disks`	Display a list of the device paths for installed SCSI disk controllers.
`show-displays`	Display a list of the device paths for installed display devices.
`show-nets`	Display a list of the device paths for installed Ethernet controllers.
`show-sbus`	Display list of installed SBus devices.
`show-tapes`	Display a list of the device paths for installed SCSI tape controllers.
`show-ttys`	Display a list of the device paths for tty devices.
`.traps`	Display a list of the SPARC trap types.
`.version`	Display the version and date of the OpenBoot PROM.

Emergency Commands

These commands must be typed from the keyboard, they will not work from a console which is attached via the serial ports. With the exception of the Stop-A command, these commands are issued by pressing and holding down the indicated keys on the keyboard immediately after the system has been powered on. The keys must be held down until the monitor has checked their status. The Stop-A command can be issued at any time after the console display begins, and the keys do not need to be held down once they've been pressed. The Stop-D, Stop-F and Stop-N commands are not allowed when one of the security modes has been set. Not all commands are available on all workstations.

Stop (L1)	Bypass the Power-On Self Test (POST). This is only effective if the system has been placed into the diagnostic mode.
Stop-A (L1-A)	Abort the current operation and return to the monitor's default prompt.
Stop-D (L1-D)	Set the system's `'diag-switch?'` NVRAM parameter to `'true'`, which places the system in diagnostic mode. POST diagnostics, if present, will be run, and the messages will be displayed via the system's serial port A.

Stop-F (L1-F)	Enter the OpenBoot monitor before the monitor has probed the system for devices. Issue the `'fexit'` command to continue with system initialization.
Stop-N (L1-N)	Causes the NVRAM parameters to be reset to their default values. Note that not all parameters have default values.

Line Editor Commands

The following commands can be used while the monitor is displaying the `ok` prompt. Not all of these editing commands are available on all workstations.

CTRL-A	Place the cursor at the start of line.
CTRL-B	Move the cursor backward one character.
ESC-B	Move the cursor backward one word.
CTRL-D	Erase the character that the cursor is currently highlighting.
ESC-D	Erase the portion of word from the cursor's present position to the end of the word.
CTRL-E	Place the cursor at the end of line.
CTRL-F	Move the cursor forward one character.
ESC-F	Move the cursor forward one word.
CTRL-H	Erase the character preceding the cursor (also use Delete or Back Space)
ESC-H	Erase the portion of the word which precedes the cursor (use also CTRL-W)
CTRL-K	Erase from the cursor's present position to the end of the line.
CTRL-L	Show the command history list.
CTRL-N	Recall the next command from the command history list
CTRL-P	Recall a previous command from the command history list.
CTRL-Q	Quote the next character (used to type a control character).
CTRL-R	Retype the current line.
CTRL-U	Erase from the cursor's present position to the beginning of the line.
CTRL-Y	Insert the contents of the memory buffer into the line, in front (to the left) of the cursor.

nvramrc

The `nvramrc` is an area of the system's NVRAM where users may store Forth programs. The programs which are stored in the `nvramrc` will executed each time the system is reset, provided that the `'use-nvramrc?'` NVRAM parameter has been set to `'true'`. Refer to the *OpenBoot 2.x Command Reference Manual* book for information on how to edit and use the `nvramrc`.

Restricted Monitor

The command `'old-mode'` is used to move OpenBoot into a restricted monitor mode, causing the `>` prompt to be displayed. Only three commands are allowed while in the restricted monitor; the `'go'` command (to resume a program which was interrupted with the Stop-A command), the `'n'` command (to return to the normal OpenBoot monitor), and boot commands. The restricted monitor's boot commands will approximate the older `SunMON` monitor's boot command syntax. If a `'security-mode'` has been turned on then the restricted monitor becomes the default monitor environment. The restricted monitor may also become the default environment if the `'sunmon-compat?'` NVRAM parameter is set to true. (Note that not all workstations will have the `'sunmon-compat?'` parameter.)

SunMON PROM USAGE

The following commands are available systems with older `SunMON`-based PROM:

`+|-`
Increment or decrement the current address and display the contents of the new location.

`^C` *source destination n*
(caret-C) Copy, byte-by-byte, a block of length *n* from the `source` address to the *destination* address.

`^I` *program*
(caret-I) Display the compilation date and location of *program*.

`^T` *virtual_address*
(caret-T) Display the physical address to which *virtual_address* is mapped.

`b` [`!`] [*device* [`(`*c*`,`*u*`,`*p*`)`]] [*pathname*] [*arguments_list*]

`b[?]`
Reset appropriate parts of the system and bootstrap a program. A '`!`' (preceding the *device* argument) prevents the system reset from occurring. Programs can be loaded from various devices (such as a disk, tape, or Ethernet). '`b`' with no arguments will cause a default boot, either from a disk, or from an Ethernet controller. '`b?`' displays all boot devices and their *device*s.

device	one of	
	`le`	Lance Ethernet
	`ie`	Intel Ethernet
	`sd`	SCSI disk, CDROM
	`st`	SCSI 1/4" or 1/2" tape
	`fd`	Diskette
	`id`	IPI disk
	`mt`	Tape Master 9-track 1/2" tape
	`xd`	Xylogics 7053 disk
	`xt`	Xylogics 1/2" tape

`xy`	Xylogics 440/450 disk
c	A controller number (`0` if only one controller),
u	A unit number (`0` if only one driver), and
p	A partition.
pathname	A pathname for a program such as `/stand/diag`.
arguments_list	A list of up to seven arguments to pass to the program being booted.

`c` [*virtual_address*]
Resume execution of a program. When given, *virtual_address* is the address at which execution will resume. The default is the current PC. Registers are restored to the values shown by the `d`, and `r` commands.

`d` [*window_number*]
Display (dump) the state of the processor. The processor state is observable only after:

- An unexpected trap was encountered.
- A user program dropped into the monitor (by calling *abortent*).
- The user manually entered the monitor by typing L1–A or BREAK.

The display consists of the following:

- The special registers: PSR, PC, nPC, TBR, WIM, and Y
- Eight global registers
- 24 window registers (8 *in*, 8 *local*, and 8 *out*), corresponding to one of the 7 available windows. If a Floating-Point Unit is on board, its status register along with 32 floating-point registers are also shown.

window_number	Display the indicated *window_number*, which can be any value between `0` and `6`, inclusive. If no window is specified and the PSR's current window pointer contains a valid window number, registers from the window that was active just prior to entry into the monitor are displayed. Otherwise, registers from window 0 are displayed.

`e` [*virtual_address*] [*action*] . . .
Open the 16-bit word at *virtual_address* (default zero). The address is interpreted in the address space defined by the `s` command. See the `a` command for a description of *action*.

`f` *virtual_address1 virtual_address2 pattern* [*size*]
Fill the bytes, words, or long words from *virtual_address1* (lower) to *virtual_address2* (higher) with the constant, *pattern*. The `size` argument can take one of the following values:

`b`	byte format (the default)
`w`	word format

`l` long word format

For example, the following command fills the address block from `0x1000` to `0x2000` with the word pattern, `0xABCD`:

```
f 1000 2000 ABCD W
```

`g` [*vector*] [*argument*]
`g` [*virtual_address*] [*argument*]
Goto (jump to) a predetermined or default routine (first form), or to a user-specified routine (second form). The value of *argument* is passed to the routine. If the *vector* or *virtual_address* argument is omitted, the value in the PC is used as the address to jump to.

To set up a predetermined routine to jump to, a user program must, prior to executing the monitor's `g` command, set the variable `*romp->v_vector_cmd` to be equal to the virtual address of the desired routine. Predetermined routines need not necessarily return control to the monitor.

The default routine, defined by the monitor, prints the user-supplied *vector* according to the format supplied in *argument*. This format can be one of:

`%x` hexadecimal

`%d` decimal

`g0`
Force a panic and produce a crash dump when the monitor is running as a result of the system being interrupted,

`g4`
(Sun-4 systems only) Force a kernel stack trace when the monitor is running as a result of the system being interrupted,

`h`
Display the help menu for monitor commands and their descriptions. To return to the monitor's basic command level, press ESCAPE or `q` before pressing RETURN.

`i` [*cache_data_offset*] [*action*] . . .
Modify cache data RAM command. Display and/or modify one or more of the cache data addresses. See the `a` command for a description of *action*.

`j` [*cache_tag_offset*] [*action*] . . .
Modify cache tag RAM command. Display and/or modify the contents of one or more of the cache tag addresses. See the `a` command for a description of *action*.

`k` [*reset_level*]
Reset the system, where *reset_level* is:

`0` Reset VMEbus, interrupt registers, video monitor (Sun-4 systems). This is the default.

`1` Software reset.

`2` Power-on reset. Resets and clears the memory. Runs the EPROM-based diagnostic self test, which can take several minutes, depending upon how much memory is being tested.

`kb`
Display the system banner.

`l` [*virtual_address*] [*action*] . . .
Open the long word (32 bit) at memory address *virtual_address* (default zero). The address is interpreted in the address space defined by the `s` command (below). See the `a` command for a description of *action*.

`m` [*virtual_address*] [*action*] . . .
Open the segment map entry that maps *virtual_address* (default zero). The address is interpreted in the address space defined by the `s` command. See the `a` command for a description of *action*.

`ne`

`ni`
Disable, enable, or invalidate the cache, respectively.

`o` [*virtual_address*] [action*]* . . .
Open the byte location specified by *virtual_address* (default zero). The address is interpreted in the address space defined by the `s` command. See the `a` command for a description of *action*.

`p` [*virtual_address*] [*action*]. . .
Open the page map entry that maps *virtual_address* (default zero) in the address space defined by the `s` command. See the `a` command for a description of *action*.

`q` [*eeprom_offset*] [*action*]. . .
Open the EEPROM *eeprom_offset* (default zero) in the EEPROM address space. All addresses are referenced from the beginning or base of the EEPROM in physical address space, and a limit check is performed to insure that no address beyond the EEPROM physical space is accessed. This command is used to display or modify configuration parameters, such as: the amount of memory to test during self test, whether to display a standard or custom banner, if a serial port (A or B) is to be the system console, etc. See the `a` command for a description of *action*.

`r` [*register_number*]
`r` [*register_type*]
`r` [*w window_number*]
Display and/or modify one or more of the IU or FPU registers. A hexadecimal *register_number* can be one of:

`0x00–0x0f`	window(0,i0)–window(0,i7), window(0,i0)—window(0,i7)
`0x16–0x1f`	window(1,i0)–window(1,i7), window(1,i0)—window(1,i7)

`0x20–0x2f`	window(2,i0)–window(2,i7), window(2,i0)—window(2,i7)
`0x30–0x3f`	window(3,i0)–window(3,i7), window(3,i0)—window(3,i7)
`0x40–0x4f`	window(4,i0)–window(4,i7), window(4,i0)—window(4,i7)
`0x50–0x5f`	window(5,i0)–window(5,i7), window(5,i0)—window(5,i7)
`0x60–0x6f`	window(6,i0)–window(6,i7), window(6,i0)—window(6,i7)
`0x70–0x77`	`g0, g1, g2, g3, g4, g5, g6, g7`
`0x78–0x7d`	PSR, PC, nPC, WIM, TBR, Y.
`0x7e–0x9e`	FSR, f0–f31

Register numbers can only be displayed after an unexpected trap, a user program has entered the monitor using the *abortent* function, or the user has entered the monitor by manually typing L1–A or BREAK.

If a *register_type* is given, the first register of the indicated type is displayed. *register_type* can be one of:

`f`	floating-point
`g`	global
`s`	special

If `w` and a *window_number* (`0`—`6`) are given, the first *in*-register within the indicated window is displayed. If *window_number* is omitted, the window that was active just prior to entering the monitor is used. If the PSR's current window pointer is invalid, window 0 is used.

`s` [*asi*])
Set or display the Address Space Identifier. With no argument, `s` displays the current Address Space Identifier. The *asi* value can be one of:

`0x2`	control space
`0x3`	segment table
`0x4`	Page table
`0x8`	user instruction
`0x9`	supervisor instruction
`0xa`	user data
`0xb`	supervisor data

`0xc` flush segment

`0xd` flush page

`0xe` flush context

`0xf` cache data

`u` [`echo`]

`u` [*port*] [*options*] [*baud_rate*]

`u` [`u`] [*virtual_address*]

With no arguments, display the current I/O device characteristics including: current input device, current output device, baud rates for serial ports A and B, an input-to-output echo indicator, and virtual addresses of mapped UART devices. With arguments, set or configure the current I/O device. With the `u` argument (`uu`. . .), set the I/O device to be the *virtual_address* of a UART device currently mapped.

`echo` Can be either `e` to enable input to be echoed to the output device, or `ne`, to indicate that input is not echoed.

port Assign the indicated *port* to be the current I/O device. *port* can be one of:

- `a` serial port A
- `b` serial port B
- `k` the workstation keyboard
- `s` the workstation screen

baud_rate Any legal baud rate.

options can be any combination of:

- `i` input
- `o` output
- `u` UART
- `e` echo input to output
- `ne` do not echo input
- `r` reset indicated serial port (`a` and `b` ports only)

If either `a` or `b` is supplied, and no *options* are given, the serial port is assigned for both input and output. If `k` is supplied with no options, it is assigned for input only. If `s` is supplied with no options, it is assigned for output only.

`v` *virtual_address1 virtual_address2* [`size`]
Display the contents of *virtual_address1* (lower) *virtual_address2* (higher) in the format specified by `size`:

`b`	byte format (the default)
`w`	word format
`l`	long word format

Enter return to pause for viewing; enter another return character to resume the display. To terminate the display at any time, press the space bar.

For example, the following command displays the contents of virtual address space from address `0x1000` to `0x2000` in word format:

```
v 1000 2000 W
```

`w` [*virtual_address*] [*argument*]
Set the execution vector to a predetermined or default routine. Pass *virtual_address* and *argument* to that routine.

To set up a predetermined routine to jump to, a user program must, prior to executing the monitor's `w` command, set the variable `*romp->v_vector_cmd` to be equal to the virtual address of the desired routine. Predetermined routines need not necessarily return control to the monitor.

The default routine, defined by the monitor, prints the user-supplied *vector* according to the format supplied in *argument*. This format can be one of:

`%x`	hexadecimal
`%d`	decimal

`x`
Display a menu of extended tests. These diagnostics permit additional testing of such things as the I/O port connectors, video memory, workstation memory and keyboard, and boot device paths.

`y c` *context_number*

`y p|s` *context_number virtual_address*
Flush the indicated context, context page, or context segment.

`c`	flush context *context_number*
`p`	flush the page beginning at *virtual_address* within context *context_number*
`s`	flush the segment beginning at *virtual_address* within context *context_number*

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Architecture	SPARC

SEE ALSO `tip`(1), `boot`(1M), `eeprom`(1M), `attributes`(5)

OpenBoot 2.x Command Reference Manual

NAME

mount, umount – mount or unmount file systems and remote resources

SYNOPSIS

```
mount [-p | -v]

mount [-F FSType] [generic_options] [-o specific_options] [-O]special |
     mount_point

mount [-F FSType] [generic_options] [-o specific_options] [-O] special
     mount_point

mount -a [-F FSType] [-V] [current_options] [-o specific_options]
     [mount_point...]

umount [-f] [-V] [-o specific_options]special | mount_point

umount -a [-f] [-V] [-o specific_options] [mount_point...]
```

DESCRIPTION

`mount` attaches a file system to the file system hierarchy at the *mount_point*, which is the pathname of a directory. If *mount_point* has any contents prior to the `mount` operation, these are hidden until the file system is unmounted.

`umount` unmounts a currently mounted file system, which may be specified either as a *mount_point* or as *special*, the device on which the file system resides.

The table of currently mounted file systems can be found by examining the mounted file system information file. This is provided by a file system that is usually mounted on `/etc/mnttab`. The mounted file system information is described in `mnttab`(4). Mounting a file system adds an entry to the mount table; a `umount` removes an entry from the table.

When invoked with both the *special* and *mount_point* arguments and the `-F` option, `mount` validates all arguments except for *special* and invokes the appropriate *FSType*-specific `mount` module. If invoked with no arguments, `mount` lists all the mounted file systems recorded in the mount table, `/etc/mnttab`. If invoked with a partial argument list (with only one of *special* or *mount_point*, or with both *special* or *mount_point* specified but not *FSType*), `mount` will search `/etc/vfstab` for an entry that will supply the missing arguments. If no entry is found, and the special argument starts with "/", the default local file system type specified in `/etc/default/fs` will be used. Otherwise the default remote file system type will be used. The default remote file system type is determined by the first entry in the `/etc/dfs/fstypes` file. After filling in missing arguments, `mount` will invoke the *FSType*-specific `mount` module.

Only a super-user can mount or unmount file systems using `mount` and `umount`. However, any user can use `mount` to list mounted file systems and resources.

OPTIONS

`-F` *FSType*
Used to specify the *FSType* on which to operate. The *FSType* must be specified or must be determinable from `/etc/vfstab`, or by consulting `/etc/default/fs` or `/etc/dfs/fstypes`.

`-a` [*mount_points*. . .]
Perform mount or umount operations in parallel, when possible.

If mount points are not specified, `mount` will mount all file systems whose `/etc/vfstab` "mount at boot" field is "yes". If mount points are specified, then `/etc/vfstab` "mount at boot" field will be ignored.

If mount points are specified, `umount` will only `umount` those mount points. If none is specified, then `umount` will attempt to `unmount` all file systems in `/etc/mnttab`, with the exception of certain system required file systems: `/`, `/usr`, `/var`, `/var/adm`, `/var/run`, `/proc`, `/dev/fd` and `/tmp`.

`-f`
Forcibly unmount a file system.

Without this option, `umount` does not allow a file system to be unmounted if a file on the file system is busy. Using this option can cause data loss for open files; programs which access files after the file system has been unmounted will get an error (`EIO`).

`-p`
Print the list of mounted file systems in the `/etc/vfstab` format. Must be the only option specified. See `BUGS`.

`-v`
Print the list of mounted file systems in verbose format. Must be the only option specified.

`-V`
Echo the complete command line, but do not execute the command. `umount` generates a command line by using the options and arguments provided by the user and adding to them information derived from `/etc/mnttab`. This option should be used to verify and validate the command line.

generic_options
Options that are commonly supported by most *FSType*-specific command modules. The following options are available:

`-m`
Mount the file system without making an entry in `/etc/mnttab`.

`-g`
Globally mount the file system. On a clustered system, this globally mounts the file system on all nodes of the cluster. On a non-clustered system this has no effect.

`-o`
Specify *FSType*-specific options in a comma separated (without spaces) list of suboptions and keyword-attribute pairs for interpretation by the *FSType*-specific module of the command. (See `mount_ufs`(1M).) When you use `-o` with a file system that has an entry in `/etc/vfstab`, any `mount` options entered for that file system in `/etc/vfstab` are ignored.

`-O`
Overlay mount. Allow the file system to be mounted over an existing mount point, making the underlying file system inaccessible. If a mount is attempted on a pre-existing mount point without setting this flag, the mount will fail, producing the error "device busy".

`-r`
Mount the file system read-only.

USAGE

See `largefile`(5) for the description of the behavior of `mount` and `umount` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

FILES

`/etc/mnttab`	Table of mounted file systems.
`/etc/default/fs`	Default local file system type. Default values can be set for the following flags in `/etc/default/fs`. For example: `LOCAL=ufs` `LOCAL:` The default partition for a command if no *FSType* is specified.
`/etc/vfstab`	List of default parameters for each file system.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`mount_cachefs`(1M), `mount_hsfs`(1M), `mount_nfs`(1M), `mount_pcfs`(1M), `mount_tmpfs`(1M), `mount_ufs`(1M), `mountall`(1M), `umountall`(1M), `mnttab`(4), `vfstab`(4), `attributes`(5), `largefile`(5), `lofs`(7FS), `pcfs`(7FS)

NOTES

If the directory on which a file system is to be mounted is a symbolic link, the file system is mounted on the directory to which the symbolic link refers, rather than on top of the symbolic link itself.

BUGS

The `mount -p` output is incorrect for `cachefs`.

NAME

mountall, umountall – mount, unmount multiple file systems

SYNOPSIS

mountall [-F *FSType*] [-l | -r] [*file_system_table*]

umountall [-k] [-s] [-F *FSType*] [-l | -r]

umountall [-k] [-s] [-h *host*]

DESCRIPTION

mountall is used to mount file systems specified in a file system table. The file system table must be in vfstab(4) format. If no *file_system_table* is specified, /etc/vfstab is used. If – is specified as *file_system_table*, mountall reads the file system table from the standard input. mountall mounts only those file systems with the mount at boot field set to yes in the *file_system_table*.

For each file system in the file system table, the following logic is executed: if there exists a file/usr/lib/fs/*FSType*/fsckall, where *FSType* is the type of the file system, save that file system in a list to be passed later, and all at once, as arguments to the /usr/lib/fs/*FSType*/fsckall script. The /usr/lib/fs/*FSType*/fsckall script checks all of the file systems in its argument list to determine whether they can be safely mounted. If no /usr/lib/fs/*FSType*/fsckall script exists for the *FSType* of the file system, the file system is individually checked using fsck(1M). If the file system does not appear mountable, it is fixed using fsck before the mount is attempted. File systems with a – entry in the fsckdev field are mounted without first being checked.

umountall causes all mounted file systems except root, /usr, /var, /var/adm, /var/run, /proc, and /dev/fd to be unmounted. If the *FSType* is specified, mountall and umountall limit their actions to the *FSType* specified. There is no guarantee that umountall unmounts *busy* file systems, even if the -k option is specified.

OPTIONS

The following options are supported:

-F — Specify the *FSType* of the file system to be mounted or unmounted.

-h *host* — Unmount all file systems listed in /etc/mnttab that are remote-mounted from host.

-k — Use the fuser -k *mount-point* command. See the fuser(1M) for details. The -k option sends the SIGKILL signal to each process using the file. As this option spawns kills for each process, the kill messages might not show up immediately. There is no guarantee that umountall unmounts *busy* file systems, even if the -k option is specified.

-l — Limit the action to local file systems.

-r — Limit the action to remote file system types.

-s — Do not perform the umount operation in parallel.

FILES

/etc/mnttab — Mounted file system table

/etc/vfstab — Table of file system defaults

`/usr/lib/fs/`*FSType*`/fsckall`	Script called by `mountall` to perform the file system check of all file systems of type *FSType*

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`fsck`(1M), `fuser`(1M), `mount`(1M), `mnttab`(4), `vfstab`(4), `attributes`(5)

DIAGNOSTICS

No messages are printed if the file systems are mountable and clean.

Error and warning messages come from `fsck`(1M) and `mount`(1M).

NAME

mount_cachefs – mount CacheFS file systems

SYNOPSIS

mount -F cachefs [*generic_options*] -o backfstype=*file_system_type* [*specific_options*] [-O] *special mount_point*

DESCRIPTION

The CacheFS-specific version of the `mount` command mounts a cached file system; if necessary, it NFS-mounts its back file system. It also provides a number of CacheFS-specific options for controlling the caching process. For more information regarding back file systems, refer to the *System Administration Guide: Basic Administration*.

OPTIONS

To mount a CacheFS file system, use the generic `mount` command with the `-F` option followed by the argument `cachefs`.

See `mount`(1M) for a list of supported *generic_options*.

`-o` *specific_options*
Specify CacheFS file system specific options in a comma-separated list with no intervening spaces.

`acdirmax=`*n*
Specifies that cached attributes are held for no more than *n* seconds after directory update. After *n* seconds, all directory information is purged from the cache. The default value is `30` seconds.

`acdirmin=`*n*
Specifies that cached attributes are held for at least *n* seconds after directory update. After *n* seconds, CacheFS checks to see if the directory modification time on the back file system has changed. If it has, all information about the directory is purged from the cache and new data is retrieved from the back file system. The default value is `30` seconds.

`acregmax=`*n*
Specifies that cached attributes are held for no more than *n* seconds after file modification. After *n* seconds, all file information is purged from the cache. The default value is `30` seconds.

`acregmin=`*n*
Specifies that cached attributes are held for at least *n* seconds after file modification. After *n* seconds, CacheFS checks to see if the file modification time on the back file system has changed. If it has, all information about the file is purged from the cache and new data is retrieved from the back file system. The default value is `30` seconds.

`actimeo=`*n*
Sets `acregmin`, `acregmax`, `acdirmin`, and `acdirmax` to *n*.

`backfstype=`*file_system_type*
The file system type of the back file system (can be `nfs` or `hsfs`).

`backpath=`*path*
Specifies where the back file system is already mounted. If this argument is not supplied, CacheFS determines a mount point for the back file system. The back file system must be read-only.

`cachedir=`*directory*
The name of the cache directory.

`cacheid=`*ID*
ID is a string specifying a particular instance of a cache. If you do not specify a cache ID, CacheFS will construct one.

`demandconst`
Verifies cache consistency only when explicitly requested, rather than the periodic checking that is done by default. A consistency check is requested by using the `-s` option of the `cfsadmin`(1M) command. This option is useful for back file systems that change infrequently, for example, `/usr/openwin`. `demandconst` and `noconst` are mutually exclusive.

`local-access`
Causes the front file system to interpret the mode bits used for access checking instead of having the back file system verify access permissions. Do not use this argument with secure NFS.

`noconst`
Disables cache consistency checking. By default, periodic consistency checking is enabled. Specify `noconst` only when you know that the back file system will not be modified. Trying to perform cache consistency check using `cfsadmin -s` will result in error. `demandconst` and `noconst` are mutually exclusive.

`ro | rw`
Read-only or read-write (default).

`suid | nosuid`
Allow (default) or disallow `setuid` execution.

`write-around | non-shared`
Write modes for CacheFS. The `write-around` mode (the default) handles writes the same as NFS does; that is, writes are made to the back file system, and the affected file is purged from the cache. You can use the `non-shared` mode when you are sure that no one else will be writing to the cached file system. In this mode, all writes are made to both the front and the back file system, and the file remains in the cache.

`-O`
Overlay mount. Allows the filesystem to be mounted over an existing mount point, making the underlying filesystem inaccessible. If a mount is attempted on a pre-existing mount point without setting this flag, mount will fail with the error: `mount -F cachefs: mount failed Device busy`.

EXAMPLES

EXAMPLE 1 CacheFS-mounting a File System

The following example CacheFS-mounts the file system `server1:/user2`, which is already NFS-mounted on `/usr/abc` as `/xyz`.

```
example# mount -F cachefs -o backfstype=nfs,backpath=/usr/abc,
     cachedir=/cache1 server1:/user2 /xyz
```

The lines similar to the following appear in the `/etc/mnttab` file after the `mount` command is executed:

```
server1:/user2   /usr/abc        nfs
/usr/abc         /cache1/xyz    cachefs     backfstype=nfs
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`cfsadmin`(1M), `fsck_cachefs`(1M), `mount`(1M), `attributes`(5) *System Administration Guide: Basic Administration*

BUGS

The output for the *generic_option* `-p` output is incorrect for `cachefs`.

NAME

mountd – server for NFS mount requests and NFS access checks

SYNOPSIS

/usr/lib/nfs/mountd [-v] [-r]

DESCRIPTION

mountd is an RPC server that answers requests for NFS access information and file system mount requests. It reads the file /etc/dfs/sharetab to determine which file systems are available for mounting by which remote machines. See sharetab(4). nfsd running on the local server will contact mountd the first time an NFS client tries to access the file system to determine whether the client should get read-write, read-only, or no access. This access can be dependent on the security mode used in the remoted procedure call from the client. See share_nfs(1M).

The command also provides information as to what file systems are mounted by which clients. This information can be printed using the showmount(1M) command.

The mountd daemon is automatically invoked in run level 3.

Only super user can run the mountd daemon.

OPTIONS

The following options are supported:

-r Reject mount requests from clients. Clients that have file systems mounted will not be affected.

-v Run the command in verbose mode. Each time mountd determines what access a client should get, it will log the result to the console, as well as how it got that result.

FILES

/etc/dfs/sharetab shared file system table

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnfssu

SEE ALSO

nfsd(1M), share_nfs(1M), showmount(1M), sharetab(4), attributes(5)

NOTES

If nfsd is running, mountd must also be running in order to be assured that the NFS server can respond to requests, otherwise, the NFS service can hang.

Some routines that compare hostnames use case-sensitive string comparisons; some do not. If an incoming request fails, verify that the case of the hostname in the file to be parsed matches the case of the hostname called for, and attempt the request again.

NAME

mount_hsfs – mount hsfs file systems

SYNOPSIS

mount -F hsfs [*generic_options*] [-o *FSType-specific_options*] [-O] *special* | *mount_point*

mount -F hsfs [*generic_options*] [-o *FSType-specific_options*] [-O] *special* *mount_point*

DESCRIPTION

`mount` attaches a High Sierra file system (hsfs) to the file system hierarchy at the *mount_point*, which is the pathname of a directory. If *mount_point* has any contents prior to the `mount` operation, these are hidden until the file system is unmounted.

If `mount` is invoked with *special* or *mount_point* as the only arguments, `mount` will search `/etc/vfstab` to fill in the missing arguments, including the *FSType-specific_options*; see `mount`(1M) for more details.

If the file system being mounted contains `Rock Ridge` extensions, by default they will be used, enabling support of features not normally available under High Sierra file systems such as symbolic links, and special files.

OPTIONS

generic_options

See `mount`(1M) for the list of supported options.

`-o`

Specify `hsfs` file system specific options. If invalid options are specified, a warning message is printed and the invalid options are ignored. The following options are available:

`global | noglobal`

If `global` is specified and supported on the file system, and the system in question is part of a cluster, the file system will be globally visible on all nodes of the cluster. If `noglobal` is specified, the mount will not be globally visible. The default behavior is `noglobal`.

`ro`

Mount the file system read-only. This option is required.

`nrr`

no `Rock Ridge`: if `Rock Ridge` extensions are present in the file system, ignore them; interpret it as a regular High Sierra file system.

`notraildot`

File names on High Sierra file systems consist of a proper name and an extension separated by a '.' (dot) character. By default, the separating dot is always considered part of the file's name for all file access operations, even if there is no extension present. Specifying `notraildot` makes it optional to specify the trailing dot to access a file whose name lacks an extension.

Exceptions: This option is effective only on file systems for which `Rock Ridge` extensions are not active, either because they are not present on the CD-ROM, or they are explicitly ignored via the `nrr` option. If `Rock Ridge` extensions are active, `hsfs` quietly ignores this option.

`nomaplcase`
File names on High Sierra cdroms with no `Rock Ridge` extensions present should be uppercase characters only. By default, `hsfs` maps file names read from a `non-Rock Ridge` disk to all lowercase characters. `nomaplcase` turns off this mapping. The exceptions for `notraildot` discused above apply to `nomaplcase`.

`nosuid`
By default the file system is mounted with setuid execution allowed. Specifying `nosuid` causes the file system to be mounted with setuid execution disallowed.

`-O`
Overlay mount. Allow the file system to be mounted over an existing mount point, making the underlying file system inaccessible. If a mount is attempted on a pre-existing mount point without setting this flag, the mount will fail, producing the error `device busy`.

FILES

`/etc/mnttab` table of mounted file systems

`/etc/vfstab` list of default parameters for each file system

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`mount`(1M), `mountall`(1M), `mount`(2), `mnttab`(4), `vfstab`(4), `attributes` (5)

NOTES

If the directory on which a file system is to be mounted is a symbolic link, the file system is mounted on the directory to which the symbolic link refers, rather than on top of the symbolic link itself.

NAME

mount_nfs – mount remote NFS resources

SYNOPSIS

mount [-F nfs] [*generic_options*] [-o *specific_options*] [-O] *resource*

mount [-F nfs] [*generic_options*] [-o *specific_options*] [-O] *mount_point*

mount [-F nfs] [*generic_options*] [-o *specific_options*] [-O] *resource mount_point*

DESCRIPTION

The mount utility attaches a named *resource* to the file system hierarchy at the pathname location *mount_point*, which must already exist. If *mount_point* has any contents prior to the mount operation, the contents remain hidden until the *resource* is once again unmounted.

If the resource is listed in the /etc/vfstab file, the command line can specify either *resource* or *mount_point*, and mount consults /etc/vfstab for more information. If the -F option is omitted, mount takes the file system type from /etc/vfstab.

If the resource is not listed in the /etc/vfstab file, then the command line must specify both the *resource* and the *mount_point*.

host can be an IPv4 or IPv6 address string. As IPv6 addresses already contain colons, enclose *host* in a pair of square brackets when specifying an IPv6 address string. Otherwise the first occurrence of a colon can be interpreted as the separator between the host name and path, for example, [1080::8:800:200C:417A]:tmp/file. See inet(7P) and inet6(7P).

host:*pathname*
: Where *host* is the name of the NFS server host, and *pathname* is the path name of the directory on the server being mounted. The path name is interpreted according to the server's path name parsing rules and is not necessarily slash-separated, though on most servers, this is the case.

nfs://*host*[:*port*]/*pathname*
: This is an NFS URL and follows the standard convention for NFS URLs as described in *NFS URL Scheme*, RFC 2224. See the discussion of URL's and the public option under NFS FILE SYSTEMS for a more detailed discussion.

A comma-separated list of *host*:*pathname* and/or *nfs*://*host*[:*port*]/*pathname* resources
: See the discussion of replicated file systems and failover under NFS FILE SYSTEMS for a more detailed discussion.

A comma-separated list of hosts followed by a :*pathname* suffix
: See the discussion of replicated file systems and failover under NFS FILE SYSTEMS for a more detailed discussion.

The mount command maintains a table of mounted file systems in /etc/mnttab, described in mnttab(4).

OPTIONS

See mount(1M) for the list of supported *generic_options*.

-o *specific_options*
: Set file system specific options according to a comma-separated list with no intervening spaces.

`acdirmax=`*n*
Hold cached attributes for no more than *n* seconds after directory update. The default value is `60`.

`acdirmin=`*n*
Hold cached attributes for at least *n* seconds after directory update. The default value is `30`.

`acregmax=`*n*
Hold cached attributes for no more than *n* seconds after file modification. The default value is `60`.

`acregmin=`*n*
Hold cached attributes for at least *n* seconds after file modification. The default value is `3`.

`actimeo=`*n*
Set *min* and *max* times for regular files and directories to *n* seconds.

`bg | fg`
If the first attempt fails, retry in the background, or, in the foreground. The default is `fg`.

`forcedirectio | noforcedirectio`
If `forcedirectio` is specified, then for the duration of the mount, forced direct I/O is used. If the filesystem is mounted using `forcedirectio`, data is transferred directly between client and server, with no buffering on the client. If the filesystem is mounted using `noforcedirectio`, data is buffered on the client. `forcedirectio` is a performance option that is of benefit only in large sequential data transfers. The default behavior is `noforcedirectio`.

`grpid`
By default, the GID associated with a newly created file obeys the System V semantics; that is, the GID is set to the effective GID of the calling process. This behavior may be overridden on a per-directory basis by setting the set-GID bit of the parent directory; in this case, the GID of a newly created file is set to the GID of the parent directory (see `open`(2) and `mkdir`(2)). Files created on file systems that are mounted with the `grpid` option obeys BSD semantics independent of whether the set-GID bit of the parent directory is set; that is, the GID is unconditionally inherited from that of the parent directory.

`hard | soft`
Continue to retry requests until the server responds (`hard`) or give up and return an error (`soft`). The default value is `hard`.

`intr | nointr`
Allow (do not allow) keyboard interrupts to kill a process that is hung while waiting for a response on a hard-mounted file system. The default is `intr`, which makes it possible for clients to interrupt applications that may be waiting for a remote mount.

`noac`
Suppress data and attribute caching. The data caching that is suppressed is the write-behind. The local page cache is still maintained, but data copied into it is immediately written to the server.

`nocto`
Do not perform the normal close-to-open consistency. When a file is closed, all modified data associated with the file is flushed to the server and not held on the client. When a file is opened the client sends a request to the server to validate the client's local caches. This behavior ensures a file's consistency across multiple NFS clients. When `-nocto` is in effect, the client does not perform the flush on close and the request for validation, allowing the possiblity of differences among copies of the same file as stored on multiple clients.

This option can be used where it can be guaranteed that accesses to a specified file system are made from only one client and only that client. Under such a condition, the effect of `-nocto` can be a slight performance gain.

`port=`*n*
The server IP port number. The default is `NFS_PORT`. If the `port` option is specified, and if the resource includes one or more NFS URLs, and if any of the URLs include a `port` number, then the `port` number in the option and in the URL must be the same.

`posix`
Request POSIX.1 semantics for the file system. Requires a mount Version 2 `mountd`(1M) on the server. See `standards`(5) for information regarding POSIX.

`proto=`<*netid*>
<*netid*> is a value of network_id field from entry in the `/etc/netconfig` file. By default, the transport protocol used for the NFS mount is the first available connection oriented transport supported on both the client and the server. If no connection oriented transport is found, then the first available connectionless transport is used. This default behavior can be overridden with the `proto=`<*netid*> option.

`public`
The `public` option forces the use of the public file handle when connecting to the NFS server. The resource specified may or may not have an NFS URL. See the discussion of URL's and the public option under `NFS FILE SYSTEMS` for a more detailed discussion.

`quota | noquota`
Enable or prevent `quota`(1M) to check whether the user is over quota on this file system; if the file system has quotas enabled on the server, quotas are still checked for operations on this file system.

`remount`
Remounts a read-only file system as read-write (using the `rw` option). This option cannot be used with other `-o` options, and this option works only on currently mounted read-only file systems.

`retrans=`*n*
Set the number of NFS retransmissions to *n*. The default value is `5`. For connection-oriented transports, this option has no effect because it is assumed that the transport performs retransmissions on behalf of NFS.

`retry=`*n*
The number of times to retry the `mount` operation. The default for the `mount` command is `10000`.

The default for the automounter is `0`, in other words, do not retry. You might find it useful to increase this value on heavily loaded servers, where automounter traffic is dropped, causing unnecessary "server not responding" errors.

`ro | rw`
resource is mounted read-only or read-write. The default is `rw`.

`rsize=`*n*
Set the read buffer size to *n* bytes. The default value is `32768` when using Version 3 of the NFS protocol. The default can be negotiated down if the server prefers a smaller transfer size. When using Version 2, the default value is `8192`.

`sec=`*mode*
Set the security *mode* for NFS transactions. If `sec=` is not specified, then the default action is to use `AUTH_SYS` over NFS Version 2 mounts, or to negotiate a *mode* over NFS Version 3 mounts. NFS Version 3 mounts negotiate a security mode when the server returns an array of security modes. The client picks the first mode in the array that is supported on the client. Only one mode can be specified with the `sec=` option. See `nfssec`(5) for the available *mode* options.

`secure`
This option has been deprecated in favor of the `sec=`*dh* option.

`suid | nosuid`
Allow or disallow `setuid` execution. The default is `suid`.

`timeo=`*n*
Set the NFS timeout to *n* tenths of a second. The default value is `11` tenths of a second for connectionless transports, and `600` tenths of a second for connection-oriented transports.

`vers=`<*NFS version number*>
By default, the version of NFS protocol used between the client and the server is the highest one available on both systems. If the NFS server does not support NFS Version 3 protocol, then the NFS mount uses NFS Version 2 protocol.

`wsize=`*n*
Set the write buffer size to *n* bytes. The default value is `32768` when using Version 3 of the NFS protocol. The default can be negotiated down if the server prefers a smaller transfer size. When using Version 2, the default value is `8192`.

`xattr` | `noxattr`
Allow or disallow the creation and manipulation of extended attributes. The default is `xattr`. See `fsattr`(5) for a description of extended attributes.

`-O`
Overlay mount. Allow the file system to be mounted over an existing mount point, making the underlying file system inaccessible. If a mount is attempted on a pre-existing mount point without setting this flag, the mount fails, producing the error "device busy."

NFS FILE SYSTEMS

Background versus Foreground
File systems mounted with the `bg` option indicate that `mount` is to retry in the background if the server's mount daemon (`mountd`(1M)) does not respond. `mount` retries the request up to the count specified in the `retry=`*n* option. (Note that the default value for `retry` differs between `mount` and `automount`. See the description of `retry`, above.) Once the file system is mounted, each NFS request made in the kernel waits `timeo=`*n* tenths of a second for a response. If no response arrives, the time-out is multiplied by `2` and the request is retransmitted. When the number of retransmissions has reached the number specified in the `retrans=`*n* option, a file system mounted with the `soft` option returns an error on the request; one mounted with the `hard` option prints a warning message and continues to retry the request.

Hard versus Soft
File systems that are mounted read-write or that contain executable files should always be mounted with the `hard` option. Applications using `soft` mounted file systems may incur unexpected I/O errors, file corruption, and unexpected program core dumps. The soft option is not recommended.

Authenticated requests
The server may require authenticated NFS requests from the client. `sec=`*dh* authentication might be required. See `nfssec`(5).

URLs and the public option
If the `public` option is specified, or if the *resource* includes and NFS URL, `mount` attempts to connect to the server using the public file handle lookup protocol. See *WebNFS Client Specification*, RFC 2054. If the server supports the public file handle, the attempt is successful; `mount` does not need to contact the server's `rpcbind`(1M), and the `mountd`(1M) daemons to get the port number of the `mount` server and the initial file handle of *pathname*, respectively. If the NFS client and server are separated by a firewall that allows all outbound connections through specific ports, such as `NFS_PORT`, then this enables NFS operations through the firewall. The public option and the NFS URL can be specified independently or together. They interact as specified in the following matrix:

	resource style	
	host:*pathname*	**NFS URL**
public option	Force public file handle and fail `mount` if not supported. Use Native paths.	Force public file handle and fail `mount` if not supported. Use Canonical paths.
default	Use MOUNT protocol.	Try public file handle with Canonical paths. Fall back to MOUNT protocol if not supported.

A Native path is a path name that is interpreted according to conventions used on the native operating system of the NFS server. A Canonical path is a path name that is interpreted according to the URL rules. See *Uniform Resource Locators (URL)*, RFC 1738. See EXAMPLES for uses of Native and Canonical paths.

Replicated file systems and failover

resource can list multiple read–only file systems to be used to provide data. These file systems should contain equivalent directory structures and identical files. It is also recommended that they be created by a utility such as `rdist`(1). The file systems may be specified either with a comma–separated list of *host:/pathname* entries and/or NFS URL entries, or with a comma –separated list of hosts, if all file system names are the same. If multiple file systems are named and the first server in the list is down, failover uses the next alternate server to access files. If the read–only option is not chosen, replication is disabled. File access is blocked on the original if NFS locks are active for that file.

File Attributes

To improve NFS read performance, files and file attributes are cached. File modification times get updated whenever a write occurs. However, file access times may be temporarily out-of-date until the cache gets refreshed.

The attribute cache retains file attributes on the client. Attributes for a file are assigned a time to be flushed. If the file is modified before the flush time, then the flush time is extended by the time since the last modification (under the assumption that files that changed recently are likely to change soon). There is a minimum and maximum flush time extension for regular files and for directories. Setting `actimeo=`*n* sets flush time to *n* seconds for both regular files and directories.

Setting `actimeo=0` disables attribute caching on the client. This means that every reference to attributes is satisfied directly from the server though file data is still cached. While this guarantees that the client always has the latest file attributes from the server, it has an adverse effect on performance through additional latency, network load, and server load.

Setting the `noac` option also disables attribute caching, but has the further effect of disabling client write caching. While this guarantees that data written by an application is written directly to a server, where it can be viewed immediately by other clients, it has a significant adverse effect on client write performance. Data written into memory-mapped file pages (`mmap`(2)) are not written directly to this server.

EXAMPLES

EXAMPLE 1 Mounting an NFS File System

To mount an NFS file system:

```
example# mount serv:/usr/src /usr/src
```

EXAMPLE 2 Mounting An NFS File System Read-Only With No suid Privileges

To mount an NFS file system read-only with no suid privileges:

```
example# mount -r -o nosuid serv:/usr/src /usr/src
```

EXAMPLE 3 Mounting An NFS File System Over Version 2, with the UDP Transport

To mount an NFS file system over Version 2, with the UDP transport:

```
example# mount -o vers=2,proto=udp serv:/usr/src /usr/src
```

EXAMPLE 4 Mounting an NFS File System Using An NFS URL

To mount an NFS file system using an NFS URL (a canonical path):

```
example# mount nfs://serv/usr/man /usr/man
```

EXAMPLE 5 Mounting An NFS File System Forcing Use Of The Public File Handle

To mount an NFS file system and force the use of the public file handle and an NFS URL (a canonical path) that has a non 7–bit ASCII escape sequence:

```
example# mount -o public nfs://serv/usr/%A0abc /mnt/test
```

EXAMPLE 6 Mounting an NFS File System Using a Native Path

To mount an NFS file system using a native path (where the server uses colons (":") as the component separator) and the public file handle:

```
example# mount -o public serv:C:doc:new /usr/doc
```

EXAMPLE 7 Mounting a Replicated Set of NFS File Systems with the Same Pathnames

To mount a replicated set of NFS file systems with the same pathnames:

```
example# mount serv–a,serv–b,serv–c:/usr/man /usr/man
```

EXAMPLE 8 Mounting a Replicated Set of NFS File Systems with Different Pathnames

To mount a replicated set of NFS file systems with different pathnames:

```
example# mount serv–x:/usr/man,serv–y:/var/man,nfs://serv-z/man /usr/man
```

FILES

`/etc/mnttab`	table of mounted file systems
`/etc/dfs/fstypes`	default distributed file system type
`/etc/vfstab`	table of automatically mounted resources

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnfscu

SEE ALSO

`rdist`(1), `mountall`(1M), `mountd`(1M), `quota`(1M), `mkdir`(2), `mmap`(2), `mount`(2), `open`(2), `umount`(2), `mnttab`(4), `attributes`(5), `fsattr`(5), `nfssec`(5), `standards`(5), `inet`(7P), `inet6`(7P), `lofs`(7FS)

Callaghan, Brent, *WebNFS Client Specification*, RFC 2054, October 1996.

Callaghan, Brent, *NFS URL Scheme*, RFC 2224, October 1997.

Berners-Lee, Masinter & McCahill , *Uniform Resource Locators (URL)*, RFC 1738, December 1994.

NOTES

An NFS server should not attempt to mount its own file systems. See `lofs`(7FS).

If the directory on which a file system is to be mounted is a symbolic link, the file system is mounted on *the directory to which the symbolic link refers,* rather than being mounted on top of the symbolic link itself.

SunOS 4.x used the `biod` maintenance procedure to perform parallel read-ahead and write-behind on NFS clients. SunOS 5.x made `biod` obsolete with multi-threaded processing, which transparently performs parallel read-ahead and write-behind.

Since the root `(/)` file system is mounted read-only by the kernel during the boot process, only the `remount` option (and options that can be used in conjunction with `remount`) affect the root (`/`) entry in the `/etc/vfstab` file.

NAME

mount_pcfs – mount pcfs file systems

SYNOPSIS

mount -F pcfs [*generic_options*] [-o *FSType-specific_options*] *special* | *mount_point*

mount -F pcfs [*generic_options*] [-o *FSType-specific_options*] *special mount_point*

DESCRIPTION

`mount` attaches an MS-DOS file system (`pcfs`) to the file system hierarchy at the *mount_point*, which is the pathname of a directory. If *mount_point* has any contents prior to the `mount` operation, these are hidden until the file system is unmounted.

If `mount` is invoked with *special* or *mount_point* as the only arguments, `mount` will search `/etc/vfstab` to fill in the missing arguments, including the *FSType-specific_options*; see `mount`(1M) for more details.

The *special* argument can be one of two special device file types:

- A floppy disk, such as `/dev/diskette0` or `/dev/diskette1`.
- A DOS logical drive on a hard disk expressed as *device-name*`:`*logical-drive* , where *device-name* specifies the special block device-file for the whole disk and *logical-drive* is either a drive letter (c through z) or a drive number (1 through 24). Examples are `/dev/dsk/c0t0d0p0:`*c* and `/dev/dsk/c0t0d0p0:`*1*.

The *special* device file type must have a formatted MS-DOS file system with either a 12-bit, 16-bit, or 32-bit File Allocation Table.

OPTIONS

generic_options
See `mount`(1M) for the list of supported options.

`-o`
Specify `pcfs` file system specific options. The following options are available:

`rw|ro`
Mount the file system read/write or read-only. The default is `rw`.

`foldcase|nofoldcase`
Force uppercase characters in filenames to lowercase when reading them from the filesystem. This is for compatibility with the previous behavior of `pcfs`. The default is `nofoldcase`.

FILES

`/etc/mnttab` table of mounted file systems

`/etc/vfstab` list of default parameters for each file system

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWesu

SEE ALSO

`mount`(1M), `mountall`(1M), `mount`(2), `mnttab`(4), `vfstab`(4), `attributes` (5), `pcfs`(7FS)

NOTES If the directory on which a file system is to be mounted is a symbolic link, the file system is mounted on the directory to which the symbolic link refers, rather than on top of the symbolic link itself.

NAME

mount_tmpfs – mount tmpfs file systems

SYNOPSIS

mount [-F tmpfs] [-o *specific_options*] [-O] *special mount_point*

DESCRIPTION

tmpfs is a memory based file system which uses kernel resources relating to the VM system and page cache as a file system.

mount attaches a tmpfs file system to the file system hierarchy at the pathname location *mount_point*, which must already exist. If *mount_point* has any contents prior to the mount operation, these remain hidden until the file system is once again unmounted. The attributes (mode, owner, and group) of the root of the tmpfs filesystem are inherited from the underlying *mount_point*, provided that those attributes are determinable. If not, the root's attributes are set to their default values.

The *special* argument is usually specified as swap but is in fact disregarded and assumed to be the virtual memory resources within the system.

OPTIONS

-o *specific_options* Specify tmpfs file system specific options in a comma-separated list with no intervening spaces. If invalid options are specified, a warning message is printed and the invalid options are ignored. The following options are available:

size=*sz* The *sz* argument controls the size of this particular tmpfs file system. If the argument is has a 'k' suffix, the number will be interpreted as a number of kilobytes. An 'm' suffix will be interpreted as a number of megabytes. No suffix is interpreted as bytes. In all cases, the actual size of the file system is the number of bytes specified, rounded up to the physical pagesize of the system.

suid | nosuid Allow or disallow setuid execution. The default is suid.

xattr | noxattr Allow or disallow the creation and manipulation of extended attributes. The default is xattr. See fsattr(5) for a description of extended attributes.

-O Overlay mount. Allow the file system to be mounted over an existing mount point, making the underlying file system inaccessible. If a mount is attempted on a pre-existing mount point without setting this flag, the mount will fail, producing the errordevice busy.

FILES

/etc/mnttab table of mounted file systems

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO `mount`(1M), `mkdir`(2), `mount`(2), `open`(2), `umount`(2), `mnttab`(4), `attributes`(5), `fsattr`(5), `tmpfs`(7FS)

NOTES If the directory on which a file system is to be mounted is a symbolic link, the file system is mounted on the directory to which the symbolic link refers, rather than on top of the symbolic link itself.

NAME

mount_udfs – mount a udfs file system

SYNOPSIS

mount -F udfs [*generic_options*] [-o *specific_options*] [-O] *special mount_point*

mount -F udfs [*generic_options*] [-o *specific_options*] [-O] *special* | *mount_point*

DESCRIPTION

The `mount` utility attaches a `udfs` file system to the file system hierarchy at the *mount_point*, which is the pathname of a directory. If *mount_point* has any contents prior to the mount operation, these are hidden until the file system is unmounted.

If mount is invoked with either *special* or *mount_point* as the only arguments, `mount` searches `/etc/vfstab` to fill in the missing arguments, including the *specific_options*. See `mount`(1M).

If *special* and *mount_point* are specified without any *specific_options*, the default is `rw`.

If the directory on which a file system is to be mounted is a symbolic link, the file system is mounted on the directory to which the symbolic link refers, rather than on top of the symbolic link itself.

OPTIONS

See `mount`(1M) for the list of supported *generic_options*.

The following options are supported:

`-o` *specific_options*
: Specify `udfs` file system specific options in a comma-separated list with no intervening spaces. The following *specific_options* are available:

  m
  : Mount the file system without making an entry in `/etc/mnttab`.

  nosuid
  : Mount the file system with setuid execution disallowed. You can also use `nosuid` to disallow `setuid` when mounting devices.

    By default, the file system is mounted with `setuid` execution allowed.

  remount
  : Remount the file system as read-write. The option is used in conjunction with the `rw` option.

    A file system mounted read-only can be remounted as read-write. This option fails if the file system is not currently mounted.

  rw | ro
  : Read-write (`rw`) or read-only (`ro`). `rw` is the default.

`-O`
: Overlay mount. Allow the file system to be mounted over an existing mount point, making the underlying

file system inaccessible. If a mount is attempted on a pre-existing mount point without setting this flag, the mount fails, producing the error `device busy`.

FILES

`/etc/mnttab` Table of mounted file systems

`/etc/vfstab` List of default parameters for each file system

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWudf

SEE ALSO

`fsck`(1M), `fsck_udfs`(1M), `mount`(1M), `mountall`(1M), `mount`(2), `mnttab`(4), `vfstab`(4), `attributes`(5)

DIAGNOSTICS

not super user
The command is run by a non-root user. Run as root.

no such device
The device name specified does not exist.

not a directory
The specified mount point is not a directory.

is not an udfs file system
The device specified does not contain a udf 1.50 file system or the udfs file system module is not available.

is already mounted
The specified device is already in use.

not a block device
The device specified is not a block device. Use block device to mount.

write-protected
The device is read-only.

is corrupted. needs checking
The file system is in an inconsistent state. Run `fsck`.

NOTES

Copy-protected files can be stored on DVD-ROM media using Universal Disk Format (UDF). Reading these copy-protected files is not possible as this involves an authentication process. Unless an authentication process between the host and the drive is completed, reading these copy-protected files after mounting and before the authentication process, returns an error.

NAME

mount_ufs – mount ufs file systems

SYNOPSIS

mount -F ufs [*generic_options*] [-o *specific_options*] [-O] *special* | *mount_point*

mount -F ufs [*generic_options*] [-o *specific_options*] [-O] *special mount_point*

DESCRIPTION

The mount utility attaches a ufs file system to the file system hierarchy at the *mount_point*, which is the pathname of a directory. If *mount_point* has any contents prior to the mount operation, these are hidden until the file system is unmounted.

If mount is invoked with *special* or *mount_point* as the only arguments, mount will search /etc/vfstab to fill in the missing arguments, including the *specific_options*. See mount(1M).

If *special* and *mount_point* are specified without any *specific_options*, the default is rw.

If the directory on which a file system is to be mounted is a symbolic link, the file system is mounted on the directory to which the symbolic link refers, rather than on top of the symbolic link itself.

OPTIONS

See mount(1M) for the list of supported *generic_options*.

The following options are supported:

-o *specific_options*

Specify ufs file system specific options in a comma-separated list with no intervening spaces. If invalid options are specified, a warning message is printed and the invalid options are ignored. The following options are available:

noatime

By default, the file system is mounted with normal access time (atime) recording. If noatime is specified, the file system will ignore access time updates on files, except when they coincide with updates to the ctime or mtime. See stat(2). This option reduces disk activity on file systems where access times are unimportant (for example, a Usenet news spool).

noatime turns off access time recording regardless of dfratime or nodfratime.

dfratime | nodfratime

By default, writing access time updates to the disk may be deferred (dfratime) for the file system until the disk is accessed for a reason other than updating access times. nodfratime disables this behavior.

forcedirectio | noforcedirectio

If forcedirectio is specified and supported by the file system, then for the duration of the mount, forced direct I/O will be used. If the filesystem is mounted using forcedirectio, data is transferred directly between user address space and the disk. If the filesystem is mounted using noforcedirectio, data is buffered in kernel address space when data is

transferred between user address space and the disk. `forcedirectio` is a performance option that is of benefit only in large sequential data transfers. The default behavior is `noforcedirectio`.

`global | noglobal`
If `global` is specified and supported on the file system, and the system in question is part of a cluster, the file system will be globally visible on all nodes of the cluster. If `noglobal` is specified, the mount will not be globally visible. The default behavior is `noglobal`. The `global` option is mutually exclusive of the `nbmand` option, described below.

`intr | nointr`
Allow (do not allow) keyboard interrupts to kill a process that is waiting for an operation on a locked file system. The default is `intr`.

`largefiles | nolargefiles`
If `nolargefiles` is specified and supported by the file system, then for the duration of the mount it is guaranteed that all regular files in the file system have a size that will fit in the smallest object of type `off_t` supported by the system performing the mount. The mount will fail if there are any files in the file system not meeting this criterion. If `largefiles` is specified, there is no such guarantee. The default behavior is `largefiles`.

If `nolargefiles` is specified, `mount` will fail for `ufs` if the file system to be mounted has contained a large file (a file whose size is greater than or equal to 2 Gbyte) since the last invocation of `fsck` on the file system. The large file need not be present in the file system at the time of the mount for the mount to fail; it could have been created previously and destroyed. Invoking `fsck` (see `fsck_ufs`(1M)) on the file system will reset the file system state if no large files are present. After invoking `fsck`, a successful mount of the file system with `nolargefiles` specified indicates the absence of large files in the file system; an unsuccessful mount attempt indicates the presence of at least one large file.

`logging | nologging`
If `logging` is specified, then logging is enabled for the duration of the mounted file system. Logging is the process of storing transactions (changes that make up a complete UFS operation) in a log before the transactions are applied to the file system. Once a transaction is stored, the transaction can be applied to the file system later. This prevents file systems from becoming inconsistent, therefore eliminating the need to run `fsck`. And, because `fsck` can be bypassed, logging reduces the time required to reboot a system if it crashes, or after an unclean halt. The default behavior is `nologging`.

The log is allocated from free blocks on the file system, and is sized approximately 1 Mbyte per 1 Gbyte of file system, up to a maximum of 64 Mbytes. Logging can be enabled on any UFS, including root (`/`). The log created sby UFS logging is continually flushed as it fills up. The log is totally flushed when the file system is unmounted or as a result of the `lockfs -f` command.

`m`
Mount the file system without making an entry in `/etc/mnttab`.

`nbmand | nonbmand`
This option specifies that non-blocking mandatory locking semantics should be allowed on this file system. Non-blocking mandatory locking is disallowed by default. If the file system is mounted with the `nbmand` option, then applications can use the `fcntl`(2) interface to place non-blocking mandatory locks on files and the system will enforce those semantics. Enabling this option can cause standards conformant applications to see unexpected errors.

Do not use the `nbmand` option with `/`, `/var` and `/usr`.

The `remount` option should not be used to change the `nbmand` disposition of the file system. The `nbmand` option is mutually exclusive of the `global` option, described above.

`onerror` = *action*
This option specifies the action that UFS should take to recover from an internal inconsistency on a file system. Specify *action* as `panic`, `lock`, or `umount`. These values cause a forced system shutdown, a file system lock to be applied to the file system, or the file system to be forcibly unmounted, respectively. The default is `panic`.

`quota`
Quotas are turned on for the file system.

`remount`
Remounts a file system with a new set of options. All options not explicitly set with `remount` revert to their default values.

`rq`
Read-write with quotas turned on. Equivalent to `rw, quota`.

`ro | rw`
Read-only or read-write. Default is `rw`.

`suid | nosuid`
Allow or disallow `setuid/setgid` execution. The default is `suid`. This option also allows/disallows opening any device-special entries that appear within the filesystem.

This option is highly recommended whenever the file system is shared via NFS with the `root=` option, because, without it, NFS clients could add `setuid` programs to the server, or create devices that could open security holes.

`-O`
Overlay mount. Allow the file system to be mounted over an existing mount point, making the underlying file system inaccessible. If a mount is attempted on a pre-existing mount point without setting this flag, the mount will fail, producing the error “device busy”.

The `mount_ufs` command supports the `xattr` flag, to allow the creation and manipulation of extended attributes. See `fsattr`(5) for a description of extended attributes. The `xattr` flag is always on.

FILES

`/etc/mnttab` table of mounted file systems

`/etc/vfstab` list of default parameters for each file system

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`fsck`(1M), `fsck_ufs`(1M), `mount`(1M), `mountall`(1M), `fcntl`(2), `mount`(2), `stat`(2), `mnttab`(4), `vfstab`(4), `attributes`(5), `fsattr`(5), `largefile`(5)

NOTES

Since the root (`/`) file system is mounted read-only by the kernel during the boot process, only the `remount` option (and options that can be used in conjunction with `remount`) affect the root (`/`) entry in the `/etc/vfstab` file.

NAME

mount_xmemfs – mount xmemfs file systems

SYNOPSIS

mount `-F xmemfs [`*generic_options*`] -o[largebsize,]size=`*sz* `[-O]` *special mount_point*

DESCRIPTION

`xmemfs` is an extended memory file system which provides file system semantics to manage and access large amounts of physical memory which can exceed 4 GB in size.

`mount` attaches a `xmemfs` file system to the file system hierarchy at the pathname location `mount_point`, which must already exist. If `mount_point` has any contents prior to the mount operation, these remain hidden until the file system is once again unmounted. The attributes (`mode`, `owner`, and `group`) of the root of the `xmemfs` filesystem are inherited from the underlying `mount_point`, provided that those attributes are determinable. If not, the root's attributes are set to their default values.

The special argument is not currently used by `xmemfs` but a placeholder, (such as `xmem`), needs to be specified nevertheless.

OPTIONS

See `mount`(1M) for the list of supported *generic_options*.

`-o`*specific_options*

Specify `xmemfs` file system specific options in a comma-separated list with *no intervening spaces*. If invalid options are specified, a warning message is printed and the invalid options are ignored.

The `size=`*sz* *specific option* is required.

The following options are available:

size=*sz*

The *sz* argument specifies the desired size of this particular `xmemfs` file system. If the *sz* argument has a `k` suffix, the number is interpreted as kilobytes. An `m` suffix is interpreted as megabytes and `g` is interpreted as gigabytes. A *sz* specified with no suffix is interpreted as bytes.

In all cases, the actual size of the file system is the number of bytes specified, rounded up to the physical `pagesize` of the system or to the large page size if `largebsize` is specified.

This *specific_option* is required.

largebsize

If `largebsize` is specified, `xmemfs` uses the large memory page size as the file system block

size. On `IA32`, the large memory page size with `mmu36` which supports PAE (Physical Address Extension) is 2 MB. The large memory page size without `mmu36/PAE` is 4 MB. If there is no large page support, the file system block size is `PAGESIZE`.

`-O` Overlay mount. Allow the file system to be mounted over an existing mount point, making the underlying file system inaccessible. If a mount is attempted on a pre-existing mount point without setting this flag, the mount fails, producing the error `device busy`.

FILES

`/etc/mnttab` table of mounted file systems

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu
Architecture	i386
Interface Stability	Evolving

SEE ALSO

`mount`(1M), `mount`(2), `mkdir`(2), `open`(2), `umount`(2), `mnttab`(4), `attributes`(5),`xmemfs`(7FS)

NOTES

If the directory on which a file system is to be mounted is a symbolic link, the file system is mounted on the directory to which the symbolic link refers, rather than on top of the symbolic link itself.

The only file types allowed on xmemfs are directories and regular files. The execution of object files resident in xmemfs is not supported. Execution is prevented by not allowing users to set execute permissions on regular files.

NAME

mpstat – report per-processor or per-processor-set statistics

SYNOPSIS

/usr/bin/mpstat [-a] [-p | -P *set*] [*interval* [*count*]]

DESCRIPTION

`mpstat` reports processor statistics in tabular form. Each row of the table represents the activity of one processor. The first table summarizes all activity since boot; each subsequent table summarizes activity for the preceding interval. All values are rates (events per second) unless otherwise noted.

During execution of this kernel status command, the "state" of the kernel can change. An example would be CPUs going online or offline. `mpstat` reports this as `State change`.

`mpstat` reports the following information:

`CPU` or `SET`	Without the `-a` option, `mpstat` reports `CPU`, the processor ID. With the `-a` option, `mpstat` reports `SET`, the processor set ID.
`minf`	minor faults
`mjf`	major faults
`xcal`	inter-processor cross-calls
`intr`	interrupts
`ithr`	interrupts as threads (not counting clock interrupt)
`csw`	context switches
`icsw`	involuntary context switches
`migr`	thread migrations (to another processor)
`smtx`	spins on mutexes (lock not acquired on first try)
`srw`	spins on readers/writer locks (lock not acquired on first try)
`syscl`	system calls
`usr`	percent user time
`sys`	percent system time
`wt`	percent wait time
`idl`	percent idle time

With the `-a` option, `mpstat` also reports:

`sze`	number of processors in the set

With the `-p` option, `mpstat` also reports:

`set`	processor set membership of the CPU

OPTIONS

The following options are supported:

`-a`	Aggregate output by processor set. Sort the output by set. The default output is sorted by CPU number.
`-p`	Report processor set membership of each CPU. Sort the output by *set*. The default output is sorted by CPU number.
`-P` *set*	Display only those processors in the specified *set*.
interval	Report once each *interval* seconds.
count	Only print *count* reports.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`sar`(1), `iostat`(1M), `sar`(1M), `vmstat`(1M), `attributes`(5)

NOTES

The sum of CPU utilization might vary slightly from 100 because of rounding errors in the production of a percentage figure.

NAME

msgid – generate message IDs

SYNOPSIS

/usr/sbin/msgid

DESCRIPTION

The `msgid` utility generates message IDs.

A message ID is a numeric identifier that, with a high probability, uniquely identifies a message. The probability of two distinct messages having the same ID is about one in a million. Specifically, the message ID is a hash signature on the message's unexpanded format string, generated by `STRLOG_MAKE_MSGID()` as defined in `<sys/strlog.h>`.

`syslogd`(1M) is a simple filter that takes strings as input and produces those same strings, preceded by their message IDs, as output. Every message logged by `syslogd`(1M) includes the message ID. The message ID is intended to serve as a small, language-independent identifier.

EXAMPLES

EXAMPLE 1 Using the msgid command to generate a message ID

The following example uses the `msgid` command to generate a message ID for the `echo` command.

```
example# echo hello | msgid205790 hello
```

EXAMPLE 2 Using the msgid command to generate a message catalog

The following example uses the `msgid` command to enumerate all of the messages in the binary `ufs`, to generate a message catalog.

```
example# strings /kernel/fs/ufs | msgid137713 free:
    freeing free frag, dev:0x%lx, blk:%ld, cg:%d, ino:%lu, fs:%s
567420 ialloccg: block not in mapfs = %s
845546 alloc: %s: file system full
...
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`syslogd`(1M), `attributes`(5), `log`(7d)

NAME

mvdir – move a directory

SYNOPSIS

/usr/sbin/mvdir *dirname name*

DESCRIPTION

`mvdir` moves directories within a file system. `dirname` must be a directory. If *name* does not exist, it will be created as a directory. If *name* does exist, and is a directory, `dirname` will be created as *name/dirname*. `dirname` and *name* may not be on the same path; that is, one may not be subordinate to the other. For example:

```
example% mvdir x/y x/z
```

is legal, but

```
example% mvdir x/y x/y/z
```

is not.

OPERANDS

`dirname` The name of the directory that is to be moved to another directory in the filesystem.

name The name of the directory into which `dirname` is to be moved. If *name* does not exist, it will be created. It may not be on the same path as `dirname`.

USAGE

See `largefile`(5) for the description of the behavior of `mvdir` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXIT STATUS

`0` Successful operation.

`>0` Operation failed.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`mkdir`(1), `mv`(1), `attributes`(5), `largefile`(5)

NAME

named-bootconf – convert name server configuration files

SYNOPSIS

```
named-bootconf
```

DESCRIPTION

`named-bootconf` converts named configuration files from BIND 4 format to BIND 8 format.

Comments from the source file will not always appear at the appropriate place in the target file.

EXAMPLES

EXAMPLE 1 Using `named-bootconf`

The following command shows conversion of the `named.boot` file:

```
example# named-bootconf < named.boot > named.conf
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu
Interface Stability	Standard BIND 8.2.4

SEE ALSO

`in.named`(1M), `named.conf`(4), `attributes`(5)

NAME

named-xfer – ancillary agent for inbound zone transfers

SYNOPSIS

named-xfer `-z` *zone_to_transfer* `-f` *db_file* `-s` *serial_no* `[-d` *debuglevel*`]`
`[-l` *debug_log_file*`]` `[-t` *trace_file*`]` `[-p` *port#*`]` `[-S]` *nameserver...*

DESCRIPTION

The `named-xfer` program is an ancillary program executed by `in.named` to perform an inbound zone transfer. It is rarely executed directly, and only by system administrators who are trying to debug a zone transfer problem. See RFC's 1033, 1034, and 1035 for more information on the Internet name-domain system.

OPTIONS

`-z` Specifies the name of the zone to be transferred.

`-f` Specifies the name of the file into which the zone should be dumped when it is received from the primary server.

`-s` Specifies the serial number of the current copy of this zone. If the SOA RR from the primary server does not have a serial number higher than this, the transfer will be aborted.

`-d` Print debugging information. A number after the ''d'' determines the level of messages printed.

`-l` Specifies a log file for debugging messages. The default is system-dependent but is usually in */var/tmp* or */usr/tmp*. Note that this only applies if `-d` is also specified.

`-t` Specifies a trace file which will contain a protocol trace of the zone transfer. This is probably only of interest to those debugging the name server itself.

`-p` Use a different port number. The default is the standard port number as returned by `getservbyname`(3SOCKET) for service ''domain''.

`-S` Perform a restricted transfer of only the SOA, NS records and glue A records for the zone. The SOA record will not be loaded by named but will be used to determine when to verify the NS records. See the ''stubs'' directive in `in.named`(1M) for more information.

Additional arguments are taken as name server addresses in so-called ''dotted-quad'' syntax only; no host names are allowed. At least one address must be specified. If the first one fails to transfer successfully, the additional addresses will be tried in the order given.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWinamd

SEE ALSO

`in.named`(1M), `resolver`(3RESOLV), `resolv.conf`(4), `hostname`(1),

RFC 882

RFC 883

RFC 973

RFC 974

RFC 1033

RFC 1034

RFC 1035

RFC 1123

Name Server Operations Guide for BIND

NAME

ncaconfd – Solaris Network Cache and Accelerator (NCA) configuration daemon

SYNOPSIS

/usr/lib/inet/ncaconfd

DESCRIPTION

Use the `ncaconfd` utility to set up NCA on a system. At boot time, the `ncakmod` initialization script reads in `nca.if`(4) to determine on which interface(s) NCA should run. `ncaconfd` then sets up the interface.

`ncaconfd` also operates as a daemon if the `nca_active` key is set to enabled in `ncakmod.conf`(4) file. In this case, `ncaconfd` will continue as a daemon after all the NCA interfaces have been set up, listening for routing changes. The changes are then passed to NCA to control which interface NCA should use to make active outgoing TCP connnections.

FILES

`/etc/nca/ncakmod.conf`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWncau

SEE ALSO

`nca`(1), `ncakmod`(1), `nca.if`(4), `ncakmod.conf`(4), `attributes`(5)

NAME

ncheck – generate a list of path names versus i-numbers

SYNOPSIS

ncheck [-F *FSType*] [-V] [*generic_options*] [-o *FSType-specific_options*] [*special*...]

DESCRIPTION

ncheck with no options generates a path-name versus i-number list of all files on *special*. If *special* is not specified on the command line the list is generated for all *special*s in /etc/vfstab which have a numeric fsckpass. *special* is the raw device on which the file system exists.

OPTIONS

-F — Specify the *FSType* on which to operate. The *FSType* should either be specified here or be determinable from /etc/vfstab by finding an entry in the table that has a numeric fsckpass field and an fsckdev that matches *special*.

-V — Echo the complete command line, but do not execute the command. The command line is generated by using the options and arguments provided by the user and adding to them information derived from /etc/vfstab. This option may be used to verify and validate the command line.

generic_options — Options that are commonly supported by most FSType-specific command modules. The following options are available:

- -i *i-list* — Limit the report to the files on the *i-list* that follows. The *i-list* must be separated by commas with no intervening spaces.
- -a — Print the names "." and ". ." which are ordinarily suppressed.
- -s — Report only special files and files with set-user-ID mode. This option may be used to detect violations of security policy.

-o — Specify *FSType-specific_options* in a comma separated (without spaces) list of suboptions and keyword-attribute pairs for interpretation by the FSType-specific module of the command.

USAGE

See largefile(5) for the description of the behavior of ncheck when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

FILES

/etc/vfstab — list of default parameters for each file system

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO `vfstab`(4), `attributes`(5), `largefile`(5) Manual pages for the FSType-specific modules of `ncheck`

NOTES This command may not be supported for all *FSTypes*.

NAME

ncheck_ufs – generate pathnames versus i-numbers for ufs file systems

SYNOPSIS

ncheck `-F ufs` [*generic_options*] `[-o m]` [*special*...]

DESCRIPTION

`ncheck -F ufs` generates a pathname versus i-number list of files for the `ufs` file system residing on *special*. Names of directory files are followed by '`/.`'.

OPTIONS

See `ncheck`(1M) for the list of *generic_options* supported.

`-o` Specify `ufs` file system specific options. The available option is:

`m` Print mode information.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`ff`(1M), `ncheck`(1M), `attributes`(5)

DIAGNOSTICS

When the file system structure is improper, '`??`' denotes the "parent" of a parentless file and a pathname beginning with '`. . .`' denotes a loop.

NAME

ndc – name daemon control program

SYNOPSIS

ndc [-c *channel*] [-l *localsock*] [-p *pidfile*] [-d] [-q] [-s] [-t] [*command*]

DESCRIPTION

System administrators use the `ndc` utility to control the operation of a name server. If the system administrator fails to list a command, `ndc` will prompt for one until it reads EOF.

OPTIONS

The `ndc` command supports the following options:

`-c` *channel* — Specify the rendezvous point for the control channel. The default value for *channel* is `/var/run/ndc`, a UNIX domain socket that is also the server's default control channel. If the desired control channel is a TCP/IP socket, then the format for the *channel* argument is *ipaddr/port*. For example, a value of `127.0.0.1/54` would be TCP port 54 on the local host.

`-d` — Turn on debugging mode. This option is mainly of interest to developers.

`-l` *localsock* — Bind the client side of the control channel to a specific address. Servers can be configured to reject connections that do not come from specific addresses. If the desired control channel is a TCP/IP socket, then the format for the *localsock* argument is *ipaddr/port*.

`-p` *pidfile* — Use for backwards compatibility with older name servers. It enables `ndc` to use UNIX signals for control communications. Optional with modern name servers, this capability may not be supported in future releases.The command set that is available is narrower when the signal interface is used. A likely value for the *pidfile* argument is `/var/run/named.pid`.

`-q` — Suppress prompt and result text.

`-s` — Suppress non-fatal error announcements.

`-t` — Turn on protocol and system tracing. Use this option in installation debugging.

COMMANDS

The following commands are built into the `ndc` utility. The full set of commands that the name server supports is dynamic. Use the `help` command for information on the available commands .

`/help` — Show help information for built in commands.

`/exit` — Exit from `ndc` command interpreter.

`/trace` — Toggle protocol and system tracing on and off. See `-t`.

`/debug` — Toggle debugging mode on and off. See `-d`.

`/quiet` — Toggle prompt and result information on and off. See `-q`.

`/silent` Toggle announcement of non-fatal errors on and off. See `-s`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu
Interface Stability	Standard, BIND 8.2.4

SEE ALSO

`in.named`(1M), `execvp`(2), `attributes`(5)

NOTES

When the `ndc` utility is running in *pidfile* mode, pass any arguments to start and restart commands to the new name server on the command line. If the `ndc` utility is running in *channel* mode, there is no start command, and the restart command just tells the name server to `execvp`(2) itself.

NAME

ndd – get and set driver configuration parameters

SYNOPSIS

ndd [-set] *driver parameter* [*value*]

DESCRIPTION

ndd gets and sets selected configuration parameters in some kernel drivers. Currently, ndd only supports the drivers that implement the TCP/IP Internet protocol family. Each driver chooses which parameters to make visible using ndd. Since these parameters are usually tightly coupled to the implementation, they are likely to change from release to release. Some parameters may be read-only.

If the -set option is omitted, ndd queries the named *driver*, retrieves the value associated with the specified *parameter*, and prints it. If the -set option is given, ndd passes *value*, which must be specified, down to the named *driver* which assigns it to the named *parameter*.

By convention, drivers that support ndd also support a special read-only *parameter* named "?" which can be used to list the parameters supported by the driver.

EXAMPLES

EXAMPLE 1 Getting Parameters Supported By The TCP Driver

To see which parameters are supported by the TCP driver, use the following command:

```
example% ndd /dev/tcp \?
```

The parameter name "?" may need to be escaped with a backslash to prevent its being interpreted as a shell meta character.

The following command sets the value of the parameter *ip_forwarding* in the dual stack IP driver to zero. This disables IPv4 packet forwarding.

```
example% ndd -set /dev/ip ip_forwarding 0
```

Similarly, in order to disable IPv6 packet forwarding, the value of parameter *ip6_forwarding*

```
example% ndd -set /dev/ip ip6_forwarding 0
```

To view the current IPv4 forwarding table, use the following command:

```
example% ndd /dev/ip ipv4_ire_status
```

To view the current IPv6 forwarding table, use the following command:

```
example% ndd /dev/ip ipv6_ire_status
```

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO | `nca`(1), `ioctl`(2), `attributes`(5), `arp`(7P), `ip`(7P), `ip6`(7P), `tcp`(7P), `udp`(7P)

NOTES | The parameters supported by each driver may change from release to release. Like programs that read `/dev/kmem`, user programs or shell scripts that execute `ndd` should be prepared for parameter names to change.

The `ioctl()` command that `ndd` uses to communicate with drivers is likely to change in a future release. User programs should avoid making dependencies on it.

The meanings of many `ndd` parameters make sense only if you understand how the driver is implemented.

NAME

netstat – show network status

SYNOPSIS

netstat [-anv] [-f *address_family*] [-P *protocol*]

netstat -g [-n] [-f *address_family*]

netstat -p [-n] [-f *address_family*]

netstat -s [-f *address_family*] [-P *protocol*]

netstat -m [-v]

netstat -i [-I *interface*] [-an] [-f *address_family*] [*interval* [*count*]]

netstat -r [-anv] [-f *address_family* | *filter*]

netstat -M [-ns] [-f *address_family*]

netstat -D [-I *interface*] [-f *address_family*]

DESCRIPTION

The `netstat` command displays the contents of certain network-related data structures in various formats, depending on the options you select.

The `netstat` command has the several forms shown in the SYNOPSIS section, above, listed as follows:

- The first form of the command (with no required arguments) displays a list of active sockets for each protocol.
- The second, third, and fourth forms (`-g`, `-p`, and `-s` options) display information from various network data structures.
- The fifth form (`-m` option) displays STREAMS memory statistics.
- The sixth form (`-i` option) shows the state of the interfaces.
- The seventh form (`-r` option) displays the routing table.
- The eighth form (`-M` option) displays the multicast routing table.
- The ninth form (`-D` option) displays the state of DHCP on one or all interfaces.

These forms are described in greater detail below.

With no arguments (the first form), `netstat` displays connected sockets for `PF_INET`, `PF_INET6`, and `PF_UNIX`, unless modified otherwise by the `-f` option.

OPTIONS

`-a` — Show the state of all sockets, all routing table entries, or all interfaces, both physical and logical. Normally, listener sockets used by server processes are not shown. Under most conditions, only interface, host, network, and default routes are shown and only the status of physical interfaces is shown.

`-f` *address_family* — Limit all displays to those of the specified *address_family*. The value of *address_family* can be one of the following:

`inet` For the `AF_INET` address family showing IPv4 information.

`inet6` For the `AF_INET6` address family showing IPv6 information.

`unix` For the `AF_UNIX` address family.

`-f` *filter* With `-r` only, limit the display of routes to those matching the specified filter. A filter rule consists of a "keyword:value" pair. The known keywords and the value syntax are:

`af:{inet|inet6|unix|`*number*`}`
Selects an address family. This is identical to `-f` *address_family* and both syntaxes are supported.

`{inif|outif}:{`*name*`|`*ifIndex*`|any|none}`
Selects an input or output interface. You can specify the interface by name (such as `hme0`) or by `ifIndex` number (for example, `2`). If `any` is used, the filter matches all routes having a specified interface (anything other than null). If `none` is used, the filter matches all routes having a null interface. Note that you can view the index number (*ifIndex*) for an interface with the `-a` option of `ifconfig`(1M).

`{src|dst}:{`*ip-address*`[/`*mask*`]|any|none}`
Selects a source or destination IP address. If specified with a mask length, then any routes with matching or longer (more specific) masks are selected. If `any` is used, then all but addresses but 0 are selected. If `none` is used, then address 0 is selected.

`flags:`[+ -]?[ABDGHLU]+
Selects routes tagged with the specified flags. By default, the flags as specified must be set in order to match. With a leading `+`, the flags specified must be set but others are ignored. With a leading `-`, the flags specified must not be set and others are permitted.

You can specify multiple instances of `-f` to specify multiple filters. For example:

```
% netstat -nr -f outif:hme0 -f outif:hme1 -f dst:10.0.0.0/8
```

The preceding command displays routes within network 10.0.0.0/8, with mask length 8 or greater, and an output interface of either `hme0` or `hme1`, and excludes all other routes.

	`-g`	Show the multicast group memberships for all interfaces. See DISPLAYS, below.
	`-i`	Show the state of the interfaces that are used for IP traffic. Normally this shows statistics for the physical interfaces. When combined with the `-a` option, this will also report information for the logical interfaces. See `ifconfig`(1M).
	`-m`	Show the STREAMS memory statistics.
	`-n`	Show network addresses as numbers. `netstat` normally displays addresses as symbols. This option may be used with any of the display formats.
	`-p`	Show the net to media tables. See DISPLAYS, below.
	`-r`	Show the routing tables. Normally, only interface, host, network, and default routes are shown, but when this option is combined with the `-a` option, all routes will be displayed, including cache.
	`-s`	Show per-protocol statistics. When used with the `-M` option, show multicast routing statistics instead. When used with the `-a` option, per-interface statistics will be displayed, when available, in addition to statistics global to the system. See DISPLAYS, below.
	`-v`	Verbose. Show additional information for the sockets, STREAMS memory statistics, and the routing table.
	`-I` *interface*	Show the state of a particular interface. *interface* can be any valid interface such as `hme0` or `le0`. Normally, the status and statistics for physical interfaces are displayed. When this option is combined with the `-a` option, information for the logical interfaces is also reported.
	`-M`	Show the multicast routing tables. When used with the `-s` option, show multicast routing statistics instead.
	`-P` *protocol*	Limit display of statistics or state of all sockets to those applicable to *protocol*. The protocol can be one of `ip`, `ipv6`, `icmp`, `icmpv6`, `igmp`, `udp`, `tcp`, `rawip`. `rawip` can also be specified as `raw`. The command accepts protocol options only as all lowercase.
	`-D`	Show the status of DHCP configured interfaces.
OPERANDS	*interval*	Display interface statistics accumulated since last display every *interval* seconds, repeating forever, unless *count* is specified. When invoked with *interval*, the first row of `netstat` output shows statistics accumulated since last reboot.

count Display interface statistics the number of times specified by *count*, at the interval specified by *interval*.

Active Sockets (First Form)

The display for each active socket shows the local and remote address, the send and receive queue sizes (in bytes), the send and receive windows (in bytes), and the internal state of the protocol.

The symbolic format normally used to display socket addresses is either

`hostname`.*port* when the name of the host is specified, or

network.*port* if a socket address specifies a network but no specific host.

The numeric host address or network number associated with the socket is used to look up the corresponding symbolic hostname or network name in the *hosts* or *networks* database.

If the network or hostname for an address is not known, or if the `-n` option is specified, the numerical network address is shown. Unspecified, or "wildcard", addresses and ports appear as "*". For more information regarding the Internet naming conventions, refer to `inet`(7P) and `inet6`(7P).

TCP Sockets

The possible state values for TCP sockets are as follows:

`BOUND`	Bound, ready to connect or listen.
`CLOSED`	Closed. The socket is not being used.
`CLOSING`	Closed, then remote shutdown; awaiting acknowledgment.
`CLOSE_WAIT`	Remote shutdown; waiting for the socket to close.
`ESTABLISHED`	Connection has been established.
`FIN_WAIT_1`	Socket closed; shutting down connection.
`FIN_WAIT_2`	Socket closed; waiting for shutdown from remote.
`IDLE`	Idle, opened but not bound.
`LAST_ACK`	Remote shutdown, then closed; awaiting acknowledgment.
`LISTEN`	Listening for incoming connections.
`SYN_RECEIVED`	Initial synchronization of the connection under way.
`SYN_SENT`	Actively trying to establish connection.
`TIME_WAIT`	Wait after close for remote shutdown retransmission.

Network Data Structures (Second Through Fifth Forms)

The form of the display depends upon which of the `-g`, `-m`, `-p`, or `-s` options you select.

`-g`	Displays the list of multicast group membership.
`-m`	Displays the memory usage, for example, STREAMS mblks.

`-p` Displays the net to media mapping table. For IPv4, the address resolution table is displayed. See `arp`(1M). For IPv6, the neighbor cache is displayed.

`-s` Displays the statistics for the various protocol layers.

The statistics use the MIB specified variables. The defined values for `ipForwarding` are:

`forwarding(1)` Acting as a gateway.

`not-forwarding(2)` Not acting as a gateway.

The IPv6 and ICMPv6 protocol layers maintain per-interface statistics. If the `-a` option is specified with the `-s` option, then the per-interface statistics as well as the total sums are displayed. Otherwise, just the sum of the statistics are shown.

For the second, third, and fourth forms of the command, you must specify at least `-g`, `-p`, or `-s`. You can specify any combination of these options. You can also specify `-m` (the fifth form) with any set of the `-g`, `-p`, and `-s` options. If you specify more than one of these options, `netstat` displays the information for each one of them.

Interface Status (Sixth Form)

The interface status display lists information for all current interfaces, one interface per line. If an interface is specified using the `-I` option, it displays information for only the specified interface.

The list consists of the interface name, `mtu` (maximum transmission unit, or maximum packet size)(see `ifconfig`(1M)), the network to which the interface is attached, addresses for each interface, and counter associated with the interface. The counters show the number of input packets, input errors, output packets, output errors, and collisions, respectively. For Point-to-Point interfaces, the Net/Dest field is the name or address on the other side of the link.

If the `-a` option is specified with either the `-i` option or the `-I` option, then the output includes names of the physical interface(s), counts for input packets and output packets for each logical interface, plus additional information.

If the `-n` option is specified, the list displays the IP address instead of the interface name.

If an optional *interval* is specified, the output will be continually displayed in *interval* seconds until interrupted by the user or until *count* is reached. See OPERANDS.

The physical interface is specified using the `-I` option. When used with the *interval* operand, output for the `-I` option has the following format:

```
input     le0          output        input          (Total)    output
packets  errs  packets errs  colls   packets  errs  packets   errs   colls
227681   0     659471  1     502     261331   0     99597     1      502
10       0     0       0     0       10       0     0         0      0
8        0     0       0     0       8        0     0         0      0
10       0     2       0     0       10       0     2         0      0
```

If the input interface is not specified, the first interface of address family `inet` or `inet6` will be displayed.

Routing Table (Seventh Form)

The routing table display lists the available routes and the status of each. Each route consists of a destination host or network, and a gateway to use in forwarding packets. The *flags* column shows the status of the route (`U` if "up"), whether the route is to a gateway (`G`), and whether the route was created dynamically by a redirect (`D`). If the `-a` option is specified, there will be routing entries with flags for combined routing and address resolution entries (`A`), broadcast addresses (`B`), and the local addresses for the host (`L`).

Interface routes are created for each interface attached to the local host; the gateway field for such entries shows the address of the outgoing interface.

The `use` column displays the number of packets sent using a combined routing and address resolution (`A`) or a broadcast (`B`) route. For a local (`L`) route, this count is the number of packets received, and for all other routes it is the number of times the routing entry has been used to create a new combined route and address resolution entry.

The *interface* entry indicates the network interface utilized for the route.

Multicast Routing Tables (Eighth Form)

The multicast routing table consists of the virtual interface table and the actual routing table.

DHCP Interface Information (Ninth Form)

The DHCP interface information consists of the interface name, its current state, lease information, packet counts, and a list of flags.

The states correlate with the specifications set forth in *RFC 2131*.

Lease information includes:

- when the lease began;
- when lease renewal will begin; and
- when the lease will expire.

The flags currently defined include:

`BOOTP` The interface has a lease obtained through `BOOTP`.

`BUSY` The interface is busy with a DHCP transaction.

`PRIMARY` The interface is the primary interface. See `dhcpinfo`(1).

`FAILED` The interface is in failure state and must be manually restarted.

Packet counts are maintained for the number of packets sent, the number of packets received, and the number of lease offers declined by the DHCP client. All three counters are initialized to zero and then incremented while obtaining a lease. The

counters are reset when the period of lease renewal begins for the interface. Thus, the counters represent either the number of packets sent, received, and declined while obtaining the current lease, or the number of packets sent, received, and declined while attempting to obtain a future lease.

FILES

`/etc/default/inet_type` `DEFAULT_IP` setting

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`arp`(1M), `dhcpinfo`(1), `dhcpagent`(1M), `ifconfig`(1M), `iostat`(1M), `mibiisa`(1M), `savecore`(1M), `vmstat`(1M), `hosts`(4), `inet_type`(4), `networks`(4), `protocols`(4), `services`(4), `attributes`(5), `inet`(7P), `inet6`(7P)

Droms, R., *RFC 2131, Dynamic Host Configuration Protocol*, Network Working Group, March 1997.

NOTES

When displaying interface information, `netstat` honors the `DEFAULT_IP` setting in `/etc/default/inet_type`. If it is set to `IP_VERSION4`, then `netstat` will omit information relating to IPv6 interfaces, statistics, connections, routes and the like.

However, you can override the `DEFAULT_IP` setting in `/etc/default/inet_type` on the command-line. For example, if you have used the command-line to explicitly request IPv6 information by using the `inet6` address family or one of the IPv6 protocols, it will override the `DEFAULT_IP` setting.

If you need to examine network status information following a kernel crash, use the `mdb`(1) utility on the `savecore`(1M) output.

NAME

newaliases – rebuild the data base for the mail aliases file

SYNOPSIS

newaliases

DESCRIPTION

`newaliases` rebuilds the random access data base for the mail aliases file `/etc/mail/aliases`.

`newaliases` accepts all the flags that `sendmail`(1M) accepts. However, most of these flags have no effect, except for the `-C` option and three of the `Processing Options` that can be set from a configuration file with the `-o` option:

`-C` */path/to/alt/config/file*	Use alternate configuration file.
`-oA`*file*	Specify possible alias files.
`-oL`*n*	Set the default log level to *n*. Defaults to `9`.
`-on`	Validate the RHS of aliases when rebuilding the `aliases`(4) database.

`newaliases` runs in verbose mode (`-v` option) automatically.

EXAMPLES

EXAMPLE 1 Running the `newaliases` Command

The following command runs `newaliases` on an alias file different from the `/etc/mail/aliases` default in `sendmail`(1M):

```
example% newaliases -oA/path/to/alternate/alias/file
```

EXIT STATUS

`newaliases` returns an exit status describing what it did. The codes are defined in `/usr/include/sysexits.h`.

`EX_OK`	Successful completion on all addresses.
`EX_NOUSER`	User name not recognized.
`EX_UNAVAILABLE`	Catchall. Necessary resources were not available.
`EX_SYNTAX`	Syntax error in address.
`EX_SOFTWARE`	Internal software error, including bad arguments.
`EX_OSERR`	Temporary operating system error, such as “cannot fork”.
`EX_NOHOST`	Host name not recognized.
`EX_TEMPFAIL`	Message could not be sent immediately, but was queued.

FILES

`/etc/aliases`	Symbolic link to `/etc/mail/aliases`
`/etc/mail/aliases.pag`	
`/etc/mail/aliases.dir`	`ndbm` files maintained by `newaliases`

`/etc/mail/aliases.db`	Berkeley DataBase file maintained by newaliases

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsndmu

SEE ALSO

`sendmail`(1M), `aliases`(4), `attributes`(5)

NAME

newfs – construct a new UFS file system

SYNOPSIS

newfs [-Nv] [*mkfs-options*] *raw-device*

DESCRIPTION

newfs is a "friendly" front-end to the mkfs(1M) program for making UFS file systems on disk partitions. newfs calculates the appropriate parameters to use and calls mkfs.

If run interactively (that is, standard input is a tty), newfs will prompt for confirmation before making the file system.

If the -N option is not specified and the inodes of the device are not randomized, newfs will call fsirand(1M).

You must be super-user to use this command, except when creating a UFS file system on a *diskette* (see EXAMPLES).

OPTIONS

The following options are supported:

-N — Print out the file system parameters that would be used in creating the file system without actually creating the file system. fsirand(1M) is not called here.

-v — Verbose. newfs prints out its actions, including the parameters passed to mkfs.

mkfs-options — Options that override the default parameters are:

-a *apc* — The number of alternate sectors per cylinder (SCSI devices only) to reserve for bad block replacement. The default is 0.

-b *bsize* — The logical block size of the file system in bytes (either 4096 or 8192). The default is 8192. The sun4u architecture does not support the 4096 block size.

-c *cgsize* — The number of cylinders per cylinder group (ranging from 16 to 256). The default is calculated by dividing the number of sectors in the file system by the number of sectors in a gigabyte, and then multiplying the result by 32. The default value will always be between 16 and 256. mkfs may override this value. See mkfs_ufs(1M) for details.

-C *maxcontig* — The maximum number of logical blocks, belonging to one file, that will be allocated contiguously before inserting a rotational delay. The default is determined from the disk drive's maximum transfer rate.

UFS supports no more than 1048576 byte (1MB) runs. With a logical block size of 4096, this gives a *maxcontig* limit of 256. With 8192, the limit is 128.

This parameter can be subsequently changed using the tunefs(1M) command.

This parameter also controls clustering. Regardless of the value of *gap*, clustering is enabled only when *maxcontig* is greater than 1. Clustering allows higher I/O rates for sequential I/O and is described in tunefs(1M).

`-d` *gap*

Rotational delay. The expected time (in milliseconds) to service a transfer completion interrupt and initiate a new transfer on the same disk. It is used to decide how much rotational spacing to place between successive blocks in a file. This parameter can be subsequently changed using the tunefs(1M) command. The default is 0.

`-f` *fragsize*

The smallest amount of disk space in bytes to allocate to a file. The smallest amout of disk space in bytes to allocate to a file. If the logical block size is 4096, legal values are 512, 1024, 2048, and 4096. When the logical block size is 8192, legal values are 1024, 2048, 4096, and 8192. The default value is 1024.

`-i` *nbpi*

The number of bytes per inode. This specifies the density of inodes in the file system. The number is divided into the total size of the file system to determine the fixed number of inodes to create. It should reflect the expected average size of files in the file system. If fewer inodes are desired, a larger number should be used; to create more inodes a smaller number should be given. The default for *nbpi* is as follows:.

Disk size	*Density*
–1GB	2048
–2GB	4096
–3GB	6144
3GB–	8192

`-m` *free*	The minimum percentage of free space to maintain in the file system (between 1% and 99%, inclusively). This space is off-limits to normal users. Once the file system is filled to this threshold, only the super-user can continue writing to the file system. This parameter can be subsequently changed using the `tunefs`(1M) command. The default is ((64 Mbytes/partition size) * 100), rounded down to the nearest integer and limited between 1% and 10%, inclusively.
`-n` *nrpos*	The number of different rotational positions in which to divide a cylinder group. The default is `8`.
`-o` *opt*	(`space` or `time`). The file system can either be instructed to try to minimize the time spent allocating blocks, or to try to minimize the space fragmentation on the disk. The default is `time`.
`-r` *rpm*	The speed of the disk in revolutions per minute. The default is driver- or device-specific.
`-s` *size*	The size of the file system in sectors. The default is to use the entire partition.
`-t` *ntrack*	The number of tracks per cylinder on the disk. The default is taken from the disk label.

OPERANDS

The following operands are supported:

raw-device	The name of a raw special device residing in `/dev/rdsk` (for example, `/dev/rdsk/c0t0d0s6`) on which to create the file system.

USAGE

See `largefile`(5) for the description of the behavior of `newfs` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXAMPLES

EXAMPLE 1 Verbosely displaying the parameters for the raw special device.

The following example verbosely displays the parameters for the raw special device, `c0t0d0s6`, but does not actually create a new file system:

```
example# newfs -Nv /dev/rdsk/c0t0d0s6
mkfs -F ufs -o N /dev/rdsk/c0t0d0s6 1112940 54 15 8192 1024 16 10 60
     2048 t 0 –1 8 /dev/rdsk/c0t0d0s6:     1112940 sectors in
     1374 cylinders of 15 tracks, 54 sectors 569.8MB in 86 cyl
     groups (16 c/g, 6.64MB/g, 3072 i/g) super-block backups
```

EXAMPLE 1 Verbosely displaying the parameters for the raw special device. *(Continued)*

```
    (for fsck -b #) at:
    32, 13056, 26080, 39104, 52128, 65152, 78176, 91200, 104224, . . .
```

EXAMPLE 2 Creating a UFS file system.

The following example uses the command to create a UFS file system on a diskette that is managed by Volume Manager.

```
example%  newfs /vol/dev/aliases/floppy0
newfs: construct a new file system /vol/dev/aliases/floppy0: (y/n)? y
/vol/dev/aliases/floppy0:    2880 sectors in 80 cylinders of 2 tracks,
    18 sectors   1.4MB in 5 cyl groups (16 c/g, 0.28MB/g, 128 i/g)
        super-block backups (for fsck -F ufs -o b=#) at:
     32, 640, 1184, 1792, 2336, . . .
```

EXIT STATUS

The following exit values are returned:

`0` The operation was successful.

`1, 10` Usage error or internal error. A message is output to `STDERR` explaining the error.

Other exit values may be returned by `mkfs`(1M), which is called by `newfs`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`fsck`(1M), `fsck_ufs`(1M), `fsirand`(1M), `mkfs`(1M), `mkfs_ufs`(1M), `tunefs`(1M), `fs_ufs`(4), `attributes`(5), `largefile`(5)

DIAGNOSTICS

`newfs: No such file or directory`
The device specified does not exist, or a disk partition was not specified.

special `: cannot open`
You must be super-user to use this command.

NAME

newkey – create a new Diffie-Hellman key pair in the publickey database

SYNOPSIS

newkey -h *hostname* [-s nisplus | nis | files | ldap]

newkey -u *username* [-s nisplus | nis | files | ldap]

DESCRIPTION

newkey establishes new public keys for users and machines on the network. These keys are needed when using secure RPC or secure NFS service.

newkey prompts for a password for the given *username* or hostname and then creates a new public/secret Diffie-Hellman 192 bit key pair for the user or host. The secret key is encrypted with the given password. The key pair can be stored in the /etc/publickey file, the NIS publickey map, or the NIS+ cred.org_dir table.

newkey consults the publickey entry in the name service switch configuration file (see nsswitch.conf(4)) to determine which naming service is used to store the secure RPC keys. If the publickey entry specifies a unique name service, newkey will add the key in the specified name service. However, if there are multiple name services listed, newkey cannot decide which source to update and will display an error message. The user is required to specify the source explicitly with the -s option.

In the case of NIS, newkey should be run by the superuser on the master NIS server for that domain. In the case of NIS+, newkey should be run by the superuser on a machine which has permission to update the cred.org_dir table of the new user/host domain.

In the case of NIS+, nisaddcred(1M) should be used to add new keys. newkey cannot be used to create keys other than 192-bit Diffie-Hellman.

OPTIONS

-h *hostname* — Create a new public/secret key pair for the privileged user at the given hostname. Prompts for a password for the given hostname.

-u *username* — Create a new public/secret key pair for the given *username*. Prompts for a password for the given *username*.

-s nisplus
-s nis
-s files
-s ldap — Update the database in the specified source: nisplus (for NIS+), nis (for NIS), files, or ldap (LDAP). Other sources may be available in the future.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO | `chkey`(1), `keylogin`(1), `nisaddcred`(1M), `nisclient`(1M), `nsswitch.conf`(4), `publickey`(4), `attributes`(5)

NOTES | NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

nfsd – NFS daemon

SYNOPSIS

/usr/lib/nfs/nfsd [-a] [-c #_*conn*] [-l *listen_backlog*] [-p *protocol*] [-t *device*] [*nservers*]

DESCRIPTION

nfsd is the daemon that handles client file system requests. Only the super-user can run this daemon.

The nfsd daemon is automatically invoked in run level 3 with the -a option, described below.

By default, nfsd starts over the TCP and UDP transports. You can change this with the -p option, described below.

A previously invoked nfsd daemon started with or without options must be stopped before invoking another nfsd command.

Administrators wanting to change startup parameters for nfsd should, as root, make changes in the /etc/default/nfs file rather than editing the /etc/init.d/nfs.server file. See nfs(4).

OPTIONS

The following options are supported:

-a	Start a NFS daemon over all available connectionless and connection-oriented transports, including UDP and TCP. Equivalent of setting the NFSD_PROTOCOL parameter to ALL in the nfs file.
-c #_*conn*	This sets the maximum number of connections allowed to the NFS server over connection-oriented transports. By default, the number of connections is unlimited. Equivalent of the NFSD_MAX_CONNECTIONS parameter in the nfs file.
-l	Set connection queue length for the NFS TCP over a connection-oriented transport. The default value is 32 entries. Equivalent of the NFSD_LISTEN_BACKLOG parameter in the nfs file.
-p *protocol*	Start a NFS daemon over the specified protocol. Equivalent of the NFSD_PROTOCOL parameter in the nfs file.
-t *device*	Start a NFS daemon for the transport specified by the given device. Equivalent of the NFSD_DEVICE parameter in the nfs file.

OPERANDS

The following operands are supported:

nservers	This sets the maximum number of concurrent NFS requests that the server can handle. This concurrency is achieved by up to *nservers* threads created as needed in the kernel. *nservers* should be based on the load expected on this server. 16 is the usual number of *nservers*. If *nservers* is not specified, the maximum number of concurrent NFS requests will default to 1. Equivalent of the

NFSD_MAX_CONNECTIONS parameter in the nfs file.

USAGE

If the NFS_PORTMON variable is set in /etc/system, then clients are required to use privileged ports (ports < IPPORT_RESERVED) to get NFS services. This variable is equal to zero by default. This variable has been moved from the "nfs" module to the "nfssrv" module. To set the variable, edit the /etc/system file and add this entry:

```
set nfssrv:nfs_portmon = 1
```

EXIT STATUS

0	Daemon started successfully.
1	Daemon failed to start.

FILES

.nfs*XXX*	client machine pointer to an open-but-unlinked file
/etc/default/nfs	contains startup parameters for nfsd
/etc/init.d/nfs.server	shell script for starting nfsd
/etc/system	system configuration information file

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnfssu

SEE ALSO

ps(1), mountd(1M), nfs(4), sharetab(4), system(4), attributes(5)

System Administration Guide: Naming and Directory Services (DNS, NIS, and LDAP)

NOTES

Manually starting and restarting nfsd is not recommended. If it is necessary to do so, use the NFS server start/stop script (/etc/init.d/nfs.server). See *System Administration Guide: Naming and Directory Services (DNS, NIS, and LDAP)* for more information.

NAME

nfslogd – nfs logging daemon

SYNOPSIS

```
/usr/lib/nfs/nfslogd
```

DESCRIPTION

The `nfslogd` daemon provides operational logging to the Solaris NFS server. It is the `nfslogd` daemon's job to generate the activity log by analyzing the RPC operations processed by the the NFS server. The log will only be generated for file systems exported with logging enabled. This is specified at file system export time by means of the `share_nfs`(1M) command.

Each record in the log file includes a time stamp, the IP address (or hostname if it can be resolved) of the client system, the file or directory name the operation was performed on, and the type of operation. In the basic format, the operation can either be an input (i) or output (o) operation. The basic format of the NFS server log is compatible with the log format generated by the Washington University FTPd daemon. The log format can be extended to include directory modification operations, such as `mkdir`, `rmdir`, and `remove`. The extended format is not compatible with the Washington University FTPd daemon format. See `nfslog.conf`(4) for details.

The NFS server logging mechanism is divided in two phases. The first phase is performed by the NFS kernel module, which records raw RPC requests and their results in work buffers backed by permanent storage. The location of the work buffers is specified in the `/etc/nfs/nfslog.conf` file. Refer to `nfslog.conf`(4) for more information. The second phase involves the `nfslogd` user-level daemon, which periodically reads the work buffers, interprets the raw RPC information, groups related RPC operations into single transaction records, and generates the output log. The `nfslogd` daemon then sleeps waiting for more information to be logged to the work buffers. The amount of time that the daemon sleeps can be configured by modifying the `IDLE_TIME` parameter in `/etc/default/nfslogd`. The work buffers are intended for internal consumption of the `nfslogd` daemon.

NFS operations use file handles as arguments instead of path names. For this reason the `nfslogd` daemon needs to maintain a database of file handle to path mappings in order to log the path name associated with an operation instead of the corresponding file handle. A file handle entry is added to the database when a client performs a lookup or other NFS operation that returns a file handle to the client.

Once an NFS client obtains a file handle from a server, it can hold on to it for an indefinite time, and later use it as an argument for an NFS operation on the file or directory. The NFS client can use the file handle even after the server reboots. Because the database needs to survive server reboots, it is backed by permanent storage. The location of the database is specified by the *fhtable* parameter in the `/etc/nfs/nfslog.conf` file. This database is intended for the internal use of the `nfslogd` daemon.

In order to keep the size of the file handle mapping database manageable, `nfslogd` prunes the database periodically. It removes file handle entries that have not been accessed in more than a specified amount of time. The `PRUNE_TIMEOUT` configurable parameter in `/etc/default/nfslogd` specifies the interval length between

successive runs of the pruning process. A file handle record will be removed if it has not been used since the last time the pruning process was executed. Pruning of the database can effectively be disabled by setting the PRUNE_TIMEOUT as high as INT_MAX.

When pruning is enabled, there is always a risk that a client may have held on to a file handle longer than the PRUNE_TIMEOUT and perform an NFS operation on the file handle after the matching record in the mapping database had been removed. In such case, the pathname for the file handle will not be resolved, and the log will include the file handle instead of the pathname.

There are various configurable parameters that affect the behavior of the nfslogd daemon. These parameters are found in /etc/default/nfslogd and are described below:

UMASK	Sets the file mode for the log files, work buffer files and file handle mapping database.
MIN_PROCESSING_SIZE	Specifies the minimum size, in bytes, that the buffer file must reach before processing the work information and writing to the log file. The value of MIN_PROCESSING_SIZE must be between 1 and ulimit.
IDLE_TIME	Specifies the amount of time, in seconds, the daemon should sleep while waiting for more information to be placed in the buffer file. IDLE_TIME also determines how often the configuration file will be reread. The value of IDLE_TIME must be between 1 and INT_MAX.
MAX_LOGS_PRESERVE	The nfslogd periodically cycles its logs. MAX_LOGS_PRESERVE specifies the maximum number of log files to save. When MAX_LOGS_PRESERVE is reached, the oldest files will be overwritten as new log files are created. These files will be saved with a numbered extension, beginning with filename.0. The oldest file will have the highest numbered extension up to the value configured for MAX_LOGS_PRESERVE. The value of MAX_LOGS_PRESERVE must be between 1 and INT_MAX.
CYCLE_FREQUENCY	Specifies how often, in hours, the log files are cycled. CYCLE_FREQUENCY is used to insure that the log files do not get too large.

	The value of `CYCLE_FREQUENCY` must be between 1 and `INT_MAX`.
`MAPPING_UPDATE_INTERVAL`	Specifies the time interval, in seconds, between updates of the records in the file handle to path mapping tables. Instead of updating the `atime` of a record each time that record is accessed, it is only updated if it has aged based on this parameter. The record access time is used by the pruning routine to determine whether the record should be removed from the database. The value of this parameter must be between 1 and `INT_MAX`.
`PRUNE_TIMEOUT`	Specifies when a database record times out, in hours. If the time that elapsed since the record was last accessed is greater than `PRUNE_TIMEOUT` then the record can be pruned from the database. The default value for `PRUNE_TIMEOUT` is 168 hours (7 days). The value of `PRUNE_TIMEOUT` must be between 1 and `INT_MAX`.

EXIT STATUS

The following exit values are returned:

`0` Daemon started successfully.

`1` Daemon failed to start.

FILES

`/etc/nfs/nfslogtab`

`/etc/nfs/nfslog.conf`

`/etc/default/nfslogd`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnfssu

SEE ALSO

`share_nfs`(1M), `nfslog.conf`(4), `attributes`(5)

NAME

nfsstat – NFS statistics

SYNOPSIS

nfsstat [-cnrsmza]

DESCRIPTION

nfsstat displays statistical information about the NFS and RPC (Remote Procedure Call), interfaces to the kernel. It can also be used to reinitialize this information. If no options are given the default is

```
nfsstat -csnra
```

That is, display everything, but reinitialize nothing.

OPTIONS

-a Display NFS_ACL information.

-c Display client information. Only the client side NFS, RPC, and NFS_ACL information is printed. Can be combined with the -n, -r, and -a options to print client side NFS, RPC, and NFS_ACL information only.

-m Display statistics for each NFS mounted file system. This includes the server name and address, mount flags, current read and write sizes, the retransmission count, the attribute cache timeout values, failover information, and the timers used for dynamic retransmission. Note that the dynamic retransmission timers are displayed only where dynamic retransmission is in use. By default, NFS mounts over the TCP protocols and NFS Version 3 mounts over either TCP or UDP do not use dynamic retransmission.

If you specify the -m option, this is the only option nfsstat uses. Any options specified in addition to -m are checked for validity, then ignored.

-n Display NFS information. NFS information for both the client and server side will be printed. Can be combined with the -c and -s options to print client or server NFS information only.

-r Display RPC information.

-s Display server information.

-z Zero (reinitialize) statistics. This option is for use by the super user only, and can be combined with any of the above options to zero particular sets of statistics after printing them.

DISPLAYS

The server RPC display includes the following fields:

calls The total number of RPC calls received.

badcalls The total number of calls rejected by the RPC layer (the sum of badlen and xdrcall as defined below).

nullrecv The number of times an RPC call was not available when it was thought to be received.

`badlen`	The number of RPC calls with a length shorter than a minimum-sized RPC call.
`xdrcall`	The number of RPC calls whose header could not be XDR decoded.
`dupchecks`	The number of RPC calls that looked up in the duplicate request cache.
`dupreqs`	The number of RPC calls that were found to be duplicates.

The server NFS display shows the number of NFS calls received (`calls`) and rejected (`badcalls`), and the counts and percentages for the various calls that were made.

The server NFS_ACL display shows the counts and percentages for the various calls that were made.

The client RPC display includes the following fields:

`calls`	The total number of RPC calls made.
`badcalls`	The total number of calls rejected by the RPC layer.
`badxids`	The number of times a reply from a server was received which did not correspond to any outstanding call.
`timeouts`	The number of times a call timed out while waiting for a reply from the server.
`newcreds`	The number of times authentication information had to be refreshed.
`badverfs`	The number of times the call failed due to a bad verifier in the response.
`timers`	The number of times the calculated time-out value was greater than or equal to the minimum specified time-out value for a call.
`cantconn`	The number of times the call failed due to a failure to make a connection to the server.
`nomem`	The number of times the call failed due to a failure to allocate memory.
`interrupts`	The number of times the call was interrupted by a signal before completing.
`retrans`	The number of times a call had to be retransmitted due to a timeout while waiting for a reply from the server. Applicable only to RPC over connection-less transports.
`cantsend`	The number of times a client was unable to send an RPC request over a connectionless transport when it tried to do so.

The client NFS display shows the number of calls sent and rejected, as well as the number of times a `CLIENT` handle was received (`clgets`), the number of times the `CLIENT` handle cache had no unused entries (`cltoomany`), as well as a count of the various calls and their respective percentages.

The client NFS_ACL display shows the counts and percentages for the various calls that were made.

The `-m` option includes information about mount flags set by mount options, mount flags internal to the system, and other mount information. See `mount_nfs`(1M).

The following mount flags are set by mount options:

`sec`	`sec` has one of the following values:	
	`none`	No authentication.
	`sys`	UNIX-style authentication (UID, GID).
	`short`	Short hand UNIX–style authentication.
	`dh`	`des`–style authentication (encrypted timestamps).
	`krb5`	`kerberos v5`–style authentication.
	`krb5i`	`kerberos v5`–style authentication with integrity.
	`krb5p`	`kerberos v5`–style authentication with privacy.
`hard`	Hard mount.	
`soft`	Soft mount.	
`intr`	Interrupts allowed on hard mount.	
`nointr`	No interrupts allowed on hard mount.	
`noac`	Client is not caching attributes.	
`rsize`	Read buffer size in bytes.	
`wsize`	Write buffer size in bytes.	
`retrans`	NFS retransmissions.	
`timeo`	Initial NFS timeout, in tenths of a second.	
`nocto`	No close-to-open consistency.	
`llock`	Local locking being used (no lock manager).	
`grpid`	System V group id inheritance.	
`rpctimesync`	RPC time sync.	

The following mount flags are internal to the system:

`printed`	"Not responding" message printed.

`down`	Server is down.
`dynamic`	Dynamic transfer size adjustment.
`link`	Server supports links.
`symlink`	Server supports symbolic links.
`readdir`	Use `readdir` instead of `readdirplus`.
`acl`	Server supports NFS_ACL.

The following flags relate to additional mount information:

`vers`	NFS version.
`proto`	Protocol.

The `-m` option also provides attribute cache timeout values. The following fields in `-m` ouput provide timeout values for attribute cache:

`acregmin`	Minimum seconds to hold cached file attributes.
`acregmax`	Maximum seconds to hold cached file attributes.
`acdirmin`	Minimum seconds to hold cached directory attributes.
`acdirmax`	Maximum seconds to hold cached directory attributes.

The following fields in `-m` output provide failover information:

`noresponse`	How many times servers have failed to respond.
`failover`	How many times a new server has been selected.
`remap`	How may times files have been re-evaluated to the new server.
`currserver`	Which server is currently providing NFS service. See the *System Administration Guide: IP Services* for additional details.

The fields in `-m` output shown below provide information on dynamic retransmissions. Note that these items are displayed only where dynamic retransmission is in use.

`srtt`	The value for the smoothed round-trip time, in milliseconds.
`dev`	Estimated deviation, in milliseconds.
`cur`	Current backed-off retransmission value, in milliseconds.

EXIT STATUS

The following exit values are returned:

`0`	Successful completion.
`>0`	An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnfscu

SEE ALSO

`mount_nfs`(1M), `attributes`(5)

Solaris 9 Installation Guide

System Administration Guide: IP Services

NAME

nisaddcred – create NIS+ credentials

SYNOPSIS

nisaddcred [-p *principal*] [-P *nis_principal*] [-l *login_password*] *auth_type* [*domain_name*]

nisaddcred -r [*nis_principal*] [*domain_name*]

DESCRIPTION

The nisaddcred command is used to create security credentials for NIS+ principals. NIS+ credentials serve two purposes. The first is to provide authentication information to various services; the second is to map the authentication service name into a NIS+ principal name.

When the nisaddcred command is run, these credentials get created and stored in a table named cred.org_dir in the default NIS+ domain. If *domain_name* is specified, the entries are stored in the cred.org_dir of the specified domain. The specified domain must either be the one to which you belong, or one in which you are authenticated and authorized to create credentials, that is, a subdomain. Note that the credentials of normal users must be stored in the same domain as their passwords.

It is simpler to add credentials using nisclient(1M), because it obtains the required information itself. nispopulate(1M) is used for "bulk" updates and can also be used to add credentials for entries in the hosts and the passwd NIS+ tables.

NIS+ principal names are used in specifying clients that have access rights to NIS+ objects. For more details, refer to the "Principal Names" subsection of the nis+(1) manual page. See nischmod(1), nischown(1), nis_objects(3NSL), and nis_groups(3NSL). Various other services can also implement access control based on these principal names.

The cred.org_dir table is organized as follows:

cname	auth_type	auth_name	public_data	private_data
user1.foo.com.	LOCAL	2990	10,102,44	
user1.foo.com.	DES	unix.2990@foo.com	098...819	3b8...ab2
user1.foo.com.	DHmmm-n	unix.2990@foo.com	248...428	a42...f32

The cname column contains a canonical representation of the NIS+ principal name. By convention, this name is the login name of a user, or the host name of a machine, followed by a dot ('.') followed by the fully qualified "home" domain of that principal. For users, the home domain is defined to be the domain where their DES credentials are kept. For hosts, their home domain is defined to be the domain name returned by the domainname(1M) command executed on that host.

There are two basic types of *auth_type* entries in the cred.org_dir table, those with authentication type LOCAL, and those with authentication type DES, *auth_type*, specified on the command line in upper or lower case, should be either *local* or des.

However, the `cred.org_dir` table may also be used to hold data for other values of *auth_type*. Currently, this is limited to the mechanisms listed on the `nisauthconf`(1M) man page, for which the `nisaddcred` *auth_type* argument is the same as the name of the mechanism. These mechanisms use a modified form of Secure RPC, and they are similar to the DES authentication type.

If the *auth_type* is `des`, and other authentication mechanisms are configured with `nisauthconf`(1M), then credential entries are added or updated for each mechanism configured. To only add or update 1992-bit Diffie Hellman credentials, that is, those with the *auth_type* of DES, use dh192-0 on the command line. If there are no authentication mechanisms configured, using `des` on the command line will only add or update 192-bit Diffie Hellman credentials.

Entries of type LOCAL are used by the NIS+ service to determine the correspondence between fully qualified NIS+ principal names and users identified by UIDs in the domain containing the `cred.org_dir` table. This correspondence is required when associating requests made using the `AUTH_SYS` RPC authentication flavor (see `rpc_clnt_auth`(3NSL)) to a NIS+ principal name. It is also required for mapping a UID in one domain to its fully qualified NIS+ principal name whose home domain may be elsewhere. The principal's credentials for any authentication flavor may then be sought for within the `cred.org_dir` table in the principal's home domain (extracted from the principal name). The same NIS+ principal may have LOCAL credential entries in more than one domain. Only users, and not machines, have LOCAL credentials. In their home domain, users of NIS+ should have both types of credentials.

The *auth_name* associated with the LOCAL type entry is a UID that is valid for the principal in the domain containing the `cred.org_dir` table. This may differ from that in the principal's home domain. The public information stored in *public_data* for this type contains a list of GIDs for groups in which the user is a member. The GIDs also apply to the domain in which the table resides. There is no private data associated with this type. Neither a UID nor a principal name should appear more than once among the LOCAL entries in any one `cred.org_dir` table.

The DES *auth_type* is used for Secure RPC authentication (see `secure_rpc`(3NSL)).

The authentication name associated with the DES *auth_type* is a Secure RPC *netname*. A Secure RPC netname has the form `unix.id@`*domain.com*, where *domain* must be the same as the domain of the principal. For principals that are users the `id` must be the UID of the principal in the principal's home domain. For principals that are hosts, the `id` is the host's name. In Secure RPC, processes running under effective UID 0 (root) are identified with the host principal. Unlike LOCAL, there cannot be more than one DES credential entry for one NIS+ principal in the NIS+ namespace.

The public information in an entry of authentication type DES is the public key for the principal. The private information in this entry is the private key of the principal encrypted by the principal's network password.

User clients of NIS+ should have credentials of both types in their home domain. In addition, a principal must have a LOCAL entry in the `cred.org_dir` table of each domain from which the principal wishes to make authenticated requests. A client of NIS+ that makes a request from a domain in which it does not have a LOCAL entry will be unable to acquire DES credentials. A NIS+ service running at security level 2 or higher will consider such users unauthenticated and assign them the name *nobody* for determining access rights.

This command can only be run by those NIS+ principals who are authorized to add or delete the entries in the `cred` table.

If credentials are being added for the caller itself, `nisaddcred` automatically performs a keylogin for the caller.

You can list the `cred` entries for a particular principal with `nismatch`(1).

The `cred.org_dir` NIS+ table replaces the maps *publickey.byname* and *netid.byname* used in NIS (YP).

OPTIONS

The following options are supported:

`-p` *principal*

The name *principal* specifies the name of the principal as defined by the naming rules for that specific mechanism. For example, LOCAL credential names are supplied with this option by including a string specifying a UID. For DES credentials, the name should be a Secure RPC netname of the form `unix.id@`*domain.com*, as described earlier. If the `-p` option is not specified, the *auth_name* field is constructed from the effective UID of the current process and the name of the local domain.

`-P` *nis_principal*

Use the NIS+ principal name *nis_principal*. This option should be used when creating LOCAL or DES credentials for users whose home domain is different than the local machine's default domain.

Whenever the `-P` option is not specified, `nisaddcred` constructs a principal name for the entry as follows. When it is not creating an entry of type LOCAL, `nisaddcred` calls `nis_local_principal`, which looks for an existing LOCAL entry for the effective UID of the current process in the `cred.org_dir` table and uses the associated principal name for the new entry. When creating an entry of authentication type LOCAL, `nisaddcred` constructs a default NIS+ principal name by taking the login name of the effective UID for its

	own process, and appending to it a dot ('.') followed by the local machine's default domain. If the caller is a superuser, the machine name is used instead of the login name.
`-l` *login_password*	Use the *login_password* specified as the password to encrypt the secret key for the credential entry. This overrides the prompting for a password from the shell. This option is intended for administration scripts only. Prompting guarantees not only that no one can see your password on the command line using `ps`(1) but it also checks to make sure you have not made any mistakes. *login_password* does not really have to be the user's password but if it is, it simplifies logging in.
`-r` [*nis_principal*]	Remove all credentials associated with the principal *nis_principal* from the `cred.org_dir` table. This option can be used when removing a client or user from the system. If *nis_principal* is not specified the default is to remove credentials for the current *user*. If *domain_name* is not specified, the operation is executed in the default NIS+ domain.

EXAMPLES

EXAMPLE 1 Adding the LOCAL and DES Credentials

The following examples illustrate how to add the LOCAL and DES credentials for some user, `user1`, with a UID of `2990`, who is an NIS+ user principal in the `some.domain.com.` NIS+ domain:

```
example% nisaddcred -p 2990 -P user1.some.domain.com. local
```

Note that credentials are always added in the `cred.org_dir` table in the domain where `nisaddcred` is run, unless *domain_name* is specified as the last parameter on the command line. If credentials are being added from the domain server for its clients, then *domain_name* should be specified. The caller should have adequate permissions to create entries in the `cred.org_dir` table.

The system administrator can add a DES credential for the same user, using the following example:

```
example% nisaddcred -p unix.2990@some.domain.com -P user1.some.domain.com. des
```

Please note that DES credentials can be added only after the LOCAL credentials have been added. Also, if the system is configured to use more than one authentication mechanism, credentials will be made for each mechanism configured. See `nisauthconf`(1M).

Note that the secure RPC netname does not end with a dot ('.') while the NIS+ principal name, specified with the `-P` option, does. This command should be executed from a machine in the same domain as is the user.

EXAMPLE 1 Adding the LOCAL and DES Credentials *(Continued)*

The following example shows how to add a machine's DES credentials in the same domain:

```
example% nisaddcred -p unix.foo@some.domain.com -P foo.some.domain.com. des
```

Please note that no LOCAL credentials are needed in this case.

The following example illustrates how to add a NIS+ workstation's principal DES credential:

```
example% nisaddcred -p unix.host1@sub.some.domain.com \
     -P newhost.sub.some.domain.com. des sub.some.domain.com.
```

This format is particularly useful if you are running this command from a server which is in a higher domain than sub.some.domain.com. Without the last option for domain name, `nisaddcred` would fail because it would attempt to use the default domain of some.domain.com.

The following example illustrates adding DES credentials without being prompted for the root login password:

```
example% nisaddcred -p unix.2990@some.domain.com \
     -P user1.some.domain.com. -l login_password des
```

The following example shows how to add a credential for a user using a specific authentication mechanism that was previously configured with `nisauthconf`(1M). See `nisauthconf`(1M) for a list of the valid values of *auth_type*:

```
example% nisaddcred -p unix.2990@some.domain.com \
     -P user.1.some.domain.com dh640-0
```

The password should be the same for all the credentials that belong to the user. Otherwise, only the credentials encrypted with the user's password will be used at login, and the user will have to run `chkey`(1) using the `-p` option.

The following example shows how to add a DES credential when other authentication mechanisms are configured on the system:

```
example% nisaddcred -p unix.2990@some.domain.com \
     -P user1.some.domain.com dh192-0
```

EXIT STATUS

The following exit values are returned:

`0` Successful operation.

`1` Operation failed.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`chkey`(1), `keylogin`(1), `nis+`(1), `nischmod`(1), `nischown`(1), `nismatch`(1), `nistbladm`(1), `ps`(1), `domainname`(1M), `nisclient`(1M), `nispopulate`(1M), `nis_groups`(3NSL), `nis_local_names`(3NSL), `nis_objects`(3NSL), `rpc_clnt_auth`(3NSL), `secure_rpc`(3NSL), `attributes`(5)

NOTES

NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

nisaddent – create NIS+ tables from corresponding /etc files or NIS maps

SYNOPSIS

/usr/lib/nis/nisaddent [-D *defaults*] [-Paorv] [-t *table*] *type* [*nisdomain*]

/usr/lib/nis/nisaddent [-D *defaults*] [-Paprmov] -f *file* [-t *table*] *type* [*nisdomain*]

/usr/lib/nis/nisaddent [-D *defaults*] [-Parmv] [-t *table*] -y *ypdomain* [-Y *map*] *type* [*nisdomain*]

/usr/lib/nis/nisaddent -d [-AMoq] [-t *table*] *type* [*nisdomain*]

DESCRIPTION

nisaddent creates entries in NIS+ tables from their corresponding /etc files and NIS maps. This operation is customized for each of the standard tables that are used in the administration of Solaris systems. The type argument specifies the type of the data being processed. Legal values for this type are one of aliases, bootparams, ethers, group, hosts, ipnodes, netid, netmasks, networks, passwd, protocols, publickey, rpc, services, shadow, or timezone for the standard tables, or key-value for a generic two-column (key, value) table. For a site specific table, which is not of key-value type, one can use nistbladm(1) to administer it.

The NIS+ tables should have already been created by nistbladm(1), nissetup(1M), or nisserver(1M).

It is easier to use nispopulate(1M) instead of nisaddent to populate the system tables.

By default, nisaddent reads from the standard input and adds this data to the NIS+ table associated with the type specified on the command line. An alternate NIS+ table may be specified with the -t option. For type key-value, a table specification is required.

Note that the *data* type can be different than the table name (-t). For example, the automounter tables have key-value as the table type.

Although, there is a *shadow* data type, there is no corresponding *shadow* table. Both the shadow and the passwd data is stored in the passwd table itself.

Files may be processed using the -f option, and NIS version 2 (YP) maps may be processed using the -y option. The merge option is not available when reading data from standard input.

When a *ypdomain* is specified, the nisaddent command takes its input from the dbm files for the appropriate NIS map (mail.aliases, bootparams, ethers.byaddr, group.byname, hosts.byaddr, hosts.byname, ipnodes.byaddr,ipnodes.byname, netid.byname, netmasks.byaddr, networks.byname, passwd.byname, protocols.byname, publickey.byname, rpc.bynumber, services.byname, or timezone.byname). An alternate NIS map may be specified with the -Y option. For type key-value, a map specification is required. The map must be in the /var/yp/*ypdomain* directory on the local machine. Note that *ypdomain* is case sensitive. ypxfr(1M) can be used to get the NIS maps.

If a *nisdomain* is specified, `nisaddent` operates on the NIS+ table in that NIS+ domain, otherwise the default domain is used.

In terms of performance, loading up the tables is fastest when done through the dbm files (`-y`).

To accommodate other credential entries used by other authentication mechanisms stored in the `cred.org_dir` table, the `publickey` dump output has been modified to include a special algorithm `type` field. This format is incompatible with older versions of `nisaddent`. To produce dumps that can be read by older versions of `nisaddent`, or to load dumps created by such older versions, use the `-o` option.

OPTIONS

The following options are supported:

`-a` Add the file or map to the NIS+ table without deleting any existing entries. This option is the default. Note that this mode only propagates additions and modifications, not deletions.

`-A` All data. This option specifies that the data within the table and all of the data in tables in the initial table's concatenation path be returned.

`-d` Dump the NIS+ table to the standard output in the appropriate format for the given `type`. For tables of type `key-value`, use `niscat`(1) instead. To dump the `cred` table, dump the `publickey` and the `netid` types.

`-D` *defaults* This option specifies a different set of defaults to be used during this operation. The *defaults* string is a series of tokens separated by colons. These tokens represent the default values to be used for the generic object properties. All of the legal tokens are described below.

`ttl=time` This token sets the default time to live for objects that are created by this command. The value `time` is specified in the format as defined by the `nischttl`(1) command. The default is 12 hours.

`owner=`*ownername* This token specifies that the NIS+ principal *ownername* should own the created object. The default for this value is the principal who is executing the command.

`group=`*groupname* This token specifies that the group *groupname* should be the group owner for the object that is created. The default is `NULL`.

`access=`*rights*	This token specifies the set of access rights that are to be granted for the given object. The value *rights* is specified in the format as defined by the `nischmod`(1) command. The default is `– – – –rmcdr – – –r – – –`
`-f` *file*	Specify that `file` should be used as the source of input (instead of the standard input).
`-m`	Merge the file or map with the NIS+ table. This is the most efficient way to bring an NIS+ table up to date with a file or NIS map when there are only a small number of changes. This option adds entries that are not already in the database, modifies entries that already exist (if changed), and deletes any entries that are not in the source. Use the `-m` option whenever the database is large and replicated, and the map being loaded differs only in a few entries. This option reduces the number of update messages that have to be sent to the replicas. Also see the `-r` option.
`-M`	Master server only. This option specifies that lookups should be sent to the master server. This guarantees that the most up-to-date information is seen at the possible expense that the master server may be busy, or that it may be made busy by this operation.
`-o`	Use strictly conforming `publickey` files. Dumps will not add the algorithm `type` field used by additional authentication mechanisms that might be configured using `nisauthconf`(1M). 192-bit keys that are dumped using this option can be read by previous versions of `nisaddent`. However, the algorithm field will be lost and assumed to be "0" when read. Use the `-o` option when reading `publickey` files from previous versions of `nisaddent` to avoid warnings about the missing algorithm field.
`-p`	Process the password field when loading password information from a file. By default, the password field is ignored because it is usually not valid (the actual password appears in a shadow file).
`-P`	Follow concatenation path. This option specifies that lookups should follow the concatenation path of a table if the initial search is unsuccessful.
`-q`	Dump tables in "quick" mode. The default method for dumping tables processes each entry individually. For some tables, for example, hosts, multiple entries must be combined into a single line, so extra requests to the server must be made. In "quick" mode, all of the entries for a table are retrieved in one call to the server, so the table can be dumped more quickly. However, for

large tables, there is a chance that the process will run out of virtual memory and the table will not be dumped.

`-r` Replace the file or map in the existing NIS+ table by first deleting any existing entries, and then add the entries from the source (`/etc` files, or NIS+ maps). This option has the same effect as the `-m` option. The use of this option is *strongly* discouraged due to its adverse impact on performance, unless there are a large number of changes.

`-t` *table* Specify that *table* should be the NIS+ table for this operation. This should be a relative name as compared to your default domain or the `domainname` if it has been specified.

`-v` Verbose.

`-y` *ypdomain* Use the `dbm` files for the appropriate NIS map, from the NIS domain *ypdomain*, as the source of input. The files are expected to be on the local machine in the `/var/yp/`*ypdomain* directory. If the machine is not an NIS server, use `ypxfr`(1M) to get a copy of the `dbm` files for the appropriate map.

`-Y` *map* Use the `dbm` files for *map* as the source of input.

EXAMPLES

EXAMPLE 1 Using `nisaddent`

This example adds the contents of `/etc/passwd` to the `passwd.org_dir` table:

```
example% cat /etc/passwd | nisaddent passwd
```

The next example adds the shadow information. Note that the table type here is "shadow", not "passwd", even though the actual information is stored in the `passwd` table:

```
example% cat /etc/shadow | nisaddent shadow
```

This example replaces the `hosts.org_dir` table with the contents of `/etc/hosts` (in verbose mode):

```
example% nisaddent -rv -f /etc/hosts hosts
```

This example merges the `passwd` map from `yypdomain` with the `passwd.org_dir.nisdomain` table (in verbose mode). The example assumes that the `/var/yp/myypdomain` directory contains the `yppasswd` map.

```
example% nisaddent -mv -y myypdomain passwd nisdomain
```

This example merges the `auto.master` map from `myypdomain` with the `auto_master.org_dir` table:

```
example% nisaddent -m -y myypdomain -Y auto.master \
     -t auto_master.org_dir key-value
```

EXAMPLE 1 Using `nisaddent` *(Continued)*

This example dumps the `hosts.org_dir` table:

```
example% nisaddent -d hosts
```

This example dumps the `ipnodes.org_dir` table:

```
example% nisaddent -d ipnodes
```

ENVIRONMENT VARIABLES

`NIS_DEFAULTS`	This variable contains a default string that will override the NIS+ standard defaults. If the `-D` switch is used, those values will then override both the `NIS_DEFAULTS` variable and the standard defaults. To avoid security accidents, the access rights in the `NIS_DEFAULTS` variable are ignored for the `passwd` table (but access rights specified with `-D` are used).
`NIS_PATH`	If this variable is set, and neither the *nisdomain* nor the *table* are fully qualified, each directory specified in `NIS_PATH` will be searched until the table is found (see `nisdefaults`(1)).

EXIT STATUS

The following exit values are returned:

`0`	Successful operation.
`1`	Failure caused by an error other than parsing.
`2`	A parsing error occurred on an entry. A parsing error does not cause termination; the invalid entries are simply skipped.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`niscat`(1), `nischmod`(1), `nischttl`(1), `nisdefaults`(1), `nistbladm`(1), `nisauthconf`(1M), `nispopulate`(1M), `nisserver`(1M), `nissetup`(1M), `ypxfr`(1M), `hosts`(4), `ipnodes`(4), `passwd`(4), `shadow`(4), `attributes`(5)

NOTES

NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

nisauthconf – configure NIS+ security

SYNOPSIS

nisauthconf [-v] [*mechanism*,...]

DESCRIPTION

nisauthconf controls which authentication flavors NIS+ should use when communicating with other NIS+ clients and servers. If the command is not executed, then NIS+ will default to the AUTH_DES authentication flavor when running security level 2. See rpc.nisd(1M).

nisauthconf takes a list of authentication *mechanism*'s in order of preference. An authentication *mechanism* may use one or more authentication flavors listed below. If des is the only specified mechanism, then NIS+ only use AUTH_DES with other NIS+ clients and servers. If des is the first mechanism, then other authentication *mechanism*'s after des will be ignored by NIS+, except for nisaddcred(1M). After changing the mechanism configuration, the keyserv(1M) daemon must be restarted. Note that doing so will remove encryption keys stored by the running keyserv process. This means that a reboot usually is the safest option when the mechanism configuration has been changed.

The following mechanisms are available:

Authentication *mechanism*	**Authentication Flavor**
des	AUTH_DES
dh640–0	RPCSEC_GSS using 640-bit Diffie-Hellman keys
dh1024–0	RPCSEC_GSS using 1024-bit Diffie-Hellman keys

If no mechanisms are specified, then a list of currently configured mechanisms is printed.

OPTIONS

-v Displays a verbose table listing the currently configured authentication mechanisms.

EXAMPLES

EXAMPLE 1 Configuring a System with only RPCSEC_GSS Authentication Flavor

To configure a system to use only the RPCSEC_GSS authentication flavor with 640-bit Diffie-Hellman keys, execute the following as root:

```
example# /usr/lib/nis/nisauthconf dh640-0
```

EXAMPLE 2 Configuring a System with both RPCSEC_GSS and AUTH_DES Authentication Flavors

To configure a system to use both RPCSEC_GSS (with 640-bit Diffie-Hellman keys) and AUTH_DES authentication flavors:

EXAMPLE 2 Configuring a System with both `RPCSEC_GSS` and `AUTH_DES` Authentication Flavors *(Continued)*

```
example# /usr/lib/nis/nisauthconf dh640-0 des
```

EXAMPLE 3 Transitioning to Other Authentication Flavors

The following example can be used while adding credentials for a new mechanism before NIS+ is authenticating with the new mechanism:

```
example#  /usr/lib/nis/nisauthconf des dh640-0
```

Note that except for `nisaddcred`(1M), NIS+ will not use mechanisms that follow '`des`.'

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` An error occurred.

FILES

`/etc/rpcsec/nisplussec.conf`
NIS+ authentication configuration file. This file may change or be removed in future versions of Solaris.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`nis+`(1), `keyserv`(1M), `nisaddcred`(1M), `rpc.nisd`(1M), `attributes`(5)

NOTES

A NIS+ client of a server that is configured for either dh640–0 or dh1024–0 must run Solaris 7 or later, even if the server is also configured with `des`.

NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

nisbackup – backup NIS+ directories

SYNOPSIS

nisbackup [-v] *backup-dir directory...*

nisbackup [-v] -a *backup-dir*

DESCRIPTION

nisbackup backs up a NIS+ directory object on a NIS+ master server. Updates to the NIS+ database will be temporarily disabled while nisbackup is running. The *backup-dir* is a UNIX directory that must exist prior to running nisbackup. The nisbackup command can be used to backup an individual NIS+ directory object or all (-a) of the NIS+ directory objects served by a master server. The NIS+ directory objects being backed up will be placed into subdirectories under the *backup-dir* directory. These subdirectories are named according to the NIS+ directory object they contain. nisbackup operates on individual NIS+ directory objects (for example, org_dir.wiz.com). This allows an administrator to selectively backup specific directories.

The rpc.nisd(1M) process must be running on the master server with a stable NIS+ database for nisbackup to complete. nisbackup will not attempt to correct any corruption in the NIS+ database, so it is important that backups be done regularly as part of the NIS+ administration.

The first synopsis is used to backup a single NIS+ directory object or a list of NIS+ directory objects. The objects can be partially qualified or fully qualified. The machine on which the command is executing must be the master for the NIS+ directory objects specified.

The second synopsis will backup all of the NIS+ directory objects that are served by this master. The -a option is the recommended method of backing up a master server, since it will backup all NIS+ directory objects that are served by this master. If this server is a master server for more than one domain, the backup will include NIS+ directories that belong to all of the domains served. Individual NIS+ directory objects can be selected for restoring from a *backup-dir* created with the -a option. See nisrestore(1M).

The -a option only includes directory objects for which this server is the master. It is possible, but not recommended, to configure a master server as a replica for other domains. The objects belonging to those replicated domains will not be backed up with the -a option. The backup of replicated objects must be run on the master server for those objects.

Do not use the same *backup-dir* to backup different master servers. Each master server must have its own *backup-dir*.

nisbackup will set the rpc.nisd(1M) to read only mode, which will disable updates to the NIS+ database. This is neccessary to ensure the consistency of the backup. For this reason, nisbackup should not be run while large numbers of updates are being applied to the NIS+ database. Update utilities such as nisaddent(1M) should not be run simultaneously with nisbackup.

OPTIONS

`-a` Creates a backup of all NIS+ directory objects for which this server is a master.

`-v` Verbose option. Additional output will be produced and sent to `syslog`(3C) upon execution of the command (see `syslog.conf`(4)).

OPERANDS

backup-dir The directory into which the subdirectories containing the backed up objects are placed. This must be created prior to running `nisbackup`.

directory The NIS+ directory object(s) being backed up.

EXAMPLES

EXAMPLE 1 Backup of the `org_dir` NIS+ directory object of the domain `foo.com` on a master server to a directory named `/backup`

To backup the `org_dir` NIS+ directory object of the domain `foo.com` on a master server to a directory named `/backup`:

```
master_server# nisbackup /backup org_dir.foo.com.
```

EXAMPLE 2 Backup of the entire NIS+ domain `foo.com` to a directory named `/backup`

To backup the entire NIS+ domain `foo.com` to a directory named `/backup`:

```
master_server# nisbackup /backup foo.com. \
              org_dir.foo.com. groups_dir.foo.com. \
              ctx_dir.foo.com.
```

EXAMPLE 3 Backup of an entire NIS+ database to a backup directory named `/backup`

To backup an entire NIS+ database to a backup directory named `/backup`:

```
master_server# nisbackup -a /backup
```

EXIT STATUS

`0` Successful completion.

`1` An error occurred.

FILES

/backup-dir/backup_list
This ascii file contains a list of all the objects contained in this *backup-dir* directory.

/backup-dir/directory-object
A subdirectory that is created in the *backup-dir* that contains the NIS+ directory-object backup.

/backup-dir/directory-object`/data`
A subdirectory that contains the data files that are part of the NIS+ directory-object backup.

/backup-dir/directory-object`/last.upd`
This data file contains timestamp information about the directory-object.

/backup-dir/directory-object/`data.dict`
A NIS+ data dictionary for all of the objects contained in the NIS+ directory-object backup.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`nis+`(1), `nisdefaults`(1), `nisrm`(1), `nisrestore`(1M), `rpc.nisd`(1M), `syslog`(3C), `xfn`(3XFN), `nisfiles`(4), `syslog.conf`(4), `attributes`(5)

NOTES

NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

nis_cachemgr – NIS+ utility to cache location information about NIS+ servers

SYNOPSIS

/usr/sbin/nis_cachemgr [-i] [-v]

DESCRIPTION

The nis_cachemgr daemon maintains a cache of NIS+ directory objects and active servers for domains. It is responsible for locating servers for a domain on behalf of client processes. This improves performance because only one process has to search for servers. The cache contains location information necessary to contact the NIS+ servers. This includes transport addresses, information neeeded to authenticate the server, and a time to live field which gives a hint on how long the directory object can be cached. The cache helps to improve the performance of the clients that are traversing the NIS+ name space. nis_cachemgr should be running on all the machines that are using NIS+. However, it is not required that the nis_cachemgr program be running in order for NIS+ requests to be serviced.

The cache maintained by this program is shared by all the processes that access NIS+ on a machine. The cache is maintained in a file that is memory mapped by all the processes. See mmap(2). On start up, nis_cachemgr initializes the cache from the cold start file and preserves unexpired entries that already exist in the cache file. See nisinit(1M). Thus, the cache survives machine reboots.

The nis_cachemgr program is normally started from a system startup script. nisshowcache(1M) can be used to look at the cached objects and active servers.

The nisprefadm(1M) command can be used to control which NIS+ servers the nis_cachemgr program will try to select.

The nis_cachemgr program makes NIS+ requests under the NIS+ principal name of the host on which it runs. Before running nis_cachemgr, security credentials for the host should be added to the cred.org_dir table in the host's domain using nisaddcred(1M). Credentials of type DES will be needed if the NIS+ service is operating at security level 2 (see rpc.nisd(1M)). See the DIAGNOSTICS section, below. Additionally, a "keylogin -r " should be done on the machine.

OPTIONS

-i Force nis_cachemgr to ignore the previous cache file and reinitialize the cache from just the cold start file. By default, the cache manager initializes itself from both the cold start file and the old cache file, thereby maintaining the entries in the cache across machine reboots.

-v This flag sets verbose mode. In this mode, the nis_cachemgr program logs not only errors and warnings, but also additional status messages. The additional messages are logged using syslog(3C) with a priority of LOG_INFO.

FILES

/var/nis/NIS_SHARED_DIRCACHE	the shared cache file
/var/nis/NIS_COLD_START	the coldstart file
/etc/init.d/rpc	initialization scripts for NIS+

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`keylogin`(1), `nisaddcred`(1M), `nisinit`(1M), `nisprefadm`(1M), `nisshowcache`(1M), `rpc.nisd`(1M), `mmap`(2), `rpc`(3NSL), `syslog`(3C), `nisfiles`(4), `attributes`(5)

DIAGNOSTICS

The `nis_cachemgr` daemon logs error messages and warnings using `syslog`(3C). Error messages are logged to the `DAEMON` facility with a priority of `LOG_ERR` . Warning messages are logged with a priority of `LOG_WARNING`. Additional status messages can be obtained using the `-v` option.

NOTES

NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

nisclient – initialize NIS+ credentials for NIS+ principals

SYNOPSIS

```
/usr/lib/nis/nisclient -c [-x] [-o] [-v] [-l <network_password>]
     [-d <NIS+_domain>] client_name...

/usr/lib/nis/nisclient -i [-x] [-v] -h <NIS+_server_host>
     [-a <NIS+_server_addr>] [-k <key_domain>] [-d <NIS+_domain>] [-S 0 |
     2]

/usr/lib/nis/nisclient -u [-x] [-v]

/usr/lib/nis/nisclient -r [-x]
```

DESCRIPTION

The nisclient shell script can be used to:

- create NIS+ credentials for hosts and users
- initialize NIS+ hosts and users
- restore the network service environment

NIS+ credentials are used to provide authentication information of NIS+ clients to NIS+ service.

Use the first synopsis (`-c` option) to create individual NIS+ credentials for hosts or users. You must be logged in as a NIS+ principal in the domain for which you are creating the new credentials. You must also have write permission to the local "cred" table. The *client_name* argument accepts any valid host or user name in the NIS+ domain (for example, the *client_name* must exist in the hosts or passwd table). `nisclient` verifies each *client_name* against both the `host` and `passwd` tables, then adds the proper NIS+ credentials for hosts or users. Note that if you are creating NIS+ credentials outside of your local domain, the host or user must exist in the `host` or `passwd` tables in both the local and remote domains.

By default, `nisclient` will not overwrite existing entries in the credential table for the hosts and users specified. To overwrite, use the `-o` option. After the credentials have been created, `nisclient` will print the command that must be executed on the client machine to initialize the host or the user. The `-c` option requires a network password for the client which is used to encrypt the secret key for the client. You can either specify it on the command line with the `-l` option or the script will prompt you for it. You can change this network password later with `passwd`(1) or `chkey`(1).

`nisclient -c` is not intended to be used to create NIS+ credentials for all users and hosts which are defined in the passwd and hosts tables. To define credentials for all users and hosts, use `nispopulate`(1M).

Use the second synopsis (`-i` option) to initialize a NIS+ client machine. The `-i` option can be used to convert machines to use NIS+ or to change the machine's domainname. You must be logged in as super-user on the machine that is to become a NIS+ client. Your administrator must have already created the NIS+ credential for this host by using `nisclient -c` or `nispopulate -C`. You will need the network password your administrator created. `nisclient` will prompt you for the network password to decrypt your secret key and then for this machine's root login password to generate a

new set of secret/public keys. If the NIS+ credential was created by your administrator using `nisclient -c`, then you can simply use the initialization command that was printed by the `nisclient` script to initialize this host instead of typing it manually.

To initialize an unauthenticated NIS+ client machine, use the `-i` option with `-S` *0*. With these options, the `nisclient -i` option will not ask for any passwords.

During the client initialization process, files that are being modified are backed up as *files*`.no_nisplus`. The files that are usually modified during a client initialization are: `/etc/defaultdomain`, `/etc/nsswitch.conf`, `/etc/inet/hosts`, and, if it exists, `/var/nis/NIS_COLD_START`. Notice that a file will not be saved if a backup file already exists.

The `-i` option does not set up a NIS+ client to resolve hostnames using DNS. Please refer to the DNS documentation for information on setting up DNS. (See `resolv.conf`(4)).

It is not necessary to initialize either NIS+ root master servers or machines that were installed as NIS+ clients using `suninstall`(1M).

Use the third synopsis (`-u` option) to initialize a NIS+ user. You must be logged in as the user on a NIS+ client machine in the domain where your NIS+ credentials have been created. Your administrator should have already created the NIS+ credential for your username using `nisclient -c` or `nispopulate`(1M). You will need the network password your administrator used to create the NIS+ credential for your username. `nisclient` will prompt you for this network password to decrypt your secret key and then for your login password to generate a new set of secret/public keys.

Use the fourth synopsis (`-r` option) to restore the network service environment to whatever you were using before `nisclient -i` was executed. You must be logged in as super-user on the machine that is to be restored. The restore will only work if the machine was initialized with `nisclient -i` because it uses the backup files created by the `-i` option.

Reboot the machine after initializing a machine or restoring the network service.

OPTIONS

The following options are supported:

`-a` <*NIS+_server_addr*>	Specifies the IP address for the NIS+ server. This option is used *only* with the `-i` option.
`-c`	Adds `DES` credentials for NIS+ principals.
`-d` <*NIS+_domain*>	Specifies the NIS+ domain where the credential should be created when used in conjunction with the `-c` option. It specifies the name for the new NIS+ domain when used in conjunction with the `-i` option. The default is your current domainname.

`-h` <*NIS+_server_host*>	Specifies the NIS+ server's hostname. This option is used *only* with the `-i` option.
`-i`	Initializes a NIS+ client machine.
`-l` <*network_password*>	Specifies the network password for the clients. This option is used *only* with the `-c` option. If this option is not specified, the script will prompt you for the network password.
`-k` <*key_domain*>	This option specifies the domain where root's credentials are stored. If a domain is not specified, then the system default domain is assumed.
`-o`	Overwrites existing credential entries. The default is not to overwrite. This is used *only* with the `-c` option.
`-r`	Restores the network service environment.
`-S` *0*|*2*	Specifies the authentication level for the NIS+ client. Level `0` is for unauthenticated clients and level `2` is for authenticated (`DES`) clients. The default is to set up with level `2` authentication. This is used *only* with the `-i` option. `nisclient` always uses level `2` authentication (`DES`) for both `-c` and `-u` options. There is no need to run `nisclient` with `-u` and `-c` for level `0` authentication. To configure authentication mechanisms other than DES at security level 2, use `nisauthconf`(1M) before running `nisclient`.
`-u`	Initializes a NIS+ user.
`-v`	Runs the script in verbose mode.
`-x`	Turns the "echo" mode on. The script just prints the commands that it would have executed. Notice that the commands are not actually executed. The default is off.

EXAMPLES

EXAMPLE 1 Adding the DES Credential in the Local Domain

To add the `DES` credential for host `sunws` and user `fred` in the local domain:

```
example% /usr/lib/nis/nisclient -c sunws fred
```

EXAMPLE 2 Adding the DES Credential in a Specified Domain

To add the `DES` credential for host `sunws` and user `fred` in domain `xyz.sun.com.`:

```
example% /usr/lib/nis/nisclient -c -d xyz.sun.com. sunws fred
```

EXAMPLE 3 Initializing the Host in a Specific Domain

To initialize host `sunws` as a NIS+ client in domain `xyz.sun.com.` where `nisplus_server` is a server for the domain `xyz.sun.com.`:

```
example# /usr/lib/nis/nisclient -i -h nisplus_server -d xyz.sun.com
```

The script will prompt you for the IP address of `nisplus_server` if the server is not found in the `/etc/hosts` file. The `-d` option is needed only if your current domain name is different from the new domain name.

EXAMPLE 4 Initializing the Host as an Unauthenticated Client in a Specific Domain

To initialize host `sunws` as an unauthenticated NIS+ client in domain `xyz.sun.com.` where `nisplus_server` is a server for the domain `xyz.sun.com`:

```
example# /usr/lib/nis/nisclient -i -S 0 \
   -h nisplus_server -d xyz.sun.com. -a 129.140.44.1
```

EXAMPLE 5 Initializing the User as a NIS+ principal

To initialize user `fred` as a NIS+ principal, log in as user `fred` on a NIS+ client machine.

```
example% /usr/lib/nis/nisclient -u
```

FILES

`/var/nis/NIS_COLD_START`	This file contains a list of servers, their transport addresses, and their Secure RPC public keys that serve the machines default domain.
`/etc/defaultdomain`	The system default domainname.
`/etc/nsswitch.conf`	Configuration file for the name-service switch.
`/etc/inet/hosts`	Local host name database.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`chkey`(1), `keylogin`(1), `nis+`(1), `passwd`(1), `keyserv`(1M), `nisaddcred`(1M), `nisauthconf`(1M), `nisinit`(1M), `nispopulate`(1M), `suninstall`(1M), `nsswitch.conf`(4), `resolv.conf`(4), `attributes`(5)

NOTES NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

nisinit – NIS+ client and server initialization utility

SYNOPSIS

nisinit -r

nisinit -pY | D | N*parent_domain host*...

nisinit -c [-k <*key_domain*>] -H *host* | -B | -C *coldstart*

DESCRIPTION

nisinit initializes a machine to be a NIS+ client or an NIS+ root master server. It may be easier to use nisclient(1M) or nisserver(1M) to accomplish this same task.

OPTIONS

-r

Initialize the machine to be a NIS+ root server. This option creates the file /var/nis/data/root.object and initialize it to contain information about this machine. It uses the sysinfo(2) system call to retrieve the name of the default domain.

To initialize the machine as an NIS+ root server, it is advisable to use the "-r" option of nisserver(1M), instead of using "nisinit -r".

-p Y | D | N *parent_domain host . . .*

This option is used on a root server to initialize a /var/nis/data/parent.object to make this domain a part of the namespace above it. Only root servers can have parent objects. A parent object describes the namespace ''above'' the NIS+ root. If this is an isolated domain, this option should not be used. The argument to this option tells the command what type of name server is serving the domain above the NIS+ domain. When clients attempt to resolve a name that is outside of the NIS+ namespace, this object is returned with the error NIS_FOREIGNNS indicating that a name space boundary has been reached. It is up to the client to continue the name resolution process.

The parameter *parent_domain* is the name of the parent domain in a syntax that is native to that type of domain. The list of host names that follow the domain parameter are the names of hosts that serve the parent domain. If there is more than one server for a parent domain, the first host specified should be the master server for that domain.

Y — Specifies that the parent directory is a NIS version 2 domain.

D — Specifies that the parent directory is a DNS domain.

N — Specifies that the parent directory is another NIS+ domain. This option is useful for connecting a pre-existing NIS+ subtree into the global namespace.

Note that in the current implementation, the NIS+ clients do not take advantage of the -p feature. Also, since the parent object is currently not replicated on root replica servers, it is recommended that this option not be used.

-c

Initializes the machine to be a NIS+ client. There are three initialization options available: initialize by coldstart, initialize by hostname, and initialize by broadcast.

The most secure mechanism is to initialize from a trusted coldstart file. The second option is to initialize using a hostname that you specify as a trusted host. The third method is to initialize by broadcast and it is the least secure method.

`-C` *coldstart*
: Causes the file *coldstart* to be used as a prototype coldstart file when initializing a NIS+ client. This coldstart file can be copied from a machine that is already a client of the NIS+ namespace. For maximum security, an administrator can encrypt and encode (with `uuencode`(1C)) the coldstart file and mail it to an administrator bringing up a new machine. The new administrator would then decode (with `uudecode`), decrypt, and then use this file with the `nisinit` command to initialize the machine as an NIS+ client. If the coldstart file is from another client in the same domain, the `nisinit` command may be safely skipped and the file copied into the `/var/nis` directory as `/var/nis/NIS_COLD_START`.

`-H` *hostname*
: Specifies that the host `hostname` should be contacted as a trusted NIS+ server. The `nisinit` command will iterate over each transport in the `NETPATH` environment variable and attempt to contact `rpcbind`(1M) on that machine. This hostname *must* be reachable from the client without the name service running. For IP networks this means that there must be an entry in `/etc/hosts` for this host when `nisinit` is invoked.

`-B`
: Specifies that the `nisinit` command should use an IP broadcast to locate a NIS+ server on the local subnet. Any machine that is running the NIS+ service may answer. No guarantees are made that the server that answers is a server of the organization's namespace. If this option is used, it is advisable to check with your system administrator that the server and domain served are valid. The binding information can be dumped to the standard output using the `nisshowcache`(1M) command.

Note that `nisinit -c` will just enable navigation of the NIS+ name space from this client. To make NIS+ your name service, modify the file `/etc/nsswitch.conf` to reflect that. See `nsswitch.conf`(4) for more details.

`-k` *<key_domain>*
: This option specifies the domain where root's credentials are stored. If it is not specified, then the system default domain is assumed. This domain name is used to create the `/var/nis/NIS_COLD_START` file.

RETURN VALUES

`nisinit` returns `0` on success and `1` on failure.

EXAMPLES

EXAMPLE 1 Initializing the Machine as a NIS+ Client using the Host *freddy* as a Trusted Server

This example initializes the machine as an NIS+ client using the host *freddy* as a trusted server.

```
example# nisinit -cH freddy
```

EXAMPLE 2 Setting up a Client using a Trusted Coldstart File

This example sets up a client using a trusted coldstart file.

```
example# nisinit -cC /tmp/colddata
```

EXAMPLE 3 Setting up a Client Using an IP Broadcast

This example sets up a client using an IP broadcast.

```
example# nisinit -cB
```

EXAMPLE 4 Setting up a Root Server

This example sets up a root server.

```
example# nisinit -r
```

ENVIRONMENT VARIABLES

`NETPATH` This environment variable may be set to the transports to try when contacting the NIS+ server (see `netconfig`(4)). The client library will only attempt to contact the server using connection oriented transports.

FILES

`/var/nis/NIS_COLD_START`	This file contains a list of servers, their transport addresses, and their Secure RPC public keys that serve the machine's default domain.
`/var/nis/data/root.object`	This file describes the root object of the NIS+ namespace. It is a standard XDR-encoded NIS+ directory object that can be modified by authorized clients using the `nis_modify()` interface.
`/var/nis/data/parent.object`	This file describes the namespace that is logically above the NIS+ namespace. The most common type of parent object is a DNS object. This object contains contact information for a server of that domain.
`/etc/hosts`	Internet host table.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO nis+(1), uuencode(1C), nisclient(1M), nisserver(1M), nisshowcache(1M), sysinfo (2), hosts(4), netconfig(4), nisfiles(4), attributes(5)

NOTES NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

nisldapmaptest – test NIS+ and LDAP mapping configuration files

SYNOPSIS

```
nisldapmaptest [-s | -r | -d] [-l | -t object] [-v] [-i] [-o] [-m
     conffile] [-x attr=val…] [ col=val…]
```

DESCRIPTION

Use the `nisldapmaptest` utility to test NIS+ to LDAP mapping configuration files. See `NIS+LDAPmapping`(4). The `nisldapmaptest` utility uses much of the same internal interface as the `rpc.nisd`(1M) does to read, add, modify, or delete LDAP data, as specified by the column name and value operand pairs. `nisldapmaptest` does not read or modify any of the `rpc.nisd`(1M) database files.

See NOTES for details on important differences between the ways that `nisldapmaptest` and `rpc.nisd`(1M) operate on LDAP data.

OPTIONS

The `nisldapmaptest` utility supports the following options:

`-d` Delete data in LDAP.

`-i` Ignore failures when obtaining information from the NIS+ server. This enables `nisldapmaptest` to work to some extent, even if the NIS+ server is unreachable, or if the system is not a NIS+ client. However, NIS+ lookups are still attempted, so there may be NIS+ error messages.

In this mode, `nisldapmaptest` also tries to guess things such as NIS+ object types and derives table column information from the mapping rules in the configuration files. Avoid using the `-i` option to add, modify, or delete, until you have determined that the `nisldapmaptest`'s guesses are adequate for your needs.

`-l` Parse the configuration file into internal data structures, and then print out the configuration per those structures. Note that the printed data is not in configuration file format.

Either `-l` or `-t` must be specified. If both are present, `-l` is ignored.

`-m` *conffile* Specify the name of the `NIS+LDAPmapping`(4) configuration file. The default directory is `/var/nis` , and the default mapping file is `NIS+LDAPmapping`.

`-o` For NIS+ tables, work on the NIS+ object itself, specified by means of the `-t` option, not on the table entries.

`-r` Replace or add data in LDAP.

`-s` Search for data in LDAP. This is the default.

`-t` *object* Specify the NIS+ object on which to operate. If the object name is not fully qualified, that is, it does not end in a dot, the value of the `nisplusLDAPbaseDomain` attribute is appended.

`-v` Set the verbose flag. This flag produces extra diagnostic information.

`-x attr=val...` Specify mapping attribute and value pairs to override those obtained by means of the configuration file. Although any attributes defined on `NIS+LDAPmapping`(4) or `rpc.nisd`(4) can be specified, the ones that control `rpc.nisd`(1M) operation have no effect on `nisldapmaptest`.

OPERANDS

The following operands are supported:

`col=val...` NIS+ column and value pairs used to specify which entries should be looked up, added, modified, or deleted. For additions and modifications, use `col=val` to specify the new values.

EXAMPLES

EXAMPLE 1 Searching for a User

Use the following example to search for the user `xyzzy` in the LDAP container specified for the `passwd.org_dir` table.

```
example% nisldapmaptest -t passwd.org_dir name=xyzzy
```

EXAMPLE 2 Listing Table Entries

Use the following example to list all entries in the container specified for the `services.org_dir` table.

```
example% nisldapmaptest -t services.org_dir
```

EXAMPLE 3 Listing an Object

Use the following example to list the `services.org_dir` object itself, as it is stored in LDAP.

```
example% nisldapmaptest -o -t services.org_dir
```

EXAMPLE 4 Modifying a Table Entry

Use the following example to modify the membership list of the group `grp`, in the container specified for the `group.org_dir` table, to be `mem1`, `mem2`, and `mem3`.

```
example% nisldapmaptest -r -t group.org_dir name=grp \
     members=mem1,mem2,mem3
```

EXAMPLE 5 Deleting a Table Entry

Use the following example to delete the host called `bad` from the container specified for the `hosts.org_dir` table.

```
example% nisldapmaptest -d -t hosts.org_dir name=bad
```

EXIT STATUS

The following exit values are returned:

`0` The requested operation was successful.

`!= 0` An error occurred.

FILES

`/var/nis/NIS+LDAPmapping.template`

`/etc/default/rpd.nisd`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisr
Interface Stability	Obsolete

SEE ALSO

`rpc.nisd`(1M), `NIS+LDAPmapping`(4), `rpc.nisd`(4), `attributes`(5)

NOTES

There are several differences between the ways that `nisldapmaptest` and `rpc.nisd` operate:

1. `nisldapmaptest` obtains information about NIS+ by means of the NIS+ API calls, while `rpc.nisd` looks in its internal database. Thus, if the NIS+ server is not available, `nisldapmaptest` may be unable to determine NIS+ object types or table column information.
2. While `nisldapmaptest` can add, modify, or delete LDAP data, it does not modify any NIS+ data.
3. When operating on table entries, if `nisldapmaptest` is unable to obtain the entry from NIS+, it composes LDAP operations using only the supplied `col=val` operands. Depending on the mapping used, this can result in extra LDAP operations, for example, attempting to obtain a `DN` for add, modify, or delete.
4. The default value for `nisplusLDAPbaseDomain` is the system domain name per `sysinfo`(2) in `nisldapmaptest`, but the internal notion of the domain it serves in `rpc.nisd`. While the two usually are the same, this is not necessarily always the case.
5. When more than one NIS+ entry maps to a single LDAP entry, `nisldapmaptest` may be unable to perform a complete update, unless you make sure that the `col=val` specification picks up all relevant NIS+ entries. For example, if you have the `services.org_dir` NIS+ entries:

```
cname   name    proto   port

x       x       tcp     12345
x       y       tcp     12345
x       z       tcp     12345
```

then specifying `cname=x` will pick up all three entries and create or modify the corresponding LDAP entry to have three `CN` values: `x`, `y`, and `z`. However, specifying `name=x` will match just the first NIS+ entry, and create or modify the LDAP entry to have just one `CN`: `x`.

NAME

nislog – display the contents of the NIS+ transaction log

SYNOPSIS

/usr/sbin/nislog [-h *num* | -t *num*] [-v] [*directory*...]

DESCRIPTION

`nislog` displays the contents of the NIS+ server transaction log on the standard output. This command can be used to track changes in the namespace. The `/var/nis/trans.log` file contains the transaction log maintained by the NIS+ server. When updates occur, they are logged to this file and then propagated to replicas as log transactions. When the log is checkpointed, updates that have been propagated to the replicas are removed.

The `nislog` command can only be run on an NIS+ server by superuser. It displays the log entries for that server only.

If *directory* is not specified, the entire log is searched. Otherwise, only those logs entries that correspond to the specified directories are displayed.

OPTIONS

`-h` *num*	Display *num* transactions from the "head" of the log. If the numeric parameter is `0`, only the log header is displayed.
`-t` *num*	Display *num* transactions from the "tail" of the log. If the numeric parameter is `0`, only the log header is displayed.
`-v`	Verbose mode.

FILES

`/var/nis/trans.log`	transaction log

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`nis+`(1), `rpc.nisd`(1M), `nisfiles`(4), `attributes`(5)

NOTES

NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

nisping – send ping to NIS+ servers

SYNOPSIS

/usr/lib/nis/nisping [-uf] [-H *hostname*] [-r | *directory*]

/usr/lib/nis/nisping -C [-a] [-H *hostname*] [*directory*]

DESCRIPTION

In the first SYNOPSIS line, the `nisping` command sends a "ping" to all replicas of a NIS+ directory. Once a replica receives a ping, it will check with the master server for the directory to get updates. Prior to pinging the replicas, this command attempts to determine the last update "seen" by a replica and the last update logged by the master. If these two timestamps are the same, the ping is not sent. The `-f` (force) option will override this feature.

Under normal circumstances, NIS+ replica servers get the new information from the master NIS+ server within a short time. Therefore, there should not be any need to use `nisping`.

In the second SYNOPSIS line, the `nisping -C` command sends a checkpoint request to the servers. If no *directory* is specified, the home domain, as returned by `nisdefaults`(1), is checkpointed. If all directories, served by a given server, have to be checkpointed, then use the `-a` option.

On receiving a checkpoint request, the servers would commit all the updates for the given *directory* from the table log files to the database files. This command, if sent to the master server, will also send updates to the replicas if they are out of date. This option is needed because the database log files for NIS+ are not automatically checkpointed. `nisping` should be used at frequent intervals (such as once a day) to checkpoint the NIS+ database log files. This command can be added to the `crontab`(1) file. If the database log files are not checkpointed, their sizes will continue to grow.

If the server specified by the `-H` option does not serve the directory, then no ping is sent.

Per-server and per-directory access restrictions may apply; see `nisopaccess`(1). `nisping` uses `NIS_CPTIME` and `NIS_PING` (resync (`ping`) of replicas), or `NIS_CHECKPOINT` (for checkpoint). Since the `NIS_PING` operation does not return a status, the `nisping` command is typically unable to indicate success or failure for resyncs.

OPTIONS

`-a`	Checkpoint all directories on the server.
`-C`	Send a request to checkpoint, rather than a ping, to each server. The servers schedule to commit all the transactions to stable storage.
`-H` *hostname*	Only the host `hostname` is sent the ping, checked for an update time, or checkpointed.
`-f`	Force a ping, even though the timestamps indicate there is no reason to do so. This option is useful for debugging.

`-r`	This option can be used to update or get status about the root object from the root servers, especially when new root replicas are added or deleted from the list. If used without `-u` option, `-r` will send a ping request to the servers serving the root domain. When the replicas receive a ping, they will update their root object if needed. The `-r` option can be used with all other options except with the `-C` option; the root object need not be checkpointed.
`-u`	Display the time of the last update; no servers are sent a ping.

RETURN VALUES

`–1`	No servers were contacted, or the server specified by the `-H` switch could not be contacted.
`0`	Success.
`1`	Some, but not all, servers were successfully contacted.

EXAMPLES

EXAMPLE 1 Using `nisping`

This example pings all replicas of the default domain:

```
example% nisping
```

Note that this example will not ping the the `org_dir` and `groups_dir` subdirectories within this domain.

This example pings the server *example* which is a replica of the *org_dir.foo.com.* directory:

```
example% nisping -H example org_dir.foo.com.
```

This example checkpoints all servers of the *org_dir.bar.com.* directory.

```
example% nisping -C org_dir.bar.com.
```

ENVIRONMENT VARIABLES

`NIS_PATH`	If this variable is set, and the NIS+ directory name is not fully qualified, each directory specified will be searched until the directory is found.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`crontab`(1), `nisdefaults`(1), `nisopaccess`(1), `nislog`(1M), `nisfiles`(4), `attributes`(5)

NOTES NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

nispopulate – populate the NIS+ tables in a NIS+ domain

SYNOPSIS

/usr/lib/nis/nispopulate `-Y [-x] [-f] [-n] [-u] [-v] [-S` *0* `|` *2*`] [-l <`*network_passwd*`>] [-d <`*NIS+_domain*`>] -h <`*NIS_server_host*`> [-a <`*NIS_server_addr*`>] -y <`*NIS_domain*`> [`*table*`] ...`

/usr/lib/nis/nispopulate `-F [-x] [-f] [-u] [-v] [-S 0 | 2] [-d <`*NIS+_domain*`>] [-l <`*network_passwd*`>] [-p <`*directory_path*`>] [`*table*`] ...`

/usr/lib/nis/nispopulate `-C [-x] [-f] [-v] [-d <`*NIS+_domain*`>] [-l <`*network_passwd*`>] [`*hosts* `|` *passwd*`]`

DESCRIPTION

The `nispopulate` shell script can be used to populate NIS+ tables in a specified domain from their corresponding files or NIS maps. `nispopulate` assumes that the tables have been created either through `nisserver`(1M) or `nissetup`(1M).

The table argument accepts standard names that are used in the administration of Solaris systems and non-standard *key-value* type tables. See `nisaddent`(1M) for more information on *key-value* type tables. If the table argument is not specified, `nispopulate` will automatically populate each of the standard tables. These standard (default) tables are: `auto_master`, `auto_home`, `ethers`, `group`, `hosts`, `ipnodes`, `networks`, `passwd`, `protocols`, `services`, `rpc`, `netmasks`, `bootparams`, `netgroup`, `aliases` and `shadow`. Note that the `shadow` table is only used when populating from files. The non-standard tables that `nispopulate` accepts are those of *key-value* type. These tables must first be created manually with the `nistbladm`(1) command.

Use the first synopsis (`-Y`) to populate NIS+ tables from NIS maps. `nispopulate` uses `ypxfr`(1M) to transfer the NIS maps from the NIS servers to the `/var/yp/`<*NIS_domain*> directory on the local machine. Then, it uses these files as the input source. Note that <*NIS_domain*> is case sensitive. Make sure there is enough disk space for that directory.

Use the second synopsis (`-F`) to populate NIS+ tables from local files. `nispopulate` will use those files that match the table name as input sources in the current working directory or in the specified directory.

Note that when populating the `hosts`, `ipnodes`, and `passwd` tables, `nispopulate` will automatically create the NIS+ credentials for all users and `hosts` (`ipnodes`) that are defined in the `hosts`, `ipnodes`, and `passwd` tables, respectively. A network passwd is required to create these credentials. This network password is used to encrypt the secret key for the new users and hosts. This password can be specified using the `-l` option or it will use the default password, "nisplus". `nispopulate` will not overwrite any existing credential entries in the credential table. Use `nisclient`(1M) to overwrite the entries in the `cred` table. It creates both LOCAL and DES credentials for users, and only DES credentials for hosts. To disable automatic credential creation, specify the "`-S` *0*" option.

The third synopsis (`-C`) is used to populate NIS+ credential table with level 2 authentication (DES) from the `hosts`, `ipnodes` and `passwd` tables of the specified domain. The valid table arguments for this operation are `hosts`, `ipnodes` and

`passwd`. If this argument is not specified then it will use `hosts`, `ipnodes` and `passwd` as the input source. If other authentication mechanisms are configured using `nisauthconf`(1M), the NIS+ credential table will be loaded with credentials for those mechanisms.

If `nispopulate` was earlier used with "`-S` *0*" option, then no credentials were added for the hosts or the users. If later the site decides to add credentials for all users and hosts, then this (`-C`) option can be used to add credentials.

OPTIONS

`-a` <*NIS_server_addr*>	Specifies the IP address for the NIS server. This option is only used with the `-Y` option.
`-C`	Populate the NIS+ credential table from `hosts`, `ipnodes`, and `passwd` tables using DES authentication (security level 2). If other authentication mechanisms are configured using `nisauthconf`(1M), the NIS+ credential table will be populated with credentials for those mechanisms.
`-d` <*NIS+_domain.*>	Specifies the NIS+ domain. The default is the local domain.
`-F`	Populates NIS+ tables from files.
`-f`	Forces the script to populate the NIS+ tables without prompting for confirmation.
`-h` <*NIS_server_host*>	Specifies the NIS server hostname from where the NIS maps are copied from. This is only used with the `-Y` option. This hostname must be present in the NIS+ `hosts` or `ipnodes` table, or in the `/etc/hosts` or `/etc/inet/ipnodes` file. If the hostname is not defined, the script will prompt you for its IP address, or you can use the `-a` option to specify the address manually.
`-l` <*network_passwd*>	Specifies the network password for populating the NIS+ credential table. This is only used when you are populating the `hosts`, `ipnodes`, and `passwd` tables. The default passwd is “nisplus”.
`-n`	Does not overwrite local NIS maps in `/var/yp/`<*NISdomain*> directory if they already exist. The default is to overwrite the existing NIS maps in the local `/var/yp/`<*NISdomain*> directory. This is only used with the `-Y` option.
`-p` <*directory_path*>	Specifies the directory where the files are stored. This is only used with the `-F` option. The default is the current working directory.

`-S` *0*|*2*	Specifies the authentication level for the NIS+ clients. Level 0 is for unauthenticated clients and no credentials will be created for users and hosts in the specified domain. Level 2 is for authenticated (DES) clients and DES credentials will be created for users and hosts in the specified domain. The default is to set up with level 2 authentication (DES). There is no need to run `nispopulate` with `-C` for level 0 authentication. Also, if other authentication mechanisms are configured with `nisauthconf`(1M), credentials for those mechanisms will also be populated for the NIS+ clients.
`-u`	Updates the NIS+ tables (ie., adds, deletes, modifies) from either files or NIS maps. This option should be used to bring an NIS+ table up to date when there are only a small number of changes. The default is to add to the NIS+ tables without deleting any existing entries. Also, see the `-n` option for updating NIS+ tables from existing maps in the `/var/yp` directory.
`-v`	Runs the script in verbose mode.
`-x`	Turns the "echo" mode on. The script just prints the commands that it would have executed. Note that the commands are not actually executed. The default is off.
`-Y`	Populate the NIS+ tables from NIS maps.
`-y` <*NIS_domain*>	Specifies the NIS domain to copy the NIS maps from. This is only used with the `-Y` option. The default domainname is the same as the local domainname.

EXAMPLES

EXAMPLE 1 Using `nispopulate`

To populate all the NIS+ standard tables in the domain *xyz.sun.com.* from NIS maps of the *yp.sun.COM* domain as input source where host *yp_host* is a YP server of *yp.sun.COM*:

```
nis_server# /usr/lib/nis/nispopulate -Y -y yp.sun.COM \
-h yp_host -d xyz.sun.com.
```

To update all of the NIS+ standard tables from the same NIS domain and hosts shown above:

```
nis_server# /usr/lib/nis/nispopulate -Y -u -y yp.sun.COM -h yp_host \
-d xyz.sun.com.
```

To populate the `hosts` table in domain *xyz.sun.com.* from the `hosts` file in the `/var/nis/files` directory and using "somepasswd" as the network password for key encryption:

EXAMPLE 1 Using `nispopulate` *(Continued)*

```
nis_server# /usr/lib/nis/nispopulate -F -p \
 /var/nis/files -l somepasswd hosts
```

To populate the `passwd` table in domain *xyz.sun.com.* from the `passwd` file in the `/var/nis/files` directory without automatically creating the NIS+ credentials:

```
nis_server# /usr/lib/nis/nispopulate -F -p /var/nis/files \
 -d xys.sun.com. -S 0 passwd
```

To populate the credential table in *domain xyz.sun.com.* for all users defined in the passwd table.

```
nis_server# /usr/lib/nis/nispopulate -C -d xys.sun.com. passwd
```

To create and populate a non-standard key-value type NIS+ table, "private", from the file `/var/nis/files/private`: (nispopulate assumes that the private.org_dirkey-value type table has already been created).

```
nis_server# /usr/bin/nistbladm -D access=og=rmcd,nw=r \
-c private key=S,nogw= value=,nogw= private.org.dir
nis_server# /usr/lib/nis/nispopulate -F -p /var/nis/files private
```

ENVIRONMENT VARIABLES

`nispopulate` normally creates temporary files in the directory `/tmp`. You may specify another directory by setting the environment variable `TMPDIR` to your chosen directory. If `TMPDIR` is not a valid directory, then `nispopulate` will use `/tmp`).

FILES

`/etc/inet/hosts`	local host name database
`/etc/inet/ipnodes`	local database associating names of nodes with IP addresses
`/var/yp`	NIS (YP) domain directory
`/var/nis`	NIS+ domain directory
`/tmp`	

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`nis+`(1), `nistbladm`(1), `nisaddcred`(1M), `nisaddent`(1M), `nisauthconf`(1M), `nisclient`(1M), `nisserver`(1M), `nissetup`(1M), `rpc.nisd`(1M), `ypxfr`(1M), `attributes`(5)

NOTES NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

nisprefadm – NIS+ utility to set server preferences for NIS+ clients

SYNOPSIS

/usr/bin/nisprefadm `-a {-L | -G} [-o` *opt-string*`] [-d` *domain*`] [-C` *client*`]` *server*...

/usr/bin/nisprefadm `-m {-L | -G} [-o` *opt-string*`] [-d` *domain*`] [-C` *client*`]` *oldserver=newserver*...

/usr/bin/nisprefadm `-r {-L | -G} [-o` *opt-string*`] [-d` *domain*`] [-C` *client*`]` *server*...

/usr/bin/nisprefadm `-u {-L | -G} [-o` *opt-string*`] [-d` *domain*`] [-C` *client*`]` *server*...

/usr/bin/nisprefadm `-x {-L | -G} [-d` *domain*`] [-C` *client*`]`

/usr/bin/nisprefadm `-l {-L | -G} [-d` *domain*`] [-C` *client*`]`

/usr/bin/nisprefadm `-F`

DESCRIPTION

`nisprefadm` defines which servers are to be preferred by NIS+ clients. This information is used by `nis_cachemgr`(1M) to control the order in which it selects which server to use for a particular domain. On a client system, the cache manager first looks for a local preferred server list in `/var/nis`. If it doesn't find one, it looks for an entry with its host name in the NIS+ table. Finally, if it doesn't find it there, it looks for an entry for its subnet.

By default, `nis_cachemgr` puts all servers that are on the same subnet as the client system (that is, local servers) are on the preferred server list. In some cases this default preferred server list is inadequate. For example, if all of the servers for a domain are remote, but some are *closer* than others, the cache manager should try to select the closer one. Because the cache manager has no reliable way to determine the distance to remote servers, `nisprefadm` is used to provide this information.

The preferred server information is stored either globally in a NIS+ table (with the `-G` option) or locally in a file, `/var/nis/client_info` (with the `-L` option). It is preferable to store the information globally so that it can be used by all clients on a subnet. The `nis_cachemgr` process on a client machine reloads the preferred server information periodically, depending on the machine's setup. If the local file is used, the information is reloaded every 12 hours. If the global table is used, the information is reloaded based on the TTL value of the client information table. This TTL value can be changed using `nischttl`(1). If you want your changes to take effect immediately, use the `nisprefadm -F` command. When changing local information (`-L`), `nisprefadm` automatically forces `nis_cachemgr` to reload the information.

The cache manager assigns weights to all of the servers on the preferred list. By default, local servers (that is, servers on the same subnet) are given a weight of 0. Other servers are given the weight, "infinite". This can be changed by using the `nisprefadm` command and giving a weight in parentheses after the server name. When selecting a server for a domain, the cache manager first tries to contact the

servers with the lowest weight. If it doesn't get a response, it tries the servers with the next lowest weight, and so on. If it fails to get a response from any of the preferred servers, it tries to contact the non-preferred servers.

The use of weights gives fine control over the server selection process, but care must be given to avoid assigning too many different weights. For example, if weights 0, 1, 2, and 3 are used, but all of the servers with weight 0, 1, and 2, are unavailable, then there will be a noticeable delay in selecting a server. This is because the cache manager waits 5 seconds for a response at each weight level before moving on to the next one. As a general rule, one or two weight levels provides a good balance of server selection control and performance.

When specifying a server name, it is not necessary to fully qualify the name. When the cache manager tries to access a domain, it compares the list of servers for the domain with the list of preferred servers. It will find a match if a preferred server name is a prefix of the name of a server for the domain. If a domain is served by two servers with the same prefix, the preferred server name must include enough of the domain name to distinguish the two.

The `nis_cachemgr`(1M) process automatically adds local servers (same subnet as the client) to the preferred server list with a weight of 0. Thus, it is not necessary to specify them, though it does no harm.

If you specify a weight for a server, you probably should quote the parentheses to avoid having the shell interpret them. The following command illustrates this:

```
example% nisprefadm -G -a -C client1 "srv1(2)"
```

In general, `nis_cachemgr` does a fairly good job of selecting servers on its own. Therefore, the use of `nisprefadm` is not usually necessary. Some situations in which it is recommended are:

No local servers, many remote servers
: In this case, `nis_cachemgr` needs to choose one of the remote servers. Because it doesn't have information on which is closest, it sends a ping to all of them and then selects the one that responds fastest. This may not always select the best server. If some of the servers are closer to the client than the others, they should be listed as preferred servers so that `nis_cachemgr` will try them first. This reduces the amount of network traffic for selecting a server.

Very remote servers
: In some networks there are NIS+ servers that are only reachable through very slow network connections. It is usually best to avoid unnecessary traffic over that connection. If the *pref_type*=`pref_only` option is set along with preferred servers, then only the preferred servers are contacted for domains they serve. The non-preferred servers are not tried at all; even if all of the preferred servers are unavailable. For domains that are not served by any of the preferred servers, the `pref_only` option is ignored.

OPTIONS In the SYNOPSIS, when several options are surrounded by braces (that is, by '{' and '}') one of the options must be specified.

`-a` Add the specified servers to the preferred server list.

`-C` *client* Store the preferred server information with the key, *client*. The *client* can be either a hostname or a subnet number. When a hostname is specified, the preferred server information applies to that host only. When a subnet is specified, the preferred server information applies to all clients on that subnet. The cache manager searches for host specific entries first. It only searches for subnet entries if no host entry is found. If this option is not specified, then the hostname of the machine on which the command is run is used.

`-d` *domain* Specify the *domain* to which the command is to apply.

`-F` Tells `nis_cachemgr`(1M) to refresh its preferred server information. The program periodically does this anyway, but this option forces it to do the refresh immediately. When updating the local information, `nis_cachemgr` automatically refreshes the preferred server information.

This option must be executed as `root`.

`-l` List the current preferred server information.

`-L | -G` Store the preferred server information locally in the file, `/var/nis/client_info` (the `-L` option), or globally in a NIS+ table `client.info.org-dir.`*domain* (the `-G` option). If the information is stored locally, then it only applies to the system on which the command is run. If it is stored globally then it can apply to all systems on a subnet (depending on the value of the `-C` option).

The `-L` option must be run as `root`.

`-m` Modify the preferred server list. The server specified by *oldserver* is replaced by *newserver*. This is typically used to change the weight for a server.

`-o` Specify additional options to control server selection. Currently the only valid option is *pref_type*, which can have a value of either `all` (the default) or `pref_only`. If the value is `all`, then the cache manager tries to contact non-preferred servers if all of the preferred servers fail to respond. If `pref_only` is specified, then it won't try non-preferred servers. The only exception to this is when a domain is not served by any of the preferred servers. In this case, the cache manager ignores the option. This is to avoid requiring that preferred servers be defined for every domain.

`-r` Remove the specified servers from the preferred server list.

`-u` Clear the list of preferred servers and then add the specified servers to the preferred server list.

`-x` Remove the preferred server information completely.

RETURN VALUES

`nisprefadm` returns the following values:

`0` On success.

`1` On failure.

EXAMPLES

EXAMPLE 1 Using `nisprefadm`

This command sets the preferred server list for the system on which it is run:

```
example% nisprefadm -L -a srv1 srv2
```

The information is stored in a file, `/var/nis/client_info`, so it will only affect this one system.

The following command has the same effect, but the information is stored in a NIS+ table in the default domain.

```
example% nisprefadm -G -a srv1 srv2
```

As a system administrator, you might want to set the preferred server information for a client system other than the one you are running the command on. The following command sets the preferred server information for a client system named *client1*:

```
example% nisprefadm -G -a -C client1 srv1 srv2
```

It is common for all client systems on a subnet to use the same set of preferred servers. The following command sets a preferred server list that applies to all clients on subnet, 192.85.18.0:

```
example% nisprefadm -G -a -C 192.85.18.0 srv1 srv2
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`nischttl`(1), `nis_cachemgr`(1M), `attributes`(5)

NOTES

NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

nisrestore – restore NIS+ directory backup

SYNOPSIS

nisrestore [-fv] *backup-dir directory...*

nisrestore [-fv] -a *backup-dir*

nisrestore -t *backup-dir*

DESCRIPTION

nisrestore restores an existing backup of a NIS+ directory object that was created using nisbackup(1M). The *backup-dir* is the UNIX directory that contains the NIS+ backup on the server being restored. The nisrestore command can be used to restore a NIS+ directory object or a complete NIS+ database. It also can be used as an "out of band" fast replication for a new replica server being initialized. The rpc.nisd(1M) daemon must be stopped before running nisrestore.

The first synopsis is used to restore a single directory object or a specified list of directory objects. The directory can be partially qualified or fully qualified. The server being restored will be verified against the list of servers serving the directory. If this server is not configured to serve this object, nisrestore will exit with an error. The -f option will override this check and force the operation.

The second synopsis will restore all of the directory objects contained in the *backup-dir*. Again, the server will be validated against the serving list for each of the directory objects in the *backup-dir*. If one of the objects in the *backup-dir* are not served by this server, nisrestore will exit with an error. The -f option will override this check and force the operation.

The -a option will attempt to restore all NIS+ objects contained in the *backup-dir*. If any of these objects are not served by the server, nisrestore will exit with an error. If the *backup-dir* contains objects that are not served by the server, nisrestore must be executed without the -a option and the specific directory objects listed.

The -f option will disable verification of the server being configured to serve the objects being restored. This option should be used with care, as data could be inadvertently restored to a server that doesn't serve the restored data. This option is required in the case of restoring a single server domain (master server only) or if the other NIS+ servers are unavailable for NIS+ lookups.

The combination of options -f and -a should be used with caution, as no validation of the server serving the restored objects will be done.

New replicas can be quickly added to a namespace with the nisrestore command. The steps are as follows.

Configure the new replica on the master server (see nisserver(1M)):

```
master# nisserver -R -h replica
```

Kill the rpc.nisd server process on the new replica server:

```
replica# kill rpc.nisd-pid
```

Create a backup of the NIS+ database on the master, which will include the new replica information. See `nisbackup`(1M). The `/backup` will need to be exported to the new replica. See `share_nfs`(1M).

```
master# nisbackup -a /backup
```

Restore the backup of the NIS+ database on the new replica. Use the `-f` option if `nisrestore` is unable to lookup the NIS+ objects being restored. The backup should be available through nfs or similar means. See `share_nfs`(1M).

```
replica# nisrestore -f -a //nfs-mnt/backup
```

Restart the `rpc.nisd`(1M) process on the new replica, and the server will immediately be available for service.

OPTIONS

`-a`	Restores all directory objects included in the *backup-dir* partition.
`-f`	Forces the restoration of a directory without the validation of the server in the directory object's serving list.
`-t`	Lists all directory objects contained in *backup-dir*.
`-v`	Verbose option. Additional output will be produced upon execution of the command.

OPERANDS

backup-dir	The UNIX directory that contains the data files for the NIS+ directory objects to be restored.
directory	The NIS+ directory object(s) to be restored. This can be a fully or partially qualified name.

EXAMPLES

EXAMPLE 1 Restoring the Directory Object on a Replica Server from a Local UFS Partition

To restore the `org_dir` directory object of the domain `foo.com` on a replica server from a local ufs partition named `/var/backup`:

```
replica_server# nisrestore /var/backup org_dir.foo.com.
```

EXAMPLE 2 Forcing the Restore of a Backed up NIS+ Namespace to a Replica Server From the Backup Partition

To force the restore of an entire backed up NIS+ namespace to a replica server from the backup partition named `/var/backup`:

```
replica_server# nisrestore -f -a /var/backup
```

EXAMPLE 3 Restoring the Subdomain on a Master Server From a Backup that Includes Other Directory Objects

To restore the subdomain `sub.foo.com` on a master server, from a backup that includes other directory objects:

EXAMPLE 3 Restoring the Subdomain on a Master Server From a Backup that Includes Other Directory Objects *(Continued)*

```
master_server# nisrestore /var/backup sub.foo.com. \
  org_dir.sub.foo.com. groups_dir.sub.foo.com.
```

EXIT STATUS

`0` Successful completion.

`1` An error occurred.

FILES

/backup-dir/`backup_list`
This ASCII file contains a list of all the objects contained in this *backup-dir* directory. This information can be displayed with the `-t` option.

/backup-dir/directory-object
A subdirectory that is created in the *backup-dir* which contains the directory-object backup.

/backup-dir/directory-object`/data`
A subdirectory that contains the data files that are part of the directory-object backup.

/backup-dir/directory-object`/last.upd`
This data file contains timestamp information about the directory-object.

/backup-dir/directory-object`/data.dict`
A NIS+ data dictionary for all of the objects contained in this directory-object backup.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`nis+`(1), `nisdefaults`(1), `nisbackup`(1M), `nisserver`(1M), `rpc.nisd`(1M), `share_nfs`(1M), `nisfiles`(4), `attributes`(5)

NOTES

NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

nisserver – set up NIS+ servers

SYNOPSIS

/usr/lib/nis/nisserver `-r [-x] [-f] [-v] [-Y] [-d` *NIS+_domain*`] [-g` *NIS+_groupname*`] [-l` *network_passwd*`]`

/usr/lib/nis/nisserver `-M [-x] [-f] [-v] [-Y] -d` *NIS+_domain* `[-g` *NIS+_groupname*`] [-h` *NIS+_server_host*`]`

/usr/lib/nis/nisserver `-R [-x] [-f] [-v] [-Y] [-d` *NIS+_domain*`] [-h` *NIS+_server_host*`]`

DESCRIPTION

The `nisserver` shell script can be used to set up a root master, non-root master, and replica NIS+ server with level 2 security (DES). If other authentication mechanisms are configured with `nisauthconf`(1M), `nisserver` will set up a NIS+ server using those mechanisms. `nisauthconf`(1M) should be used before `nisserver`.

When setting up a new domain, this script creates the NIS+ directories (including `groups_dir` and `org_dir`) and system table objects for the domain specified. It does not populate the tables. `nispopulate`(1M) must be used to populate the tables.

OPTIONS

`-d` *NIS+_domain*	Specifies the name for the NIS+ domain. The default is your local domain.
`-f`	Forces the NIS+ server setup without prompting for confirmation.
`-g` *NIS+_groupname*	Specifies the NIS+ group name for the new domain. This option is not valid with `-R` option. The default group is admin.<domain>.
`-h` *NIS+_server_host*	Specifies the hostname for the NIS+ server. It must be a valid host in the local domain. Use a fully qualified hostname (for example, *hostx.xyz.sun.com.*) to specify a host outside of your local domain. This option is *only* used for setting up non-root master or replica servers. The default for non-root master server setup is to use the same list of servers as the parent domain. The default for replica server setup is the local hostname.
`-l` *network_password*	Specifies the network password with which to create the credentials for the root master server. This option is *only* used for master root server setup (`-r` option). If this option is not specified, the script prompts you for the login password.
`-M`	Sets up the specified host as a master server. Make sure that `rpc.nisd`(1M) is running on the new master server before this command is executed.
`-R`	Sets up the specified host as a replica server. Make sure that `rpc.nisd` is running on the new replica server.

`-r`	Sets up the server as a root master server. Use the `-R` option to set up a root replica server.
`-v`	Runs the script in verbose mode.
`-x`	Turns the echo mode on. The script just prints the commands that it would have executed. Note that the commands are not actually executed. The default is off.
`-Y`	Sets up a NIS+ server with NIS-compatibility mode. The default is to set up the server without NIS-compatibility mode.

USAGE

Use the first synopsis of the command (`-r`) to set up a root master server. To run the command, you must be logged in as super-user on the server machine.

Use the second synopsis of the command (`-M`) to set up a non-root master server for the specified domain. To run the command, you must be logged in as a NIS+ principal on a NIS+ machine and have write permission to the parent directory of the domain that you are setting up. The new non-root master server machine must already be an NIS+ client (see `nisclient`(1M)) and have the `rpc.nisd`(1M) daemon running.

Use the third synopsis of the command (`-R`) to set up a replica server for both root and non-root domains. To run the command, you must be logged in as a NIS+ principal on a NIS+ machine and have write permission to the parent directory of the domain that you are replicating. The new non-root replica server machine must already be an NIS+ client and have the `rpc.nisd` daemon running.

EXAMPLES

EXAMPLE 1 Setting up Servers

To set up a root master server for domain `sun.com.`:

```
root_server# /usr/lib/nis/nisserver -r -d sun.com.
```

For the following examples make sure that the new servers are NIS+ clients and that `rpc.nisd` is running on these hosts before executing `nisserver`. To set up a replica server for the `sun.com.` domain on host *sunreplica*:

```
root_server# /usr/lib/nis/nisserver -R -d sun.com. -h sunrep
```

To set up a non-root master server for domain *xyz.sun.com.* on host *sunxyz* with the NIS+ groupname as *admin-mgr.xyz.sun.com.*:

```
root_server# /usr/lib/nis/nisserver -M -d xyz.sun.com. -h sunxyz \
-g admin-mgr.xyz.sun.com.
```

To set up a non-root replica server for domain *xyz.sun.com.* on host *sunabc*:

```
sunxyz# /usr/lib/nis/nisserver -R -d xyz.sun.com. -h sunabc
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`nis+`(1), `nisgrpadm`(1), `nismkdir`(1), `nisaddcred`(1M), `nisauthconf`(1M), `nisclient` (1M), `nisinit`(1M), `nispopulate`(1M), `nisprefadm`(1M), `nissetup`(1M), `rpc.nisd`(1M), `attributes`(5)

NOTES

NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

nissetup – initialize a NIS+ domain

SYNOPSIS

/usr/lib/nis/nissetup [-Y] [*domain*]

DESCRIPTION

`nissetup` is a shell script that sets up a NIS+ domain to service clients that wish to store system administration information in a domain named *domain*. This domain should already exist prior to executing this command. See `nismkdir`(1) and `nisinit`(1M).

A NIS+ domain consists of a NIS+ directory and its subdirectories: `org_dir` and `groups_dir`. `org_dir` stores system administration information and `groups_dir` stores information for group access control.

`nissetup` creates the subdirectories `org_dir` and `groups_dir` in *domain*. Both subdirectories will be replicated on the same servers as the parent domain. After the subdirectories are created, `nissetup` creates the default tables that NIS+ serves. These are `auto_master`, `auto_home`, `bootparams`, `cred`, `ethers`, `group`, `hosts`, `mail_aliases`, `netmasks`, `networks`, `passwd`, `protocols`, `rpc`, `services`, and `timezone`. The `nissetup` script uses the `nistbladm`(1) command to create these tables. The script can be easily customized to add site specific tables that are created at setup time.

This command is normally executed just once per domain.

While this command creates the default tables, it does not initialize them with data. This is accomplished with the `nisaddent`(1M) command.

It is easier to use the `nisserver`(1M) script to create subdirectories and the default tables.

OPTIONS

`-Y` Specify that the domain will be served as both a NIS+ domain as well as an NIS domain using the backward compatibility flag. This will set up the domain to be less secure by making all the system tables readable by unauthenticated clients as well.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`nis+`(1), `nismkdir`(1), `nistbladm`(1), `nisaddent`(1M), `nisinit`(1M) `nisserver`(1M), `attributes`(5)

NOTES

NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

nisshowcache – NIS+ utility to print out the contents of the shared cache file

SYNOPSIS

```
/usr/lib/nis/nisshowcache [-v]
```

DESCRIPTION

`nisshowcache` prints out the contents of the per-machine NIS+ directory cache that is shared by all processes accessing NIS+ on the machine. By default, `nisshowcache` only prints out the directory names in the cache along with the list of active servers. The shared cache is maintained by `nis_cachemgr`(1M).

OPTIONS

`-v` Verbose mode. Print out the contents of each directory object, including the information on the server name and its universal addresses.

FILES

`/var/nis/NIS_SHARED_DIRCACHE`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`nis_cachemgr`(1M), `syslogd`(1M), `nisfiles`(4), `attributes`(5)

DIAGNOSTICS

Error messages are sent to the `syslogd`(1M) daemon.

NOTES

NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

nisstat – report NIS+ server statistics

SYNOPSIS

`/usr/lib/nis/nisstat` `[-H` *host*`]` `[`*directory*`]`

DESCRIPTION

The `nisstat` command queries a NIS+ server for various statistics about its operations. These statistics may vary between implementations and from release to release. Not all statistics are available from all servers. If you request a statistic from a server that does not support that statistic, it is never a fatal error. The message "unknown statistic" is returned.

By default, statistics are fetched from the server(s) of the NIS+ directory for the default domain. If *directory* is specified, servers for that directory are queried.

Supported statistics for this release are as follows:

root server	This reports whether the server is a root server.
NIS compat mode	This reports whether the server is running in NIS compat mode.
DNS forwarding in NIS mode	This reports whether the server in NIS compat mode will forward host lookup calls to DNS.
security level	This reports the security level of this server.
serves directories	This lists the directories served by this server.
Operations	This statistic returns results in the form: `OP=`*opname*`:C=`*calls*`:E=`*errors*`:T=`*micros* Where *opname* is replaced by the RPC procedure name or operation, *calls* is the number of calls to this procedure that have been made since the server started running. *errors* is the number of errors that have occurred while processing a call, and *micros* is the average time in microseconds to complete the last 16 calls.
Directory Cache	This statistic reports the number of calls to the internal directory object cache, the number of hits on that cache, the number of misses, and the hit rate percentage.
Group Cache	This statistic reports the number of calls to the internal NIS+ group object cache, the number of hits on that cache, the number of misses, and the hit rate percentage.

Static Storage	This statistic reports the number of bytes the server has allocated for its static storage buffers.
Dynamic Storage	This statistic reports the amount of heap the server process is currently using.
Uptime	This statistic reports the time since the service has been running.

Per-server and per-directory access restrictions may apply. See `nisopaccess`(1). `nisstat` uses `NIS_STATUS`.

OPTIONS

`-H` *host* — Normally all servers for the directory are queried. With this option, only the machine named *host* is queried. If the named machine does not serve the directory, no statistics are returned.

ENVIRONMENT VARIABLES

`NIS_PATH` — If this variable is set, and the NIS+ directory name is not fully qualified, each directory specified will be searched until the directory is found. See `nisdefaults`(1).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`nisdefaults`(1), `nisopaccess`(1), `attributes`(5)

NOTES

NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

nisupdkeys – update the public keys in a NIS+ directory object

SYNOPSIS

/usr/lib/nis/nisupdkeys [-a | -C] [-H *host*] [*directory*]

/usr/lib/nis/nisupdkeys -s [-a | -C]-H *host*

DESCRIPTION

This command updates the public keys in an NIS+ directory object. When the public key(s) for a NIS+ server are changed, `nisupdkeys` reads a directory object and attempts to get the public key data for each server of that directory. These keys are placed in the directory object and the object is then modified to reflect the new keys. If *directory* is present, the directory object for that directory is updated. Otherwise the directory object for the default domain is updated. The new key must be propagated to all directory objects that reference that server.

On the other hand, `nisupdkeys -s` gets a list of all the directories served by *host* and updates those directory objects. This assumes that the caller has adequate permission to change all the associated directory objects. The list of directories being served by a given server can also be obtained by `nisstat`(1M). Before you do this operation, make sure that the new address/public key has been propagated to all replicas. If multiple authentication mechanisms are configured using `nisauthconf`(1M), then the keys for those mechanisms will also be updated or cleared.

The user executing this command must have modify access to the directory object for it to succeed. The existing directory object can be displayed with the `niscat`(1) command using the `-o` option.

This command does not update the directory objects stored in the `NIS_COLD_START` file on the NIS+ clients.

If a server is also the root master server, then `nisupdkeys -s` cannot be used to update the root directory.

OPTIONS

`-a` Update the universal addresses of the NIS+ servers in the directory object. Currently, this only works for the TCP/IP family of transports. This option should be used when the IP address of the server is changed. The server's new address is resolved using `getipnodebyname`(3SOCKET) on this machine. The `/etc/nsswitch.conf` file must point to the correct source for *ipnodes* and *hosts* for this resolution to work.

`-C` Specify to clear rather than set the public key(s). Communication with a server that has no public key(s) does not require the use of secure RPC.

`-H` *host* Limit key changes only to the server named *host*. If the hostname is not a fully qualified NIS+ name, then it is assumed to be a host in the default domain. If the named host does not serve the directory, no action is taken.

`-s` Update all the NIS+ directory objects served by the specified server. This assumes that the caller has adequate access rights to

change all the associated directory objects. If the NIS+ principal making this call does not have adequate permissions to update the directory objects, those particular updates will fail and the caller will be notified. If the `rpc.nisd` on *host* cannot return the list of servers it serves, the command will print an error message. The caller would then have to invoke `nisupdkeys` multiple times (as in the first synopsis), once per NIS+ directory that it serves.

EXAMPLES

EXAMPLE 1 Using `nisupdkeys`

The following example updates the keys for servers of the *foo.bar.* domain.

```
example% nisupdkeys foo.bar.
```

This example updates the key(s) for host *fred* that serves the *foo.bar.* domain.

```
example% nisupdkeys -H fred foo.bar.
```

This example clears the public key(s) for host *wilma* in the *foo.bar.* directory.

```
example% nisupdkeys -CH wilma foo.bar.
```

This example updates the public key(s) in all directory objects that are served by the host *wilma*.

```
example% nisupdkeys -s -H wilma
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`chkey`(1), `niscat`(1), `nisaddcred`(1M), `nisauthconf`(1M), `nisstat`(1M), `getipnodebyname`(3SOCKET), `nis_objects`(3NSL), `attributes`(5)

NOTES

NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

nlsadmin – network listener service administration

SYNOPSIS

/usr/sbin/nlsadmin -x

/usr/sbin/nlsadmin [*options*] *net_spec*

/usr/sbin/nlsadmin [*options*] -N *port_monitor_tag*

/usr/sbin/nlsadmin -V

/usr/sbin/nlsadmin -c *cmd* | -o *streamname* [-p *modules*] [-A *address* | -D]

[-R *prognum* : *versnum*]

DESCRIPTION

`nlsadmin` is the administrative command for the network listener process(es) on a machine. Each network has at least one instance of the network listener process associated with it; each instance (and thus, each network) is configured separately. The listener process ''listens'' to the network for service requests, accepts requests when they arrive, and invokes servers in response to those service requests. The network listener process may be used with any network (more precisely, with any connection-oriented transport provider) that conforms to the transport provider specification.

`nlsadmin` can establish a listener process for a given network, configure the specific attributes of that listener, and start and kill the listener process for that network. `nlsadmin` can also report on the listener processes on a machine, either individually (per network) or collectively.

net_spec represents a particular listener process. Specifically, *net_spec* is the relative path name of the entry under `/dev` for a given network (that is, a transport provider). *address* is a transport address on which to listen and is interpreted using a syntax that allows for a variety of address formats. By default, *address* is interpreted as the symbolic ASCII representation of the transport address. An *address* preceded by `\x` will let you enter an address in hexadecimal notation. Note that *address* must appear as a single word to the shell, thus it must be quoted if it contains any blanks.

Changes to the list of services provided by the listener or the addresses of those services are put into effect immediately.

OPTIONS

`nlsadmin` may be used with the following combinations of options and arguments:

`-x`	Report the status of all of the listener processes installed on this machine.
net_spec	Print the status of the listener process for *net_spec* .
`-q` *net_spec*	Query the status of the listener process for the specified network, and reflects the result of that query in its exit code. If a listener process is active, `nlsadmin` will exit with a status of `0`; if no process is active, the exit

	code will be `1`; the exit code will be greater than `1` in case of error.
`-v` *net_spec*	Print a verbose report on the servers associated with *net_spec*, giving the service code, status, command, and comment for each. It also specifies the `uid` the server will run as and the list of modules to be pushed, if any, before the server is started.
`-z` *service_code net_spec*	Print a report on the server associated with *net_spec* that has service code *service_code*, giving the same information as in the `-v` option.
`-q -z` *service_code net_spec*	Query the status of the service with service code *service_code* on network *net_spec*, and exits with a status of `0` if that service is enabled, `1` if that service is disabled, and greater than `1` in case of error.
`-l` *address net_spec*	Change or set the transport address on which the listener listens (the general listener service). This address can be used by remote processes to access the servers available through this listener (see the `-a` option, below). If *address* is just a dash (" – "), `nlsadmin` reports the address currently configured, instead of changing it. A change of address takes effect immediately.
`-t` *address net_spec*	Change or set the address on which the listener listens for requests for terminal service but is otherwise similar to the `-l` option above. A terminal service address should not be defined unless the appropriate remote login software is available; if such software is available, it must be configured as service code 1 (see the `-a` option, below).
`-i` *net_spec*	Initialize an instance of the listener for the network specified by *net_spec;* that is, create and initialize the files required by the listener as well as starting that instance of the listener. Note that a particular instance

of the listener should be initialized only once. The listener must be initialized before assigning addresses or services.

`-a` *service_code*

[`-p` *modules*] [`-w` *name*] `-c` *cmd* `-y` *comment net_spec*

Add a new service to the list of services available through the indicated listener. *service_code* is the code for the service, *cmd* is the command to be invoked in response to that service code, comprised of the full path name of the server and its arguments, and *comment* is a brief (free-form) description of the service for use in various reports. Note that *cmd* must appear as a single word to the shell; if arguments are required, the *cmd* and its arguments must be enclosed in quotation marks. The *comment* must also appear as a single word to the shell. When a service is added, it is initially enabled (see the `-e` and `-d` options, below).

Service codes are alphanumeric strings, and are administered by AT&T. The numeric service codes 0 through 100 are reserved for internal use by the listener. Service code 0 is assigned to the nlps server, which is the service invoked on the general listening address. In particular, code 1 is assigned to the remote login service, which is the service automatically invoked for connections to the terminal login address.

If the `-p` option is specified, then *modules* will be interpreted as a list of STREAMS modules for the listener to push before starting the service being added. The modules are pushed in the order they are specified. *modules* should be a comma-separated list of modules, with no white space included.

If the `-w` option is specified, then *name* is interpreted as the user name from `/etc/passwd` that the listener should look up. From the user name, the listener obtains

the user ID, the group ID(s), and the home directory for use by the server. If `-w` is not specified, the default is to use the user name `listen`.

A service must explicitly be added to the listener for each network on which that service is to be available. This operation will normally be performed only when the service is installed on a machine, or when populating the list of services for a new network.

`-r` *service_code net_spec*
: Remove the entry for the *service_code* from that listener's list of services. This is normally done only in conjunction with the de-installation of a service from a machine.

`-e` *service_code net_spec*
`-d` *service_code net_spec*
: Enable or disable (respectively) the service indicated by *service_code* for the specified network. The service must previously have been added to the listener for that network (see the `-a` option, above). Disabling a service will cause subsequent service requests for that service to be denied, but the processes from any prior service requests that are still running will continue unaffected.

`-s` *net_spec*
`-k` *net_spec*
: Start and kill (respectively) the listener process for the indicated network. These operations are normally performed as part of the system startup and shutdown procedures. Before a listener can be started for a particular network, it must first have been initialized (see the `-i` option, above). When a listener is killed, processes that are still running as a result of prior service requests will continue unaffected.

Under the Service Access Facility, it is possible to have multiple instances of the listener on a single *net_spec*. In any of the above commands, the option `-N` *port_monitor_tag* may be used in place of the *net_spec* argument. This argument specifies the tag by which an instance of the listener is identified by the Service Access Facility. If the `-N` option is not specified (that is, the *net_spec* is specified in the invocation), then it will be assumed that the last component of the *net_spec* represents

the tag of the listener for which the operation is destined. In other words, it is assumed that there is at least one listener on a designated *net_spec*, and that its tag is identical to the last component of the *net_spec*. This listener may be thought of as the primary, or default, listener for a particular *net_spec* .

`nlsadmin` is also used in conjunction with the Service Access Facility commands. In that capacity, the following combinations of options can be used:

`-V`
Write the current version number of the listener's administrative file to the standard output. It is used as part of the `sacadm` command line when `sacadm` adds a port monitor to the system.

`-c` *cmd* | `-o` *streamname* [`-p` *modules*] [`-A` *address* | `-D`] [`-R` *prognum* : *versnum*]
Format the port monitor-specific information to be used as an argument to `pmadm`(1M)

The `-c` option specifies the full path name of the server and its arguments. *cmd* must appear as a single word to the shell, and its arguments must therefore be surrounded by quotes.

The `-o` option specifies the full path name of a FIFO or named STREAM through which a standing server is actually receiving the connection.

If the `-p` option is specified, then *modules* will be interpreted as a list of STREAMS modules for the listener to push before starting the service being added. The modules are pushed in the order in which they are specified. *modules* must be a comma-separated list, with no white space included.

If the `-A` option is specified, then *address* will be interpreted as the server's private address. The listener will monitor this address on behalf of the service and will dispatch all calls arriving on this address directly to the designated service. This option may not be used in conjunction with the `-D` option.

If the `-D` option is specified, then the service is assigned a private address dynamically, that is, the listener will have the transport provider select the address each time the listener begins listening on behalf of this service. For RPC services, this option will be often be used in conjunction with the `-R` option to register the dynamically assigned address with the rpcbinder. This option may not be used in conjunction with the `-A` option.

When the `-R` option is specified, the service is an RPC service whose address, program number, and version number should be registered with the rpcbinder for this transport provider. This registration is performed each time the listener begins listening on behalf of the service. *prognum* and *versnum* are the program number and version number, respectively, of the RPC service.

`nlsadmin` may be invoked by any user to generate reports; all operations that affect a listener's status or configuration may only be run by a super-user.

The options specific to the Service Access Facility may not be used together with any other options.

ERRORS

If successful, `nlsadmin` exits with a status of 0. If `nlsadmin` fails for any reason, it exits with a status greater than or equal to 2. See `-q` option for a return status of 1.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`listen`(1M), `pmadm`(1M), `rpcbind`(1M), `sacadm`(1M), `attributes`(5)

System Administration Guide: Basic Administration

NOTES

Dynamically assigned addresses are not displayed in reports as statically assigned addresses are.

NAME

nscd – name service cache daemon

SYNOPSIS

```
/usr/sbin/nscd [-f configuration-file] [-g] [-e cachename, yes | no]
     [-i cachename]
```

DESCRIPTION

`nscd` is a process that provides a cache for the most common name service requests. It starts up during multi-user boot. The default *configuration-file* `/etc/nscd.conf` determines the behavior of the cache daemon. See `nscd.conf`(4).

`nscd` provides caching for the `passwd`(4), `group`(4), `hosts`(4), `ipnodes`(4), `exec_attr`(4), `prof_attr`(4), and `user_attr`(4) databases through standard `libc` interfaces, such as `gethostbyname`(3NSL), `getipnodebyname`(3SOCKET), `gethostbyaddr`(3NSL), and others. Each cache has a separate time-to-live for its data; modifying the local database (`/etc/hosts`, `/etc/resolv.conf`, and so forth) causes that cache to become invalidated upon the next call to `nscd`. The shadow file is specifically not cached. `getspnam`(3C) calls remain uncached as a result.

`nscd` also acts as its own administration tool. If an instance of `nscd` is already running, commands are passed to the running version transparently.

In order to preserve NIS+ security, the startup script for `nscd` (`/etc/init.d/nscd`) checks the permissions on the `passwd` table if NIS+ is being used. If this table cannot be read by unauthenticated users, then `nscd` will make sure that any encrypted password information returned from the NIS+ server is supplied only to the owner of that password.

OPTIONS

Several of the options described below require a *cachename* specification. Supported values are `passwd`, `group`, `hosts`, `ipnodes`, `exec_attr`, `prof_attr`, and `user_attr`.

`-f` *configuration-file*	Causes `nscd` to read its configuration data from the specified file.
`-g`	Prints current configuration and statistics to standard output. This is the only option executable by non-root users.
`-e` *cachename*`, yes|no`	Enables or disables the specified cache.
`-i` *cachename*	Invalidate the specified cache.

EXAMPLES

EXAMPLE 1 Stopping and restarting the `nscd` daemon.

```
example# /etc/init.d/nscd stop

example# /etc/init.d/nscd start
```

FILES

`/etc/nscd.conf` determines athe behavior of the cache daemon

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`getspnam`(3C), `gethostbyname`(3NSL), `getipnodebyname`(3SOCKET), `exec_attr`(4), `group`(4), `hosts`(4), `ipnodes`(4), `nscd.conf`(4), `nsswitch.conf`(4), `passwd`(4), `prof_attr`(4), `user_attr`(4), `attributes`(5)

NOTES

The output from the `-g` option to `nscd` is subject to change. Do not rely upon it as a programming interface.

NAME

nslookup – query name servers interactively

SYNOPSIS

nslookup [– option]… *host* [*server*]

nslookup [– option]… - [*server*]

nslookup

DESCRIPTION

nslookup sends queries to Internet domain name servers. It has two modes: interactive and non-interactive. Interactive mode allows the user to contact servers for information about various hosts and domains or to display a list of hosts in a domain. Non-interactive mode is used to display just the name and requested information for a host or domain.

OPTIONS

-option Set the permissible options, as shown in the following list. These are the same options that the set command supports in interactive mode (see set in the Commands section for more complete descriptions).

all	List the current settings
class=*classname*	Restrict search according to the specified class
d2	Set exhaustive debug mode on
nod2	Set exhaustive debug mode off
debug	Set debug mode on
nodebug	Set debug mode off
defname	Set domain-appending mode on
nodefname	Set domain-appending mode off
domain=*string*	Establish the appendable domain
ignoretc	Set it to ignore packet truncation errors
noignoretc	Set it to acknowledge packet truncation errors

OPERANDS

host Inquires about the specified *host*. In this non-interactive command format, nslookup Does not prompt for additional commands.

– Causes nslookup to prompt for more information, such as host names, before sending one or more queries.

server Directs inquiries to the name server specified here in the command line rather than the one read from the /etc/resolv.conf file (see resolv.conf(4)). *server* can be either a name or an Internet address. If the specified host cannot be reached, nslookup resorts to using the name server specified in /etc/resolv.conf.

Non-interactive Mode

Non-interactive mode is selected when the name or Internet address of the host to be looked up is given as the first argument.

Within non-interactive mode, space-separated options can be specified. They must be entered before the host name, to be queried. Each option must be prefixed with a hyphen.

For example, to request extensive host information and to set the timeout to 10 seconds when inquiring about `gypsy`, enter:

```
example% nslookup –query=hinfo –timeout=10 gypsy
```

To avoid repeated entry of an option that you almost always use, place a corresponding `set` command in a `.nslookuprc` file located inside your home directory. (See `Commands` for more information about `set`.) The `.nslookuprc` file can contain several `set` commands if each is followed by a RETURN.

Entering and Leaving Interactive Mode

Interactive mode is selected when

- No arguments are supplied.
- A '–' (hyphen) character is supplied as the *host* argument.

To exit from an interactive `nslookup` session, type `Control-d` or type the command `exit` followed by RETURN.

Supported Command Interactions

The commands associated with interactive mode are subject to various limitations and run-time conventions.

The maximum length of a command line is 255 characters. When the RETURN key is pressed, command-line execution begins. While a command is running, its execution can be interrupted by typing `Control-c`.

The first word entered on the command line must be the name of a `nslookup` command unless you wish to enter the name of a host to inquire about. Any unrecognized command is handled as a host name to inquire about. To force a command to be treated as a host name to be inquired about, precede it with a backslash character.

Commands

`exit`
Exit the `nslookup` program.

`help`

`?`
Display a brief summary of commands.

`host` [*server*]
Look up information for *host* using the current default server, or using *server* if it is specified.

If the *host* supplied is an Internet address and the query type is `A` or `1PTR`, the name of the host is returned. If the *host* supplied is a name and it does not have a trailing period, the default domain name is appended to the name. (This behavior depends on the state of the `set` options `domain`, `srchlist`, `defname`, and `search`).

To look up a host that is not in the current domain, append a period to the name.

`finger` [*name*] [>> *filename*]
Connect with the finger server on the current host, which is defined by the most recent successful host lookup.

If no *name* value is specified, a list of login account names on the current host is generated.

Similar to a shell command interpreter, output can be redirected to a file using the usual redirection symbols: > and >>.

`ls` [-*options*] *domain* [>> *filename*]
List the information available for *domain*, optionally creating or appending to *filename*. The default output contains host names and their Internet addresses.

Output can be redirected to *filename* using the > and >> redirection symbols. When output is directed to a file, hash marks are shown for every 50 records received from the server. The permissible values for *options* are:

`a`	Lists aliases of hosts in the domain. This is a synonym for the command `ls -t CNAME`.
`d`	Lists all records for the domain. This is a synonym for the command `ls -t ANY`.
`h`	Lists CPU and operating system information for the domain. This is a synonym for the command `ls -t HINFO`.
`s`	Lists well-known services of hosts in the domain. This is a synonym for the command `ls -t WKS`.
`t` *querytype-value*	lists all records of the specified type (see `querytype` within the discussion of the `set` command).

`set` *token*=*value*
`set` *keyword*
Establish a preferred mode of search operation. Permissible *token* and *keyword* values are:

`all`	Display the current values of frequently-used options. Information about the current default server and host is also displayed.
`cl[ass]`=*classname*	Limit the search according to the protocol group (*classname*) for which lookup information is desired. Permissible *classname* values are:

`ANY`	A wildcard selecting all classes
`IN`	The Internet class (the default)

	`CHAOS` The Chaos class.
	`HESIOD` The MIT Athena Hesiod class.
`d2`	
`nod2`	Enable or disable exhaustive debugging mode. Essentially all fields of every packet are displayed. By default, this option is disabled.
`deb[ug]`	
`nodeb[ug]`	Enable or disable debugging mode. When debugging mode is enabled, much more information is produced about the packet sent to the server and the resulting answer. By default, this option is disabled.
`def[name]`	
`nodef[name]`	Enable or disable appending the default domain name to a single-component lookup request (one that lacks a dot). By default, this option is enabled for `nslookup`. The default value for the domain name is the value given in `/etc/resolv.conf`, unless: there is an environmental value for `LOCALDOMAIN` when `nslookup` is run; a recent value has been specified through the `srchlist` command or the `set domain` command.
`do[main]=`*string*	Change the default domain name to be appended to all lookup requests to *string*. For this option to have any effect, the `defname` option must also be enabled and the `search` option must be set in a compatible way. The domain search list contains the parents of the default domain if it has at least two components in its name. For example, if the default domain is `CC.Berkeley.EDU`, the search list is `CC.Berkeley.EDU` and `Berkeley.EDU`. Use the `set srchlist` command to specify a different list. Use the `set all` command to display the list.
`ignoretc`	
`noignoretc`	Ignore packet truncation errors. By default, this option is disabled.

`srch[list]=`*name1/name2/. . .*

Change the default domain name to *name1* and the domain search list to *name1*, *name2*, etc. A maximum of 6 names can be specified, along with slash characters to separate them. For example,

```
example% set srchlist=lcs.MIT.EDU/ai.MIT.EDU/MIT.EDU
```

sets the domain to `lcs.MIT.EDU` and the search list to all three names. This command overrides the default domain name and search list of the `set domain` command. Use the `set all` command to display the list.

`search`
`nosearch`
Enable or disable having the domain names in the domain search list appended to the request, generating a series of lookup queries if necessary until an answer is received. To take effect, the lookup request must contain at least one dot (period); yet it must not contain a trailing period. By default, this option is enabled.

`po[rt]=`*value*
Specify the default TCP/UDP name server port. By default, this value is `53`.

`q[uerytype]=`*value*
`ty[pe]=`*value*
Change the type of information returned from a query to one of:

`A`	The Internet address of the host
`CNAME`	The canonical name for an alias
`HINFO`	The host CPU and operating system type
`MD`	The mail destination
`MX`	The mail exchanger
`MB`	The mailbox domain name
`MG`	The mail group member
`MINFO`	The mailbox or mail list information
`NS`	The name server
`PTR`	The host name if the query is in the form of an Internet address; otherwise the pointer to other information
`SOA`	The domain's start-of-authority information
`TXT`	The text information
`UINFO`	The user information
`WKS`	The supported well-known services

(Other types specified in the *RFC 1035* document are valid, but they are not as useful.)

`recurse`
`norecurse`
Enable or disable having to query other name servers before abandoning a search. By default, this feature is enabled.

`ret[ry]=`*count*
Set the maximum number of times to retry a request before abandoning a search. When a reply to a request is not received within a certain amount of time (changed

with `set timeout`), the timeout period is doubled and the request is resent. The retry value controls how many times a request is resent before the request is aborted. The default for *count* is `4`.

`ro[ot]=`*host*
Change the name of the root server to *host*. This affects the `root` command. The default root server is `ns.internet.net`.

`t[timeout]=`*interval*
Change the amount of time to wait for a reply to *interval* seconds. Each retry doubles the timeout period. The default *interval* is `5` seconds.

`vc`
`novc`
Enable or disable the use of a virtual circuit when sending requests to the server. By default, this feature is disabled.

`root`
Change the default server to the server for the root of the domain name space. Currently, the host `ns.internic.net` is used; this command is a synonym for `server ns.internic.net`. The name of the root server can be changed with the `set root` command.

`server` *domain*
`lserver` *domain*
Change the default server to *domain*. `lserver` uses the initial server to look up information about *domain* while `server` uses the current default server. If an authoritative answer can not be found, the names of servers that might have the answer are returned.

EXAMPLES

EXAMPLE 1 Searching the Internet Domain Namespace

To effectively search the Internet domain namespace, it helps to know its structure. At present, the Internet domain name-space is tree-structured, with one top level domain for each country except the United States.. There are also some traditional top level domains, not explicitly tied to any particular country. These include:

`COM`	Commercial establishments
`EDU`	Educational institutions
`ORG`	Not-for-profit organizations
`GOV`	Government agencies
`MIL`	MILNET hosts

If you are looking for a specific host, you need to know something about the host's organization in order to determine the top-level domain that it belongs to. For instance, if you want to find the Internet address of a machine at UCLA, do the following:

- Connect with the root server using the `root` command. The root server of the name space has knowledge of the top-level domains.

EXAMPLE 1 Searching the Internet Domain Namespace *(Continued)*

- Since UCLA is a university, its domain name is `ucla.edu`. Connect with a server for the `ucla.edu` domain with the command `server ucla.edu`. The response produces the names of hosts that act as servers for that domain. Note: the root server does not have information about `ucla.edu`, but knows the names and addresses of hosts that do. Once located by the root server, all future queries will be sent to the UCLA name server.
- To request information about a particular host in the domain (for instance, `locus`), just type the host name. To request a listing of hosts in the UCLA domain, use the `ls` command. The `ls` command requires a domain name, (in this case, `ucla.edu`, as an argument.

If you are connected with a name server that handles more than one domain, all lookups for host names must be fully specified with its domain. For instance, the domain `harvard.edu` is served by `seismo.css.gov`, which also services the `css.gov` and `cornell.edu` domains. A lookup request for the host `aiken` in the `harvard.edu` domain must be specified as `aiken.harvard.edu`. However, the `set domain=`*name* and `set defname` commands can be used to automatically append a domain name to each request.

After a successful lookup of a host, use the `finger`(1) command to see who is on the system, or to finger a specific person. (`finger` requires the type to be `A`.)

To get other information about the host, use the `set querytype=`*value* command to change the type of information desired and request another lookup.

ENVIRONMENT VARIABLES

`HOSTALIASES` References the file containing host aliases

`LOCALDOMAIN` Overrides default domain

EXIT STATUS

The process returns the following values:

`0` On success.

`1` On failure.

FILES

`/etc/resolv.conf`
Initial domain name and name server addresses

`$HOME/.nslookuprc`
Initial option commands

`/usr/lib/nslookup.help`
Summary of commands

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu
Interface Stability	Standard BIND 8.2.4

SEE ALSO

`finger`(1), `more`(1), `in.named`(1M), `nstest`(1M), `resolver`(3RESOLV), `resolv.conf`(4), `attributes`(5)

Mockapetris, Paul. *RFC 1034, Domain Names - Concepts and Facilities*. Network Working Group. November 1987.

Mockapetris, Paul. *RFC 1035, Domain Names - Implementation and Specification*. Network Working Group. November 1987.

DIAGNOSTICS

If the lookup request is successful, an error message is produced. Possible errors are:

`Timed out`
: The server did not respond to a request after a certain amount of time (changed with `set timeout=`*value*) and a certain number of retries (changed with `set retry=`*value*).

`No response from server`
: No name server is running on the server machine.

`No records`
: The server does not have resource records of the current query type for the host, although the host name is valid. The query type is specified with the `set querytype` command.

`Non-existent domain`
: The host or domain name does not exist.

`Connection refused`
`Network is unreachable`
: The connection to the name or finger server can not be made at the current time. This error commonly occurs with `ls` and `finger` requests.

`Server failure`
: The name server found an internal inconsistency in its database and could not return a valid answer.

`Refused`
: The name server refused to service the request.

`Format error`
: The name server found that the request packet was not in the proper format. This may indicate an error in `nslookup`.

NAME

nstest – DNS test shell

SYNOPSIS

nstest [-d] [-i] [-r] [-v] [-p *port*] [*inet_addr* [*logfile*]]

DESCRIPTION

nstest is an interactive DNS test program. Queries are formed and sent by user command; any reply received is printed on the standard output. *inet_addr* is the Internet address of the DNS resolver to which nstest should send its queries. If *inet_addr* is not included, nstest first tries to contact a DNS server on the local host; if that fails, it tries the servers listed in the /etc/resolv.conf file. If a *logfile* is supplied, nstest uses it to log the queries sent and replies received.

OPTIONS

-d Causes nstest to create a file named ns_packet.dump (if it does not exist) and write into it a raw (binary) copy of each packet sent. If ns_packet.dump does exist, nstest will truncate it.

-i Sets the RES_IGNTC flag on the queries it makes. See resolver(3RESOLV) for a description of the RES_IGNTC flag.

-r Turns off the RES_RECURSE flag on the queries it makes. See resolver(3RESOLV) for a description of the RES_RECURSE flag.

-v Turns on the RES_USEVC and RES_STAYOPEN flags on the res_send () calls made. See resolver(3RESOLV) for a description of the RES_USEVC and RES_STAYOPEN flags.

-p Causes nstest to use the supplied *port* instead of the default name server port.

USAGE

When nstest starts, it prints a prompt (">") and waits for user input. DNS queries are formed by typing a *key letter* followed by the appropriate *argument*. Each *key letter* results in a call to res_mkquery () with *op* set to either IQUERY or QUERY and type set to one of the type values (defined in <arpa/nameser.h>). (Any other key letter than those listed below causes nstest to print a summary of the following table.)

Key Letter & Argument	Op	Type
a*host*	QUERY	T_A
A*addr*	IQUERY	T_A
B*user*	QUERY	T_MG
b*user*	QUERY	T_MB
c*host*	QUERY	T_CNAME
f*host*	QUERY	T_UINFO

G*gid*	IQUERY	T_GID
g*host*	QUERY	T_GID
h*host*	QUERY	T_HINFO
i*host*	QUERY	T_MINFO
M*host*	QUERY	T_MAILB
m*host*	QUERY	T_MX
n*host*	QUERY	T_NS
p*host*	QUERY	T_PTR
r*host*	QUERY	T_MR
s*host*	QUERY	T_SOA
T*host*	QUERY	T_TXT
U*uid*	IQUERY	T_UID
u*host*	QUERY	T_UID
w*host*	QUERY	T_WKS
x*host*	QUERY	T_AXFR

After the query is successfully formed, `res_send ()` is called to send it and wait for a reply. `nstest` then prints the following on the standard output:

- a summary of the request and reply packets, including the `HEADER` structure (defined in `<arpa/nameser.h>`) used in the request
- the question being asked of the name server
- an enumeration of the name server(s) being polled
- a summary of the `HEADER` structure received in the reply
- the question the name server answered
- the answer itself

EXAMPLES

EXAMPLE 1 Fetching the address of host `playground.sun.com` from the Sun name server.

To fetch the address of host `playground.sun.com` from the Sun name server, the user would enter:

```
$ nstest 192.9.5.1
> aplayground.sun.com
```

The utility `nstest` would return the following:

```
res_mkquery(0, playground.sun.com, 1, 1)
res_send()
HEADER:
```

EXAMPLE 1 Fetching the address of host `playground.sun.com` from the Sun name server. *(Continued)*

```
        opcode = QUERY, id = 1, rcode = NOERROR
        header flags:  rd
        qdcount = 1, ancount = 0, nscount = 0, arcount = 0

QUESTIONS:
        playground.sun.com, type = A, class = IN

Querying server (# 1) address = 192.9.5.1
got answer:
HEADER:
        opcode = QUERY, id = 1, rcode = NOERROR
        header flags:  qr aa rd ra
        qdcount = 1, ancount = 1, nscount = 0, arcount = 0

QUESTIONS:
        playground.sun.com, type = A, class = IN
ANSWERS:
        playground.sun.com
        type = A, class = IN, ttl = 1 day, dlen = 4
        internet address = 192.9.5.5
```

EXAMPLE 2 Looking up a PTR record.

To look up a PTR record, enter:

```
$ nstest 192.9.5.1
> p5.5.9.192.in-addr.arpa
```

The utility `nstest` would return the following:

```
res_mkquery(0, 5.5.9.192.in-addr.arpa, 1, 12)
res_send()
HEADER:
        opcode = QUERY, id = 2, rcode = NOERROR
        header flags:  rd
        qdcount = 1, ancount = 0, nscount = 0, arcount = 0

QUESTIONS:
        5.5.9.192.in-addr.arpa, type = PTR, class = IN

Querying server (# 1) address = 192.9.5.1
got answer:
HEADER:
        opcode = QUERY, id = 2, rcode = NOERROR
        header flags:  qr aa rd ra
        qdcount = 1, ancount = 1, nscount = 0, arcount = 0

QUESTIONS:
        5.5.9.192.in-addr.arpa, type = PTR, class = IN

ANSWERS:
        5.5.9.192.in-addr.arpa
        type = PTR, class = IN, ttl = 7 hours 47 mins 2 secs, dlen = 23
        domain name = playground.sun.com
```

EXAMPLE 2 Looking up a PTR record. *(Continued)*

FILES

`/usr/include/arpa/nameser.h`	include file for implementation of DNS protocol
`/usr/include/resolv.h`	include file for the resolver daemon (in.named)

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`nslookup`(1M), `resolver`(3RESOLV), `attributes`(5)

NAME

nsupdate – update Internet name servers interactively

SYNOPSIS

nsupdate [-k *keydir*:*keyname*] [-d] [-v] [*filename*]

DESCRIPTION

The nsupdate program can be used to update Internet domain name servers that support dynamic update. nsupdate uses the DNS resolver library to pass messages to the DNS server requesting the addition or deletion of DNS resource records (RR's). nsupdate reads input from *filename* or from standard input.

OPTIONS

nsupdate supports the following options:

-d	Debug mode.
-k	Sign updates with Secret Key Transaction Authentication for DNS (TSIG).
-v	Virtual circuit. Make use of TCP to communicate with the server. The default is UDP.

USAGE

nsupdate reads input records, one per line. Each line contributes a resource record to an update request. All domain names used in a single update request must belong to the same DNS zone. Updates are sent ot the master server as defined in the SOA MNAME field. A blank line causes the accumulated records to be formatted into a single update request and transmitted to the zone's authoritative name servers. Additional records may follow, which are formed into additional, but completely independent, update requests. End the input with a blank line in order to transmit the last request.

Records take one of two general forms. Prerequisite records specify conditions that must be satisfied before the request will be processed. Update records specify changes to be made to the DNS database. An update request consists of zero or more prerequisites and one or more updates. Each update request is processed atomically. All prerequisites must be satisfied, then all updates will be performed.

nsupdate understands the following input record formats:

```
prereq nxdomain domain-name
```

This format requires that no RR of any type exist with name *domain-name*.

```
prereq yxdomain domain-name
```

This format requires that at least one RR names *domain-name* must exist.

```
prereq nxrrset domain-name [class] type
```

This format requires that no RR exist of the specified type and *domain-name*.

```
prereq yxrrset domain-name [class] type [data ...]
```

This format requires that an RR exist of the specified type and *domain-name*. If data is specified, it must match exactly.

```
update delete domain-name [class] [type [data ...]]
```

This format deletes RR's names *domain-name*. If `type` (and possibly `data`) are specified, only matching records will be deleted.

```
update add domain-name ttl [class] type data ...
```

This format adds a new RR of specified `ttl`, `type` and `data`.

EXAMPLES

EXAMPLE 1 Using `nsupdate` Interactively To Change an IP Address

The following example shows the interactive use of `nsupdate` to change an IP address. It deletes any existing `A` records for a domain name and then inserts a new address. Since no prerequisites are specified, the new record will be added even if there are no existing records to delete. A trailing blank line is required to process the request.

```
example$ nsupdate
>update delete test.example.com A
>update add test.example.com 3600 A 10.1.1.1
>
```

EXAMPLE 2 Using `nsupdate` Interactively to Add a `CNAME` (Alias)

In the following example, a `CNAME` (alias) is added to the database only if there are no existing `A` or `CNAME` records for the domain name.

```
example$ nsupdate
>prereq nxrrset www.example.com A
>prereq nxrrset www.example.com CNAME
>update add www.example.com 3600 CNAME test.test.com
>
```

EXAMPLE 3 Using `nsupdate` Interactively With a Key

In the following example, `nsupdate` is signed with the key `mykey`, which is located in the directory `/var/named/keys`.

```
example$ nsupdate -k /var/named/keys:mykey
>update add ftp.example.com 60 A 192.168.5.1
>
```

FILES

`/etc/resolv.conf` Initial domain name and name server addresses.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu
Interface Stability	Standard BIND 8.2.4

SEE ALSO

`in.named`(1M),`resolver`(3RESOLV), `resolv.conf`(4), `attributes`(5)

Mockapetris, P. *RFC 1034, Domain Names - Concepts and Facilities*. Network Working Group. November 1987.

Mockapetris, P. *RFC 1035, Domain Names - Implementation and Specification*. Network Working Group. November 1987.

Vixie, P., editor, Thomson, S., Rekhter, Y. Bound, J. *RFC 2136, Dynamic Updates in the Domain Name System (DNS)*. Network Working Group. April 1997.

DIAGNOSTICS

`Qq send error`
: This message typically indicates that authoritative name servers could not be reached.

`Qq failed update packet`
: This message typically indicates that the name server has rejected the update. Either the name server does not support dynamic update, or there was an authentication failure.

`Qq res_mkupdate: packet size = size`
: If this is the only message sent, it indicates that the update was received and authenticated by the name server. However, the prerequisites may have prevented the update from being performed. Use debug mode (the `-d` option) to examine the status field in the name server's reply and determine if the update was performed.

NAME

ntpdate – set the date and time by way of NTP

SYNOPSIS

/usr/sbin/ntpdate [-bBdoqsuv] [-a *key#*] [-e *authdelay*] [-k *keyfile*] [-m] [-o *version*] [-p *samples*] [-t *timeout*] [-w] *server...*

DESCRIPTION

The `ntpdate` utility sets the local date and time. To determine the correct time, it polls the Network Time Protocol (NTP) servers on the hosts given as arguments. This utility must be run as root on the local host. It obtains a number of samples from each of the servers and applies the standard NTP clock filter and selection algorithms to select the best of these.

The reliability and precision of `ntpdate` improve dramatically with a greater number of servers. While a single server may be used, better performance and greater resistance to inaccuracy on the part of any one server can be obtained by providing at least three or four servers, if not more.

The `ntpdate` utility makes time adjustments in one of two ways. If it determines that your clock is off by more than 0.5 seconds it simply steps the time by calling `gettimeofday`(3C). If the error is less than 0.5 seconds, by default, it slews the clock's time with the offset, by way of a call to `adjtime`(2). The latter technique is less disruptive and more accurate when the offset is small; it works quite well when `ntpdate` is run by `cron` every hour or two. The adjustment made in the latter case is actually 50% larger than the measured offset. This adjustment tends to keep a badly drifting clock more accurate, at some expense to stability. This tradeoff is usually advantageous. At boot time, however, it is usually better to step the time. This can be forced in all cases by specifying the `-b` option on the command line.

The `ntpdate` utility declines to set the date if an NTP server daemon like `xntpd`(1M) is running on the same host. It can be run on a regular basis from `cron`(1M) as an alternative to running a daemon. Doing so once every one to two hours results in precise enough timekeeping to avoid stepping the clock.

OPTIONS

The following options are supported:

`-a` *key#* — Authenticate transactions, using the key number, *key#*.

`-b` — Step the time by calling `gettimeofday`(3C).

`-B` — Force the time to always be slewed using the `adjtime`(2) system call, even if the measured offset is greater than +-128 ms. The default is to step the time using `settimeofday`(3C) if the offset is greater than +-128 ms. If the offset is much greater than +-128 ms in this case, that it can take a long time (hours) to slew the clock to the correct value. During this time the host should not be used to synchronize clients.

`-d` — Display what will be done without actually doing it. Information useful for general debugging is also printed.

`-e` *authdelay* — Specify an authentication processing delay, *authdelay* in seconds. See `xntpd`(1M) for details. This number is usually small enough to

be negligible for purposes of `ntpdate`. However, specifying a value may improve timekeeping on very slow CPU's.

`-k` *keyfile* — Read keys from the file *keyfile* instead of the default file, `/etc/ntp.keys`. *keyfile* should be in the format described in `xntpd`(1M).

`-m` — Join multicast group specified in *server* and synchronize to multicast NTP packets. The standard NTP group is 224.0.1.1.

`-o` *version* — Force the program to poll as a version 1 or version 2 implementation. By default `ntpdate` claims to be an NTP version 3 implementation in its outgoing packets. However, some older software declines to respond to version 3 queries. This option can be used in these cases.

`-p` *samples* — Set the number of samples `ntpdate` acquires from each server. *samples* can be between 1 and 8 inclusive. The default is 4.

`-q` — Query only. Do not set the clock.

`-s` — Log actions by way of the `syslog`(3C) facility rather than to the standard output — a useful option when running the program from `cron`(1M).

`-t` *timeout* — Set the time `ntpdate` spends, waiting for a response. *timeout* is rounded to a multiple of 0.2 seconds. The default is 1 second, a value suitable for polling across a LAN.

`-u` — Use an unprivileged port to send the packets from. This option is useful when you are behind a firewall that blocks incoming traffic to privileged ports, and you want to synchronize with hosts beyond the firewall. The `-d` option always uses unprivileged ports.

`-v` — Be verbose. This option causes `ntpdate`'s version identification string to be logged.

`-w` — Wait until able to synchronize with a server. When the `-w` option is used together with `-m`, `ntpdate` waits until able to join the group and synchronize.

FILES

`/etc/inet/ntp.keys` — Contains the encryption keys used by `ntpdate`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWntpu

SEE ALSO `cron`(1M), `xntpd`(1M), `adjtime`(2), `gettimeofday`(3C), `settimeofday`(3C)`syslog`(3C), `attributes`(5)

NOTES The technique of compensating for clock oscillator errors to improve accuracy is inadequate. However, to further improve accuracy would require the program to save *state* from previous runs.

NAME

ntpq – standard Network Time Protocol query program

SYNOPSIS

/usr/sbin/ntpq [-inp] [-c *command*] [*host*] [...]

DESCRIPTION

ntpq queries NTP servers which implement the recommended NTP mode 6 control message format, about current state. It can also request changes in that state. The program can be run in interactive mode; or it can be controlled using command line arguments. Requests to read and write arbitrary variables can be assembled, with raw and pretty-printed output options available. By sending multiple queries to the server, ntpq can also obtain and print a list of peers in a common format.

If one or more request options are included on the command line, ntpq sends each of the requests to NTP servers running on each of the hosts given as command line arguments. By default, ntpq sends its requests to localhost, if hosts are not included on the command line. If no request options are given, ntpq attempts to read commands from the standard input and execute them on the NTP server running on the first host given on the command line. Again, ntpq defaults to localhost if no other host is specified.

ntpq uses NTP mode 6 packets to communicate with an NTP server. Thus, it can be used to query any compatible server on the network that permits queries. Since NTP is a UDP protocol, this communication will be somewhat unreliable, especially over large distances. ntpq makes one attempt to retransmit requests; requests timeout if the remote host is not heard from within a suitable period.

OPTIONS

Command line options are described below. Specifying a command line option other than -i or -n causes the specified query (queries) to be sent, immediately to the indicated host(s). Otherwise, ntpq attempts to read interactive format commands from standard input.

-c Interpret the next argument as an interactive format command and add it to the list of commands to be executed on the specified host(s). Multiple -c options may be given.

-i Operate in interactive mode; write prompts to standard output and read commands from standard input.

-n Output all host addresses in dotted-quad numeric format rather than converting them to canonical host names.

-p Print a list of the peers known to the server as well as a summary of their state. This is equivalent to the peers interactive command. See USAGE below.

USAGE

Interactive format commands consist of a keyword followed by up to four arguments. Only enough characters of the full keyword to uniquely identify the command need be typed. Normally, the output of a command is sent to standard output; but this output may be written to a file by appending a '>', followed by a file name, to the command line.

Interactive Commands

A number of interactive format commands are executed entirely within the `ntpq` program itself. They do not result in NTP mode 6 requests being sent to a server. If no request options are included on the command line, and if the standard input is a terminal device, `ntpq` prompts for these commands. The interactive commands are described below:

`? [` *command_keyword* `]`
A '?' by itself prints a list of all the command keywords known to the current version of `ntpq`. A '?' followed by a command keyword prints function and usage information about the command.

`timeout` *milliseconds*
Specifies a time out period for responses to server queries. The default is about 5000 milliseconds. Since `ntpq` retries each query once after a time out, the total waiting time for a time out is twice the time out value that is set.

`delay` *milliseconds*
Specifies a time interval to be added to timestamps included in requests which require authentication. This command is used to enable (unreliable) server reconfiguration over long delay network paths or between machines whose clocks are unsynchronized. Currently, the server does not require time stamps in authenticated requests. Thus, this command may be obsolete.

`host hostname`
Set the name of the host to which future queries are to be sent. *Hostname* may be either a host name or a numeric address.

`keyid #`
Specify of a key number to be used to authenticate configuration requests. This number must correspond to a key number the server has been configured to use for this purpose.

`passwd`
Prompts the user to type in a password which will be used to authenticate configuration requests. If an authenticating key has been specified (see `keyid` above), this password must correspond to this key. `ntpq` does not echo the password as it is typed.

`hostnames yes | no`
If "yes" is specified, host names are printed in information displays. If "no" is given, numeric addresses are printed instead. The default is "yes" unless modified using the command line `-n` switch.

`raw`
Print all output from query commands exactly as it is received from the remote server. The only formatting/filtering done on the data is to transform non- ASCII data into printable form.

`cooked`
Causes output from query commands to be "cooked". The values of variables recognized by the server are reformatted, so that they can be more easily read. Variables which `ntpq` thinks should have a decodable value, but do not, are marked with a trailing '?'.

`ntpversion` [`1` | `2` | `3`]
Sets the NTP version number which `ntpq` claims in packets (defaults is 3). Note that mode 6 control messages (and modes, for that matter) did not exist in NTP version 1. There appear to be no servers left which demand version 1.

`authenticate` [`yes` | `no`]
The command `authenticate yes` instructs `ntpq` to send authentication with all requests it makes. Normally `ntpq` does not authenticate requests unless they are write requests. Authenticated requests cause some servers to handle requests slightly differently, and can occasionally cause a slowed response if you turn authentication on before doing a peer display. `addvars` *variable_name*[=*value*] [,. . .] `rmvars` *variable_name* [,. . .] `clearvars`

The data carried by NTP mode 6 messages consists of a list of items of the form

variable_name=*value*where the "=*value*" is ignored, and can be omitted, in requests to the server to read variables. `ntpq` maintains an internal list in which data to be included in control messages can be assembled, and sent. This is accomplished with the `readlist` and `writelist` commands described below. The `addvars` command allows variables and their optional values to be added to the list. If more than one variable is to be added, the list should be comma-separated, and it should not contain white space. The `rmvars` command can be used to remove individual variables from the list; the `clearlist` command removes all variables from the list.

`debug` [`more` | `less` | `off`]
Turns internal query program debugging on and off.

`quit`
Exit `ntpq`.

Control Message Commands

Each peer known to an NTP server has a 16 bit integer *association identifier* assigned to it. NTP control messages which carry peer variables must identify the peer that the values correspond to, by including its association ID. An association ID of 0 is special. It indicates the variables are system variables, whose names are drawn from a separate name space.

Control message commands send one or more NTP mode 6 messages to the server, and cause the data returned to be printed in some format. Most commands currently implemented send a single message and expect a single response. The current exceptions are the `peers` `mreadlist` and `mreadvar` commands. The `peers` command sends a preprogrammed series of messages to obtain the data it needs. The `mreadlist` and `mreadvar` commands, iterate over a range of associations.

Control message commands are described below:

`associations`
Obtains and prints a list of association identifiers and peer statuses for in-spec peers of the server being queried. The list is printed in columns. The first of these is an index that numbers the associations from 1, for internal use. The second column contains the actual association identifier returned by the server and the third the status word for the peer. This is followed by a number of columns containing data decoded from the status word. Note that the data returned by the `associations` command is cached internally in `ntpq`. The index is then of use when dealing with "dumb" servers which use association identifiers that are hard for humans to type. For any subsequent commands which require an association identifier as an argument, the identifier can be specified by using the form, *&index*. Here *index* is taken from the previous list.

`lassociations`
Obtains and prints a list of association identifiers and peer statuses for all associations for which the server is maintaining *state*. This command differs from the `associations` command only for servers which retain *state* for out-of-spec client associations. Such associations are normally omitted from the display when the `associations` command is used, but are included in the output of `lassociations`.

`passociations`
Prints association data concerning in-spec peers from the internally cached list of associations. This command performs identically to the `associations` command except that it displays the internally stored data rather than making a new query.

`lpassociations`
Print data for all associations, including out-of-spec client associations, from the internally cached list of associations. This command differs from `passociations` only when dealing with servers which retain *state* for out-of-spec client associations.

`pstatus` *assocID*
Sends a read status request to the server for the given association. The names and values of the peer variables returned will be printed. Note that the status word from the header is displayed preceding the variables, both in hexadecimal and in pigeon English.

`readvar` [*assoc*] [*variable_name*[=*value*] [*,. . .*]]
Requests that the values of the specified variables be returned by the server by sending a read variables request. If the association ID is omitted or is given as zero the variables are system variables, otherwise they are peer variables and the values returned will be those of the corresponding peer. Omitting the variable list will send a request with no data which should induce the server to return a default display.

`rv` [*assocID*] [*variable_name*[=*value*] [*,. . .*]]
An easy-to-type short form for the `readvar` command.

`writevar` *assocID variable_name=value* [*,. . .*]
Like the `readvar` request, except the specified variables are written instead of read.

`readlist` [*assocID*]
Requests that the values of the variables in the internal variable list be returned by the server. If the association ID is omitted or is 0 the variables are assumed to be system variables. Otherwise they are treated as peer variables. If the internal variable list is empty a request is sent without data, which should induce the remote server to return a default display.

`rl` [*assocID*]
An easy-to-type short form of the `readlist` command.

`writelist` [*assocID*]
Like the `readlist` request, except the internal list variables are written instead of read.

`mreadvar` *assocID assocID* [*variable_name*[*=value*] [*,. . .*]]
Like the `readvar` command except the query is done for each of a range of (nonzero) association IDs. This range is determined from the association list cached by the most recent `associations` command.

`mrv` *assocID assocID* [*variable_name*[*=value*] [*,. . .*]]
An easy-to-type short form of the `mreadvar` command.

`mreadlist` *assocID assocID*
Like the `readlist` command except the query is done for each of a range of (nonzero) association IDs. This range is determined from the association list cached by the most recent `associations` command.

`mrl` *assocID assocID*
An easy-to-type short form of the `mreadlist` command.

`clockvar` [*assocID*] [*variable_name*[*=value*] [*,. . .*]]
Requests that a list of the server's clock variables be sent. Servers which have a radio clock or other external synchronization respond positively to this. If the association identifier is omitted or zero the request is for the variables of the "system clock". This request generally gets a positive response from all servers with a clock. Some servers may treat clocks as pseudo-peers and, hence, can possibly have more than one clock connected at once. For these servers, referencing the appropriate peer association ID shows the variables of a particular clock. Omitting the variable list causes the server to return a default variable display.

`cv` [*assocID*] [*variable_name*[*=value*] [*,. . .*]]
An easy-to-type short form of the `clockvar` command.

`peers`
Obtains a list of in-spec peers of the server, along with a summary of each peer's state. Summary information includes:

- The address of the remote peer
- The reference ID (0.0.0.0 if the ref ID is unknown)

- The stratum of the remote peer
- The type of the peer (local, unicast, multicast or broadcast) when the last packet was received
- The polling interval in seconds
- The reachability register, in octal
- The current estimated delay offset and dispersion of the peer, all in milliseconds.

The character in the left margin indicates the fate of this peer in the clock selection process. The codes mean:

`SPACE`	Discarded due to high stratum and/or failed sanity checks.
`x`	Designated falsticker by the intersection algorithm.
`.`	Culled from the end of the candidate list.
–	Discarded by the clustering algorithm.
`+`	Included in the final selection set.
`#`	Selected for synchronization; but distance exceeds maximum.
`*`	Selected for synchronization.
`o`	Selected for synchronization, pps signal in use.

Since the `peers` command depends on the ability to parse the values in the responses it gets, it may fail to work from time to time with servers which poorly control the data formats.

The contents of the host field may be given in one of four forms. It may be a host name, an IP address, a reference clock implementation name with its parameter or, `REFCLK`(*implementation number*, *parameter*). On "hostnames no" only IP–addresses will be displayed.

`lpeers`
Like `peers`, except a summary of all associations for which the server is maintaining state is printed. This can produce a much longer list of peers from inadequate servers.

`opeers`
An old form of the `peers` command with the reference ID replaced by the local interface address.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWntpu

SEE ALSO

`attributes`(5)

BUGS

The `peers` command is non-atomic. It may occasionally result in spurious error messages about invalid associations occurring and terminating the command.

The timeout value is a fixed constant. As a result, it often waits a long time to timeout, since the fixed value assumes sort of a worst case. The program should improve the time out estimate as it sends queries to a particular host; but it does not.

NAME

ntptrace – trace a chain of NTP hosts back to their master time source

SYNOPSIS

/usr/sbin/ntptrace [-vdn] [-r *retries*] [-t *timeout*] [*server*]

DESCRIPTION

`ntptrace` determines where a given Network Time Protocol (NTP) server gets its time from, and follows the chain of NTP servers back to their master time source. If given no arguments, it starts with *localhost*.

OPTIONS

The following options are supported:

`-d`	Turns on some debugging output.
`-n`	Turns off the printing of host names; instead, host IP addresses are given. This may be necessary if a nameserver is down.
`-r` *retries*	Sets the number of retransmission attempts for each host.
`-t` *timeout*	Sets the retransmission timeout (in seconds); default = 2.
`-v`	Prints verbose information about the NTP servers.

EXAMPLES

EXAMPLE 1 Sample Output From the `ntptrace` Command

The following example shows the output from the `ntptrace` command:

```
% ntptrace
  localhost: stratum 4, offset 0.0019529, synch distance 0.144135
  server2.bozo.com: stratum 2, offset 0.0124263, synch distance 0.115784
  usndh.edu: stratum 1, offset 0.0019298, synch distance 0.011993, refid 'WWVB'
```

On each line, the fields are (left to right):

- The server's host name
- The server's stratum
- The time offset between that server and the local host (as measured by `ntptrace`; this is why it is not always zero for *localhost*)
- The host's synchronization distance
- The reference clock `ID` (only for stratum-1 servers)

All times are given in seconds. Synchronization distance is a measure of the goodness of the clock's time.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWntpu

SEE ALSO

`xntpd`(1M), `attributes`(5)

BUGS This program makes no attempt to improve accuracy by doing multiple samples.

NAME

obpsym – Kernel Symbolic Debugging for OpenBoot Firmware

SYNOPSIS

```
modload -p misc/obpsym
```

DESCRIPTION

`obpsym` is a kernel module that installs OpenBoot callback handlers that provide kernel symbol information to OpenBoot. OpenBoot firmware user interface commands use the callbacks to convert numeric *addresses* to kernel symbol names for display purposes, and to convert kernel symbol names to numeric *literals* allowing symbolic names to be used as input arguments to user interface commands.

Once `obpsym` is installed, kernel symbolic names may be used anywhere at the OpenBoot firmware's user interface command prompt in place of a literal (numeric) string. For example, if `obpsym` is installed, the OpenBoot firmware commands `ctrace` and `dis` typically display symbolic names and offsets in the form *modname:symbolname + offset*. User interface Commands such as `dis` can be given a kernel symbolic name such as `ufs:ufs_mount` instead of a numeric address.

Placing the command

```
forceload: misc/obpsym
```

into the `system`(4) file forces the kernel module `misc/obpsym` to be loaded and activates the kernel callbacks during the kernel startup sequence.

`obpsym` may be useful as a kernel debugger in situations where other kernel debuggers are not useful. For example, on SPARC machines, if `obpsym` is loaded, you may be able to use the OpenBoot firmware's `ctrace` command to display symbolic names in the stack backtrace after a watchdog reset.

Kernel Symbolic Name Syntax

The syntax for a kernel symbolic name is:

[*module-name* `:`] *symbol-name*

Where *module-name* is the name of the kernel module that the symbol *symbol-name* appears in. A `NULL` module name is taken as "all modules, in no particular order" by `obpsym`. The module name `unix` is equivalent to a `NULL` module name, so that conflicts with words defined in the firmware's vocabulary can be avoided.

Typically, OpenBoot firmware reads a word from the input stream and looks the word up in its internal *vocabulary* before checking if the word is a *literal*. Thus, kernel symbols, such as `reset` may be given as `unix:reset` to avoid the unexpected side effect of the firmware finding and executing a matching word in its vocabulary.

FILES

`/etc/system`
system configuration information file

/platform/*platform-name*/kernel/misc/obpsym

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcar

SEE ALSO

kadb(1M), kernel(1M), modload(1M), modunload(1M), uname(1), system(4), attributes(5)

OpenBoot 2.x Command Reference Manual

WARNINGS

Some OpenBoot firmware user interface commands may use system resources incompatibly with the way they are used by the Unix kernel. These commands and the use of this feature as a kernel debugger may cause interactions that the Unix kernel is not prepared to deal with. If this occurs, the Unix kernel and/or the OpenBoot firmware user interface commands may react unpredictably and may panic the system, or may hang or may cause other unpredictable results. For these reasons, the use of this feature is only minimally supported and recommended to be used only as a kernel debugger of "last resort".

If a breakpoint or watchpoint is triggered while the console frame buffer is powered off, the system can crash and be left in a state from which it is difficult to recover. If one of these is triggered while the monitor is powered off, you will not be able to see the debugger output.

NOTES

platform-name can be found using the -i option of uname(1)

obpsym is supported only on architectures that support OpenBoot firmware.

On some systems, OpenBoot must be completely RAM resident so the obpsym symbol callback support can be added to the firmware, if the firmware doesn't include support for the symbol callbacks. On these systems, obpsym may complain that it requires that "you must use ramforth to use this module".

See the *OpenBoot 2.x Command Reference Manual* for details on how to use the *ramforth* command, how to place the command into *nvramrc*, and how to set *use-nvramrc?* to true. On systems with version 1.x OpenBoot firmware, *nvramrc* doesn't exist, and the *ramforth* command must be typed manually after each reset, in order to use this module.

Once installed, the symbol table callbacks can be disabled by using the following OpenBoot firmware command:

```
0 0 set-symbol-lookup
```

NAME

ocfserv – OCF server

SYNOPSIS

```
ocfserv start

ocfserv stop
```

DESCRIPTION

The OCF server, `ocfserv`, is a per-host daemon that acts as the central point of communications with all smartcards connected to the host. Any application needing to use a smartcard communicates with the smartcard through this server, which is responsible for handling all traffic to the smartcards. All APIs exposed by this project are internally implemented to communicate with the OCF server. Applications communicate with the OCRF server using a socket-based protocol.

At startup time, the server reads the properties file to determine the terminals and cards currently registered.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

USAGE

Root priveleges are required to execute this utility.

FILES

`/etc/smartcard/opencard.properties`
file where server stores properties

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWocfr

SEE ALSO

`smartcard`(1M), `attributes`(5), `smartcard`(5)

NAME

parse_dynamic_clustertoc – parse clustertoc file based on dynamic entries

SYNOPSIS

cdrom`/export/exec/sparc.Solaris_2.x/sbin/install.d/parse_dynamic_clustertoc`

cdrom`/export/exec/i386.Solaris_2.x/sbin/install.d/parse_dynamic_clustertoc`

DESCRIPTION

This script parses the `clustertoc` file before the `suninstall`(1M) process is run. `parse_dynamic_clustertoc` is called by a modified sysconfig script on the install CD. When `parse_dynamic_clustertoc` runs, it reads the clustertoc and when it encounters `SUNW_CSRMBRIFF` lines, it either checks the platform using the script's builtin function, or calls an external script. The script exits with a 0 if the cluster entry is included, otherwise it will be ignored. If the cluster entry is to be included, the `SUNW_CSRMBRIFF` =(*test test_arg*) *cluster* line is converted to `SUNW_CSRMEMBER` =*cluster*.

EXAMPLES

EXAMPLE 1 Checking For an SX Framebuffer

The following is an example of a simple external test to check for an SX Framebuffer. The entry in the `clustertoc` file is shown and following that is the script that must be placed in the `install.d/dynamic_test` directory.

```
SUNW_CSRMBRIFF=(smcc.dctoc sx)SUNWCsx
#! /bin/sh
#
# Likewise, this file is expected to live under $(TESTDIR).
#
case "$1"
in
        sx)     prtconf -p  | grep 'SUNW,sx' 1> /dev/null;;
esac
```

FILES

cdrom`/Solaris_2.x/locale/C/.clustertoc.dynamic`
Dynamic version of the `clustertoc` file

cdrom`/export/exec/sparc.Solaris_2.x/sbin/install.d/dynamic_test`
Directory that contains any additional tests

cdrom`/export/exec/i386.Solaris_2.x/sbin/install.d/dynamic_test`
Directory that contains any additional tests

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SHWPcdrom (Solaris CD)

SEE ALSO

`suninstall`(1M), `clustertoc`(4), `attributes`(5)

NAME

passmgmt – password files management

SYNOPSIS

passmgmt -a *options name*

passmgmt -m *options name*

passmgmt -d *name*

DESCRIPTION

The passmgmt command updates information in the password files. This command works with both /etc/passwd and /etc/shadow.

passmgmt -a adds an entry for user *name* to the password files. This command does not create any directory for the new user and the new login remains locked (with the string *LK* in the password field) until the passwd(1) command is executed to set the password.

passmgmt -m modifies the entry for user *name* in the password files. The name field in the /etc/shadow entry and all the fields (except the password field) in the /etc/passwd entry can be modified by this command. Only fields entered on the command line will be modified.

passmgmt -d deletes the entry for user *name* from the password files. It will not remove any files that the user owns on the system; they must be removed manually.

passmgmt can be used only by the super-user.

OPTIONS

-c *comment*	A short description of the login, enclosed in quotes. It is limited to a maximum of 128 characters and defaults to an empty field.
-h *homedir*	Home directory of *name*. It is limited to a maximum of 256 characters and defaults to /usr/*name*.
-u *uid*	UID of the *name*. This number must range from 0 to the maximum non-negative value for the system. It defaults to the next available UID greater than 99. Without the -o option, it enforces the uniqueness of a UID.
-o	This option allows a UID to be non-unique. It is used only with the -u option.
-g *gid*	GID of *name*. This number must range from 0 to the maximum non-negative value for the system. The default is 1.
-s *shell*	Login shell for *name*. It should be the full pathname of the program that will be executed when the user logs in. The maximum size of *shell* is 256 characters. The default is for this field to be empty and to be interpreted as /usr/bin/sh.
-l logname	This option changes the *name* to logname. It is used only with the -m option. The total size of each login entry is limited to a maximum of 511 bytes in each of the password files.

FILES

/etc/passwd
/etc/shadow

/etc/opasswd
/etc/oshadow

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

passwd(1), useradd(1M), userdel(1M), usermod(1M), passwd(4), shadow(4), attributes(5)

EXIT STATUS

The passmgmt command exits with one of the following values:

0 Success.

1 Permission denied.

2 Invalid command syntax. Usage message of the passmgmt command is displayed.

3 Invalid argument provided to option.

4 UID in use.

5 Inconsistent password files (for example, *name* is in the /etc/passwd file and not in the /etc/shadow file, or vice versa).

6 Unexpected failure. Password files unchanged.

7 Unexpected failure. Password file(s) missing.

8 Password file(s) busy. Try again later.

9 *name* does not exist (if -m or -d is specified), already exists (if -a is specified), or logname already exists (if -m -l is specified).

NOTES

Do not use a colon (:) or RETURN as part of an argument. It is interpreted as a field separator in the password file. The passmgmt command will be removed in a future release. Its functionality has been replaced and enhanced by useradd, userdel, and usermod. These commands are currently available.

This command only modifies password definitions in the local /etc/passwd and /etc/shadow files. If a network nameservice such as NIS or NIS+ is being used to supplement the local files with additional entries, passmgmt cannot change information supplied by the network nameservice.

NAME

patchadd – apply a patch package to a system running the Solaris operating environment

SYNOPSIS

patchadd [-d] [-u] [-B *backout_dir*] [-C *net_install_image* | -R *client_root_path* | -S *service*] *patch*

patchadd [-d] [-u] [-B *backout_dir*] [-C *net_install_image* | -R *client_root_path* | -S *service*] -M *patch_dir* | *patch_id...* | *patch_dirpatch_list*

patchadd [-C *net_install_image* | -R *client_root_path* | -S *service*] -p

DESCRIPTION

patchadd applies a patch package to a system running the Solaris 2.*x* operating environment or later Solaris environments (such as Solaris 8) that are compatible with Solaris 2.*x*. This patch installation utility cannot be used to apply Solaris 1 patches. patchadd must be run as root.

There are three forms of the patchadd command.

The first form of patchadd installs *one* patch to a system, client, service, or the mini root of a Net Install Image.

The second form of patchadd installs *more than one* patch to a system, client, service, or the mini root of a Net Install Image.

The third form of patchadd *displays* installed patches on the client, service, or the mini root of a Net Install Image.

OPTIONS

The following options are supported:

-B *backout_dir*
Saves backout data to a directory other than the package database. Specify *backout_dir* as an absolute path name.

-C *net_install_image*
Patches the files located on the mini root on a Net Install Image created by setup_install_server. Specify *net_install_image* as the absolute path name to a Solaris 2.6 or compatible version boot directory. See EXAMPLES.

-d
Does not back up the files to be patched. *The patch cannot be removed.*

-M *patch_dir patch_id ...* | *patch_dir patch_list*
Specifies the patches to be installed. Specify patches to the -M option in one of the following ways:

1. By directory location *and* patch number.

   To use the directory location and patch number, specify *patch_dir* as the absolute path name of the directory that contains spooled patches. Specify *patch_id* as the patch number of a given patch. Specifying multiple *patch_id*'s is recommended.

2. By directory location *and* the name of a file containing a patch list.

To use the directory location and a file containing a patch list, specify *patch_dir* as the absolute path name of the directory containing the file with a list of patches to be installed. Specify *patch_list* as the name of the file containing the patches to be installed.

`-p`
Displays a list of the patches currently applied.

`-R` *client_root_path*
Locates all patch files generated by `patchadd` under the directory *client_root_path*. *client_root_path* is the directory that contains the bootable root of a client from the server's perspective. Specify *client_root_path* as the absolute path name to the beginning of the directory tree under which all patch files generated by `patchadd` are to be located. `-R` cannot be specified with the `-S` option. See `NOTES`.

`-S` *service*
Specifies an alternate service (for example, Solaris_2.3). This service is part of the server and client model, and can only be used from the server's console. Servers can contain shared `/usr` file systems that are created by Host Manager. These service areas can then be made available to the clients they serve. `-S` cannot be specified with the `-R` option. See `NOTES`.

`-u`
Turns off file validation. Applies the patch even if some of the files to be patched have been modified since their original installation.

OPERANDS

The following operands are supported:

patch	The absolute path name to *patch_id*. `/var/sadm/spool/patch/104945-02` is an example of a `patch`.
patch_dir	The absolute path name to the directory that contains all the spooled patches. `/var/sadm/spool/patch` is an example of a `patch_dir`.
patch_id	The patch number of a given patch. `104945-02` is an example of a *patch_id*.
patch_list	The name of a file that contains a list of patches to install. *patch_list* files contain one *patch_id* on each line.

EXAMPLES

The examples in this section are all relative to the `/usr/sbin` directory.

EXAMPLE 1 Installing a Patch to a Standalone Machine

The following example installs a patch to a standalone machine:

```
example# patchadd /var/spool/patch/104945-02
```

EXAMPLE 1 Installing a Patch to a Standalone Machine *(Continued)*

EXAMPLE 2 Installing a Patch to a Client From the Server's Console

The following example installs a patch to a client from the server's console:

```
example# patchadd -R /export/root/client1  /var/spool/patch/104945-02
```

EXAMPLE 3 Installing a Patch to a Service From the Server's Console

The following example installs a patch to a service from the server's console:

```
example# patchadd -S Solaris_2.3 /var/spool/patch/104945-02
```

EXAMPLE 4 Installing Multiple Patches in a Single Invocation of `patchadd`

The following example installs multiple patches in a single `patchadd` invocation:

```
example# patchadd -M /var/spool/patch 104945-02  104946-02 102345-02
```

EXAMPLE 5 Installing Multiple Patches Specifying a File with the List of Patches to Install

The following example installs multiple patches specifying a file with the list of patches to install:

```
example# patchadd -M /var/spool/patch patchlist
```

EXAMPLE 6 Installing Multiple Patches to a Client and Saving the Backout Data to a Directory Other Than the Default

The following example installs multiple patches to a client and saves the backout data to a directory other than the default:

```
example# patchadd -M /var/spool/patch -R /export/root/client1
     -B /export/backoutrepository 104945-02 104946-02 102345-02
```

EXAMPLE 7 Installing a Patch to a Solaris 2.6 or Compatible Version Net Install Image

The following example installs a patch to a Solaris 2.6 or compatible version Net Install Image:

```
example# patchadd -C /export/Solaris_2.6/Tools/Boot
    /var/spool/patch/104945-02
```

EXAMPLE 8 Displaying the Patches Installed on a Client

The following example displays the patches installed on a client:

```
example# patchadd -R /export/root/client1 -p
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWswmt, SUNWcsu

DIAGNOSTICS

The following messages may help in determining some of the most common problems associated with installing a patch.

Patch Installation errors

Message

```
The prepatch script exited with return code retcode.
patchadd is terminating.
```

Explanation and Recommended Action

The prepatch script supplied with the patch exited with a return code other than `0`. Run a script trace of the prepatch script and find out why the prepatch had a bad return code. Add the `-x` option to the first line of the prepatch script to fix the problem and run `patchadd` again.

Message

```
The postpatch script exited with return code retcode.
Backing out patch.
```

Explanation and Recommended Action

The postpatch script provided with the patch exited with an error code other than `0`. This script is mostly used to cleanup files (that is, when a package is known to have ownership or permission problems) attributes that don't correspond to the patch package's objects. After the user has noted all validation errors and taken the appropriate action for each one, the user should re-run `patchadd` using the `-u` (unconditional) option. This time, the patch installation will ignore validation errors and install the patch anyway.

Message

```
Insufficient space in /var/sadm/patch to save old files.
(For 2.4 systems and previous)
```

Explanation and Recommended Action

There is insufficient space in the `/var/sadm/patch` directory to save old files. The user has three options for handling this problem: Use the `-B` option while invoking `patchadd`. This option will direct `patchadd` to: save the backout data to the user specified file system, generate additional disk space by deleting unneeded files, or override the saving of the old files by using the `-d` (do not save) option when running `patchadd`.

If the user elects not to save the old versions of the files to be patched, `patchrm` *cannot* be used. One way to regain space on a system is to remove the save area for previously applied patches. Once the user has decided that it is unlikely that a patch will be backed out, the user can remove the files that were saved by `patchadd`. The following commands should be executed to remove the saved files for `patch`*patch_id*:

`cd /var/sadm/patch/`*patch_id*
`rm -r save/*`
`rm .oldfilessaved`

After these commands have been executed, `patch` *patch_id*can no longer be backed out.

Message

`Insufficient space in /var/sadm/pkg/`*PKG*`/save to save old files.`
`(For 2.5 systems and later)`

Explanation and Recommended Action

There is insufficient space in the `/var/sadm/pkg/PKG/save` directory to save old files. The user has three options for handling this problem: (1) Use the `-B` option while invoking `patchadd`. This option will direct `patchadd` to save the backout data to the user specified file system. (See above synopsis) (2) Generate additional disk space by deleting unneeded files, or (3) override the saving of the old files by using the `-d` (do not save) option when running `patchadd`. However if the user elects not to save the old versions of the files to be patched, `patchrm` *cannot* be used. One way to regain space on a system is to remove the save area for previously applied patches. Once the user has decided that it is unlikely that a patch will be backed out, the user can remove the files that were saved by `patchadd`. The following commands should be executed to remove the saved files for patch *patch_id*:

`cd /var/sadm/pkg/`*pkgabbrev*`/save`**
`rm -r` *patch_id*

After these commands have been executed, `patch` *patch_id* can no longer be backed out.

Message

`Save of old files failed.`
`(For 2.4 systems and previous)`

Explanation and Recommended Action

Before applying the patch, the patch installation script uses `cpio` to save the old versions of the files to be patched. This error message means that the `cpio` failed. The output of the `cpio` would have been preceded this message. The user should take the appropriate action to correct the `cpio` failure. A common reason for failure will be insufficient disk space to save the old versions of the files. The user has two options for handling insufficient disk space: (1) generate additional disk space by deleting unneeded files, or (2) override the saving of the old files by using the `-d` option when running `patchadd`. However if the user elects not to save the old versions of the files to be patched, the patch *cannot* be backed out.

Message

```
Pkgadd of pkgname package failed with error code code.
See /tmp/log.patch_id for reason for failure.
```

Explanation and Recommended Action

The installation of one of the patch packages failed. `patchadd` will backout the patch to leave the system in its pre-patched state. See the log file for the reason for failure. Correct the problem and reapply the patch.

Message

```
Pkgadd of pkgname package failed with error code code.
Will not backout patch...patch re-installation.
Warning: The system may be in an unstable state!
See /tmp/log.patch_id for reason for failure.
```

Explanation and Recommended Action

The installation of one of the patch packages failed. `patchadd` will *not* backout the patch. You may manually backout the patch using `patchrm`, then re-apply the entire patch. Look in the log file for the reason `pkgadd` failed. Correct the problem and re-apply the patch.

Message

```
patchadd is unable to find the INST_RELEASE file. This file
        must be present for patchadd to function correctly.
```

Explanation and Recommended Action

The `INST_RELEASE` file is missing from the system. This file is created during either initial installation or during an update.

Message

```
A previous installation of patch patch_id was invoked
that saved files that were to be patched. Since files
were saved, you must run this instance of patchadd
without the -d option.
```

Explanation and Recommended Action

If a patch was previously installed without using the `-d` option, then the re-installation attempt must also be invoked without the `-d` option. Execute `patchadd` without the `-d` option.

Message

```
A previous installation of patch patch_id was invoked
with the -d option. (i.e. Do not save files that would
be patched) Therefore, this invocation of patchadd
must also be run with the -d option.
```

Explanation and Recommended Action

If a patch was previously installed using the `-d` option, then the re-installation attempt must also be invoked with the`-d` option. Execute `patchadd` with the `-d'` option.

Diagnostic Reference

The patch installation messages listed below are not necessarily considered errors as indicated in the explanations given. These messages are, however, recorded in the patch installation log for diagnostic reference.

Message

```
Package not patched:
PKG=SUNxxxx
Original package not installed
```

Explanation and Recommended Action

One of the components of the patch would have patched a package that is not installed on your system. This is not necessarily an error. A patch may fix a related bug for several packages.

For example, suppose a patch fixes a bug in both the online-backup and `fddi` packages. If you had online-backup installed but didn't have `fddi` installed, you would get the message :

```
Package not patched:
PKG=SUNWbf
Original package not installed
```

This message only indicates an error if you thought the package was installed on your system. If this is the case, take the necessary action to install the package, backout the patch (if it installed other packages) and re-install the patch.

Message

```
Package not patched:
PKG=SUNxxx
ARCH=xxxxxxx
VERSION=xxxxxxx
Architecture mismatch
```

Explanation and Recommended Action

One of the components of the patch would have patched a package for an architecture different from your system. This is not necessarily an error. Any patch to one of the architecture-specific packages might contain one element for each of the possible architectures. For example, assume you are running on a `sun4m`. If you were to install a patch to package `SUNWcar`, you would see the following (or similar) messages:

```
Package not patched:
PKG=SUNWcar
ARCH=sparc.sun4c
VERSION=11.5.0,REV=2.0.18
Architecture mismatch

Package not patched:
PKG=SUNWcar
ARCH=sparc.sun4u
VERSION=11.5.0,REV=2.0.18
Architecture mismatch

Package not patched:
```

```
PKG=SUNWcar
ARCH=sparc.sun4e
VERSION=11.5.0,REV=2.0.18
Architecture mismatch

Package not patched:
PKG=SUNWcar
ARCH=sparc.sun4
VERSION=11.5.0,REV=2.0.18
Architecture mismatch
```

These messages indicate an error condition only if `patchadd` does not correctly recognize your architecture.

Message

```
Package not patched:
PKG=SUNxxxx
ARCH=xxxx
VERSION=xxxxxxx
Version mismatch
```

Explanation and Recommended Action

The version of software to which the patch is applied is not installed on your system. For example, if you were running Solaris 5.5, and you tried to install a patch against Solaris 5.6, you would see the following (or similar) message:

```
Package not patched:
PKG=SUNWcsu
ARCH=sparc
VERSION=10.0.2
Version mismatch
```

This message does not necessarily indicate an error. If the version mismatch was for a package you needed patched, either get the correct patch version or install the correct package version. Then backout the patch (if necessary) and re-apply.

Message

```
Re-installing Patch.
```

Explanation and Recommended Action

The patch has already been applied, but there is at least one package in the patch that could be added. For example, if you applied a patch that had both Openwindows and Answerbook components, but your system did not have Answerbook installed, the Answerbook parts of the patch would not have been applied. If, at a later time, you `pkgadd` Answerbook, you could re-apply the patch, and the Answerbook components of the patch would be applied to the system.

Message

```
patchadd Interrupted.
patchadd is terminating.
```

Explanation and Recommended Action

`patchadd` was interrupted during execution (usually through pressing CTRL-c). `patchadd` will clean up its working files and exit.

Message

```
patchadd Interrupted.
Backing out Patch...
```

Explanation and Recommended Action

`patchadd` was interrupted during execution (usually through pressing CTRL-c). `patchadd` will clean up its working files, backout the patch, and exit.

SEE ALSO

`cpio`(1), `pkginfo`(1), `patchrm`(1M), `pkgadd`(1M), `pkgchk`(1M), `pkgrm`(1M), `smpatch`(1M), `showrev`(1M), `attributes`(5)

NOTES

To successfully install a patch to a client or server, `patchadd` must be issued twice, once with the `-R` option and once with the `-S` option. This guarantees that the patch is installed to both the `/usr` and `root` partitions. This is necessary if there are both `/usr` and `root` packages in the patch.

`pkgadd` is invoked by `patchadd` and executes the installation scripts in the *pkg*`/install` directory. The `checkinstall` script is executed with its ownership set to user `install`, if there is no user `install` then `pkgadd` executes the `checkinstall` script as nobody. The SVR4 ABI states that the `checkinstall` shall only be used as an information gathering script. If the permissions for the `checkinstall` script are changed to something other than the initial settings, `pkgadd` may not be able to open the file for reading, thus causing the patch installation to abort with the following error:

```
pkgadd: ERROR: checkinstall script did not complete successfully.
```

The permission for the `checkinstall` script should not be changed. Contents of log file for a successfull installation: `patchadd` redirects `pkgadd`'s output to the patch installation log file. For a successfull installation, `pkgadd` will produce the following message that gets inserted into the log file:

```
This appears to be an attempt to install the same architecture
and version of a package which is already installed. This
installation will attempt to overwrite this package.
```

This message does not indicate a failure, it represents the correct behavior by `pkgadd` when a patch installs correctly.

On client server machines the patch package is *not* applied to existing clients or to the client root template space. Therefore, when appropriate, *all client machines will need the patch applied directly using this same patchadd method on the client.* See instructions above for applying patches to a client. A bug affecting a package utility (for example, `pkgadd`, `pkgrm`, `pkgchk`) could affect the reliability of `patchadd` or `patchrm` which use package utilities to install and backout the patch package. It is recommended that any patch that fixes package utility problems be reviewed and, if necessary, applied before other patches are applied. Existing patches are:

Solaris 2.1:
patch 100901

Solaris 2.2:
101122

Solaris 2.3:
10133

Solaris 2.4 Sparc Platform Edition:
102039

Solaris 2.4 Intel Platform Edition:
102041

Solaris 2.5.1 Sparc Platform Edition:
104578

Solaris 2.51 Intel Platform Edition:
104579

Solaris 2.6 Sparc Platform Edition:
106292

Solaris 2.6 Intel Platform Edition:
106293

NAME

patchrm – remove a Solaris patch package and restore previously saved files

SYNOPSIS

patchrm [-f] [-B *backout_dir*] [-C *net_install_image* | -R *client_root_path* | -S *service*] *patch_id*

DESCRIPTION

patchrm removes a patch package and restores previously saved files to a system running the Solaris 2.x operating environment or later Solaris environments (such as Solaris 8) that are compatible with Solaris 2.x. patchrm cannot be used with Solaris 1 patches. patchrm must be run as root.

OPTIONS

The following options are supported:

-B *backout_dir* — Removes a patch whose backout data has been saved to a directory other than the package database. This option is only needed if the original backout directory, supplied to the patchadd command at installation time, has been moved. Specify *backout_dir* as an absolute path name.

-C *net_install_image* — Removes the patched files located on the mini root on a Net Install Image created by setup_install_server. Specify *net_install_image* as the absolute path name to a Solaris 2.6 or compatible version boot directory. See EXAMPLES.

-f — Forces the patch removal regardless of whether the patch was superseded by another patch.

-R *client_root_path* — Locates all patch files generated by patchrm under the directory *client_root_path*. *client_root_path* is the directory that contains the bootable root of a client from the server's perspective. Specify *client_root_path* as the absolute path name to the beginning of the directory tree under which all patch files generated from patchrm will be located. -R cannot be specified with the -S option.

-S *service* — Specifies an alternate service (for example, Solaris_2.3). This service is part of the server and client model, and can only be used from the server's console. Servers can contain shared /usr file systems that are created by Host Manager. These service areas can then be made available to the clients they serve. -S cannot be specified with the -R option..

OPERANDS

The following operands are supported:

patch_id — The patch number of a given patch. 104945-02 is an example of a *patch_id*.

EXAMPLES

The examples in this section assume that patch `104945-02` has been installed to the system prior to removal. All of the examples are relative to the `/usr/sbin` directory.

EXAMPLE 1 Removing a Patch From a Stand-alone System

The following example removes a patch from a standalone system:

```
example# patchrm 104945-02
```

EXAMPLE 2 Removing a Patch From a Client's System From the Server's Console

The following example removes a patch from a client's system from the server's console:

```
example# patchrm -R /export/root/client1 104945-02
```

EXAMPLE 3 Removing a Patch From a Server's Service Area

The following example removes a patch from a server's service area:

```
example# patchrm -S Solaris_2.3 104945-02
```

EXAMPLE 4 Removing a Patch From a Net Install Image

The following example removes a patch from a Net Install Image:

```
example# patchrm -C /export/Solaris_2.6/Tools/Boot 104945-02
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWswmt, SUNWcsu

DIAGNOSTICS

The following messages may help in determining some of the most common problems associated with backing out a patch.

Message

```
prebackout patch exited with return code code.
patchrm exiting.
```

Explanation and Recommended Action

The prebackout script supplied with the patch exited with a return code other than `0`. Generate a script trace of the prebackout script to determine why the

prebackout script failed. Add the `-x` option to the first line of the prepatch script to fix the problem and run `patchadd` again.

Message

`postbackout patch exited with return code` *code*`.`
`patchrm exiting.`

Explanation and Recommended Action

The postbackout script supplied with the patch exited with a return code other than `0`. Look at the postbackout script to determine why it failed. Add the `-x` option to the first line of the prepatch script to fix the problem, and, if necessary, *re-exececute the postbackout script only*.

Message

`Only one service may be defined.`

Explanation and Recommended Action

You have attempted to specify more than one service from which to backout a patch. Different services must have their patches backed out with different invocations of `patchrm`.

Message

`The -S and -R arguments are mutually exclusive.`

Explanation and Recommended Action

You have specified both a non-native service and a *client_root_path* from which to backout a patch. These two arguments are mutually exclusive. If backing out a patch from a non-native `usr` partition, the `-S` option should be used. If backing out a patch from a client's root partition (either native or non-native), the `-R` option should be used.

Message

`The` *service* `service cannot be found on this system`

Explanation and Recommended Action

You have specified a non-native service from which to backout a patch, but the specified service is not installed on your system. Correctly specify the service when backing out the patch.

Message

`Only one` *client_root_path* `may be defined.`

Explanation and Recommended Action

You have specified more than one *client_root_path* using the `-R` option. The `-R` option may be used only once per invocation of `patchrm`.

Message

`The` *dir* `directory cannot be found on this system.`

Explanation and Recommended Action
You have specified a directory using the `-R` option which is either not mounted, or does not exist on your system. Verify the directory name and re-backout the patch.

Message

```
Patch patch_id has not been successfully installed to this system.
```

Explanation and Recommended Action
You have attempted to backout a patch that is not installed on this system. If you must restore previous versions of patched files, you may have to restore the original files from the initial installation CD.

Message

```
Patch patch_id has not been successfully applied to this system.
Will remove directory dir.
```

Explanation and Recommended Action
You have attempted to back out a patch that is not applied to this system. While the patch has not been applied, a residual `/var/sadm/patch/`*patch_id* (perhaps from an unsuccessful `patchadd`) directory still exists. The patch cannot be backed out. If you must restore old versions of the patched files, you may have to restore them from the initial installation CD.

Message

```
This patch was obsoleted by patch patch_id.
Patches must be backed out in the reverse order in
which they were installed. Patch backout aborted.
```

Explanation and Recommended Action
You are attempting to backout patches out of order. Patches should never be backed-out out of sequence. This could undermine the integrity of the more current patch.

Message

```
Patch patch_id is required to be installed by an already
installed patch_id.
It cannot be backed out until the required patch is backed out first.
```

Explanation and Recommended Action
Backout the patch that is required to be installed then backout the desired patch.

Message

```
The installation of patch patch_id was interrupted.
```

Explanation and Recommended Action
A previous installation was interrupted. The interrupted patch needs to be installed before backing out the desired patch.

Message

```
Patch patch_id was installed without backing up the original
files. It cannot be backed out.
```

Explanation and Recommended Action

Either the `-d` option of `patchadd` was set when the patch was applied, or the save area of the patch was deleted to regain space. As a result, the original files are not saved and `patchrm` cannot be used. The original files can only be recovered from the original installation CD.

Message

```
pkgadd of pkgname package failed return code code.
See /var/sadm/patch/patch_id/log for reason for failure.
```

Explanation and Recommended Action

The installation of one of patch packages failed. See the log file for the reason for failure. Correct the problem and run the backout script again.

Message

```
Restore of old files failed.
```

Explanation and Recommended Action

The backout script uses the `cpio` command to restore the previous versions of the files that were patched. The output of the cpio command should have preceded this message. The user should take the appropriate action to correct the `cpio` failure. This is for Solaris 2.4 or previous versions.

SEE ALSO

`cpio`(1), `pkginfo`(1), `patchadd`(1M), `pkgadd`(1M), `pkgchk`(1M), `pkgrm`(1M), `showrev`(1M), `attributes`(5)

NOTES

On client server machines the patch package is *not* removed from existing clients or from client root template space. Therefore, when appropriate, *all client machines will need the patch removed directly using this same patchrm method on the client.* A bug affecting a package utility (for example, `pkgadd`, `pkgrm`, `pkgchk`) could affect the reliability of `patchadd` or `patchrm` which use package utilities to install and backout the patch package. It is recommended that any patch that fixes package utility problems be reviewed and, if necessary, applied before other patches are applied. Existing patches are:

Solaris 2.1:
patch 100901

Solaris 2.2:
101122

Solaris 2.3:
10133

Solaris 2.4 Sparc Platform Edition:
102039

Solaris 2.4 Intel Platform Edition:
102041

Solaris 2.5.1 Sparc Platform Edition:
104578

Solaris 2.51 Intel Platform Edition:
104579

Solaris 2.6 Sparc Platform Edition:
106292

Solaris 2.6 Intel Platform Edition:
106293

NAME

pbind – control and query bindings of processes to processors

SYNOPSIS

pbind -b *processor_id pid...*

pbind -u *pid...*

pbind [-q] [*pid...*]

DESCRIPTION

pbind controls and queries bindings of processes to processors. `pbind` binds all the LWPs (lightweight processes) of a process to a processor, or removes or displays the bindings.

When an LWP is bound to a processor, it will be executed only by that processor except when the LWP requires a resource that is provided only by another processor. The binding is not exclusive, that is, the processor is free execute other LWPs as well.

Bindings are inherited, so new LWPs and processes created by a bound LWP will have the same binding. Binding an interactive shell to a processor, for example, binds all commands executed by the shell.

Superusers may bind or unbind any process, and other users can use `pbind` to bind or unbind any process for which the user has permission to signal, that is, any process that has the same effective user ID as the user.

OPTIONS

The following options are supported:

`-b` *processor_id* — Binds all the LWPs of the specified processes to the processor *processor_id*. Specify *processor_id* as the processor ID of the processor to be controlled or queried. *processor_id* must be present and on-line. Use the `psrinfo` command to determine whether or not *processor_id* is present and on-line. See `psrinfo`(1M).

`-q` — Displays the bindings of the specified processes, or of all processes. If a process is composed of multiple LWPs, which have different bindings, the bindings of only one of the bound LWPs will be displayed.

`-u` — Removes the bindings of all LWPs of the specified processes, allowing them to be executed on any on-line processor.

OPERANDS

The following operands are supported:

pid — The process ID of the process to be controlled or queried.

EXAMPLES

EXAMPLE 1

Binding processes

The following example binds processes 204 and 223 to processor 2.

```
example% pbind -b 2 204 223
```

This command displays the following output:

```
process id 204: was 2, now 2
process id 223: was 3, now 2
```

Unbinding a process

The following example unbinds process 204.

```
example% pbind -u 204
```

Querying Bindings

The following example demonstrates that process 1 is bound to processor 0, process 149 has at least one LWP bound to CPU3, and process 101 has no bound LWPs.

```
example% pbind -q 1 149 101
```

This command displays the following output:

```
process id 1: 0
process id 149: 3
process id 101: not bound
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

SEE ALSO

`psradm`(1M), `psrinfo`(1M), `psrset`(1M), `processor_bind`(2), `processor_info`(2), `sysconf`(3C), `attributes`(5)

DIAGNOSTICS

`pbind: cannot query pid 31: No such process`
The process specified did not exist or has exited.

`pbind: cannot bind pid 31: Not owner`
The user does not have permission to bind the process.

`pbind: cannot bind pid 31: Invalid argument`
The specified processor is not on-line.

NAME

pcmciad – PCMCIA user daemon

SYNOPSIS

/usr/lib/pcmciad

DESCRIPTION

The PCMCIA user daemon provides user-level services for the PCMCIA nexus driver and PCMCIA card client drivers. There are no user-configurable options for this daemon.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpcmcu

SEE ALSO

pcmcia(4), attributes(5)

DIAGNOSTICS

pcmciad: can't open /dev/pem: No such file or directory
The user daemon could not communicate with the PCMCIA event management driver.

NAME

pfinstall – tests installation profiles

SYNOPSIS

/usr/sbin/install.d/pfinstall -D | -d *disk_config* [-c *CDpath*] *profile*

DESCRIPTION

After you create a profile, you can use the `pfinstall` command to test the profile and see if it does what you want before using it to install or upgrade a system. `pfinstall` enables you to test a profile against:

- The system's disk configuration where `pfinstall` is being run.
- Other disks by using a disk configuration file that represents a structure of a disk. See `NOTES` on how to create a disk configuration file.

To successfully and accurately test a profile for a particular Solaris release, you must test a profile within the Solaris environment of the same release. For example, if you want to test a profile for Solaris 2.6, you have to run the `pfinstall` command on a system running Solaris 2.6.

So, on a system running Solaris 2.6, you can test Solaris 2.6 initial installation profiles. However, if you want to test a Solaris 2.6 upgrade profile on a system running a previous version of Solaris, or if you don't have a Solaris 2.6 system installed yet to test Solaris 2.6 initial installation profiles, you have to boot a system from a Solaris 2.6 CD image and temporarily create a Solaris 2.6 install environment. Then, you can run `pfinstall` in the Solaris 2.6 install environment to test your profiles.

To create a temporary Solaris 2.6 install environment, boot a system from a Solaris 2.6 CD image (just as you would to install), answer any system identification questions, choose the Solaris Interactive Installation program, and exit out of the first screen that is presented. Then, from the shell, you can execute the `pfinstall` command.

OPTIONS

The following options are supported:

`-D` — `pfinstall` uses the system's disk configuration to test the profile. You must specify either this option or the `-d` option to test the profile (see `WARNINGS`).

`-d` *disk_config* — `pfinstall` uses a disk configuration file, *disk_config*, to test the profile. See `NOTES` on how to create a disk configuration file. You must specify either this option or the `-D` option to test the profile (see `WARNINGS`). This option cannot be used with an upgrade profile (`install_type upgrade`). You must always test an upgrade profile against a system's disk configuration (`-D` option).

`-c` *CDpath* — The path to the Solaris 2 installation image. This is required if the image is not mounted on `/cdrom`. (For example, use this option if you copied the installation image to disk or mounted the CD-ROM on a directory other than `/cdrom`.)

OPERANDS

The following operand is supported:

profile — The file name of the profile to test. If *profile* is not in the directory where `pfinstall` is being run, you must specify the path.

EXAMPLES

EXAMPLE 1 Testing an Upgrade Profile

The following example tests an upgrade profile, `upgrade.prof`, on a system with a previous version of the Solaris software installed.

1. Boot the system to be upgraded from a Solaris 2.6 image (just as you would to install). The image can be located in the system's local CD-ROM or on an install server.
2. Answer the system configuration questions, if prompted.
3. If you are presented with a choice of installation options, choose the Solaris Interactive Installation program.
4. Exit from the first screen of the Solaris Interactive Installation program.

   After the Solaris Interactive Installation program exits, a shell prompt is displayed.
5. Create a temporary mount point:

   ```
   example# mkdir /tmp/mnt
   ```
6. Mount the directory that contains the profile(s) you want to test.

   If you want to mount a remote NFS file system (for systems on the network), enter:

   ```
   mount -F nfs server_name:path /tmp/mnt
   ```
   If you want to mount a UFS-formatted diskette, enter:

   ```
   mount -F ufs /dev/diskette /tmp/mnt
   ```
   If you want to mount a PCFS-formatted diskette, enter:

   ```
   mount -F pcfs /dev/diskette /tmp/mnt
   ```
7. Change directory to `/tmp/mnt` where the profile resides:

   ```
   example# cd /tmp/mnt
   ```
8. Test the `upgrade.prof` profile:

   ```
   /usr/sbin/install.d/pfinstall -D upgrade.prof
   ```

EXAMPLE 2 Testing the `basic.prof` Profile

The following example tests the `basic.prof` profile against the disk configuration on a Solaris 2.6 system where `pfinstall` is being run. The path to the Solaris CD image is specified because Volume Management is being used.

```
example# /usr/sbin/install.d/pfinstall -D -c /cdrom/cdrom0/s0 basic.prof
```

EXAMPLE 3 Testing the `basic.prof` Profile

The following example tests the `basic.prof` profile against the `535_test` disk configuration file. This example uses a Solaris CD image located in the `/export/install` directory, and `pfinstall` is being run on a Solaris 2.6 system.

```
example# /usr/sbin/install.d/pfinstall -d 535_test \
     -c /export/install basic.prof
```

EXIT STATUS

`0` Successful (system rebooted).

`1` Successful (system not rebooted).

`2` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWinst

SEE ALSO

`fdisk`(1M), `prtvtoc`(1M), `attributes`(5)

Solaris 9 Installation Guide

WARNINGS

If the `-d` or `-D` option is not specified, `pfinstall` may perform an actual installation on the system by using the specified profile, and the data on the system may be `overwritten`.

NOTES

You have to test a profile on a system with the same platform type for which the profile was created.

SPARC

To create a disk configuration file (`-d` option) for a SPARC based system:

1. Locate a SPARC based system with a disk that you want to test.
2. Create a disk configuration file by redirecting the output of the `prtvtoc`(1M) command to a file.

```
example# prtvtoc /dev/rdsk/c0t3d0s2 > 535_disk
```

3. (Optional.) Concatenate disk configuration files into a single file to test a profile against multiple disks. The target numbers in the disk device names must be unique.

```
example# cat 535_disk 1G_disk > mult_disks
```

IA

To create a disk configuration file (`-d` option) for an IA based system:

1. Locate an IA based system with a disk that you want to test.
2. Create part of the disk configuration file by saving the output of the `fdisk`(1M) command to a file:

```
example# fdisk -R -W 535_disk /dev/rdsk/c0t3d0p0
```

3. Append the output of the `prtvtoc`(1M) command to the disk configuration file.

```
example# prtvtoc /dev/rdsk/c0t3d0s2 >> 535_disk
```

4. (Optional.) Concatenate disk configuration files into a single file to test a profile against multiple disks. The target numbers in the disk device names must be unique.

```
example# cat 535_disk 1G_disk > mult_disks
```

To test a profile with a specific system memory size, set `SYS_MEMSIZE` to the specific memory size (in Mbytes) before running `pfinstall`:

```
example# SYS_MEMSIZE=memory_size

example# export SYS_MEMSIZE
```

NAME

pgxconfig, GFXconfig – configure the PGX32 (Raptor GFX) Graphics Accelerator

SYNOPSIS

```
/usr/sbin/pgxconfig [-dev device-filename] [-res video-mode [try |
    noconfirm | nocheck]] [-file machine | system] [-depth 8 | 24]
    [-24only true | false] [-cachedpixmap true | false]
    [-defaults]

/usr/sbin/pgxconfig [-propt] [-prconf]

/usr/sbin/pgxconfig [-help] [-res ?]

/usr/sbin/pgxconfig [-i]
```

DESCRIPTION

The `pgxconfig` utility configures the PGX32 (Raptor GFX) Graphics Accelerator and some of the X11 window system defaults for PGX32 (Raptor GFX). A previous version of this utility was named `GFXconfig`.

The first form of `pgxconfig` shown in the synopsis above stores the specified options in the OWconfig file. These options are used to initialize the PGX32 (Raptor GFX) device the next time the window system is run on that device. Updating options in the OWconfig file provides persistence of these options across window system sessions and system reboots.

The second, third, and fourth forms, which invoke only the `-prconf`, `-propt`, `-help`, and `-res ?` options, do not update the OWconfig file. For the third form all other options are ignored.

The `-i` option starts `pgxconfig` in interactive mode.

Options may be specified for only one PGX32 (Raptor GFX) device at a time.

Only PGX32 (Raptor GFX)-specific options can be specified through `pgxconfig`. The normal window system options for specifying default depth, default visual class and so forth are still specified as device modifiers on the `openwin` command line. See the `Xsun`(1) manual page available with the SUNWxwman package.

The user can also specify the OWconfig file that is to be updated. By default, the machine-specific file in the `/usr/openwin` directory tree is updated. The `-file` option can be used to specify an alternate file to use. For example, the system-global OWconfig file in the `/etc/openwin` directory tree can be updated instead.

Both of these standard OWconfig files can only be written by root.

OPTIONS

The following options are supported:

`-cachedpixmap` true | false
: When set to `false`, it forces the PGX32 (Raptor GFX) device to use 24–bit only when running OpenWindows. The default value is `true`.

Certain applications make use of a cached pixmap when writing to the display device. Such a technique can cause garbled output and can cause the X server to crash. If you experience such behavior, try setting the `-cachedpixmap` option to `false`.

`-defaults`
Reset all option values to their default values.

`-depth` 8 | 24
Sets the screen depth to 8 or 24 bits per pixel. 24 bits per pixel enables TrueColor graphics in the window system.

`-dev` *device-filename*
Specify the PGX32 (Raptor GFX) special file. The default is `/dev/fbs/gfxp0`, or `/dev/fbs/raptor0` if applicable.

`-file machine | system`
Specify which OWconfig file to update. If `machine`, the machine-specific OWconfig file in the `/etc/openwin` directory tree is used. If `system`, the global OWconfig file in the `/usr/openwin` directory tree is used. If the file does not exist, it is created.

`-help`
Print a list of the `pgxconfig` command line options, along with a brief explanation of each.

`-i`
Start `pgxconfig` in interactive mode.

`-prconf`
Print the PGX32 (Raptor GFX) hardware configuration. Thie following is a typical display:

```
--- Hardware Configuration for /dev/fbs/gfxp0 ---
DAC: version 0x0
Type:
Board:
PROM: version 0x0
PROM Information:
RAM:
EDID Data:
Monitor Sense ID:
Card possible resolutions:  640x480x60, 800x600x75, 1024x768x60
                1024x768x70, 1024x768x75, 1280x1024x75, 1280x1024x76
                1280x1024x60, 1152x900x66, 1152x900x76, 1280x1024x67
                960x680x112S, 960x680x108S, 640x480x60i, 768x575x50i,
                1280x800x76, 1440x900x76, 1600x1000x66, 1600x1000x76,
                vga, svga, 1152, 1280, stereo, ntsc, pal
Monitor possible resolutions:  720x400x70, 720x400x88, 640x480x60
                640x480x67, 640x480x72, 640x480x75, 800x600x56,
                800x600x60, 800x600x72, 800x600x75, 832x624x75,
                1024x768x87, 1024x768x60, 1024x768x70, 1024x768x75,
                1280x1024x75, 1280x1024x76, 1152x900x66, 1152x900x76,
                1280x1024x67, 960x680x112S, vga, svga, 1152, 1280
                stereo
```

```
Current resolution setting: 1280x1024x76
Possible depths: 8, 24, 8+24
Current depth: 8
```

`-propt`
Print the current values of all PGX32 (Raptor GFX) options in the `OWconfig` file specified by the `-file` option for the device specified by the `-dev` option. Print the values of options as they would be in the `OWconfig` file after the call to `pgxconfig` would have completed. The following is a typical display:

```
--- OpenWindows Configuration for /dev/fbs/gfxp0 ---
OWconfig: machine
Video Mode: not set
Depth: 8+24
```

`-res` *video-mode* [`try` | `noconfirm` | `nocheck`]
Specify the built-in video mode used to drive the monitor connected to the specified PGX32 (Raptor GFX) device.

The format for *video-mode* can be one of the following:

width`x`*height*`x`*rate*
The *width* is the screen width in pixels, *height* is the screen height in pixels, and *rate* is the vertical frequency of the screen refresh. As a convenience, `-res` also accepts formats with @ prepended to the refresh rate rather than `x`. For example: 1280x1024@76. The list can be obtained by running `pgxconfig` with the `-res ?` option (the third form shown in the command synopsis above). Note that not all resolutions are supported by both the video board and by the monitor. The `pgxconfig` utility will not permit you to set a resolution not supported by the board unless the `noconfirm` or `nocheck` option is specified. It will also request confirmation before setting a resolution not supported by the monitor if the `nocheck` option is not specified.

Symbolic names
For convenience, the video modes listed below have symbolic names defined. Rather than the form *width*`x`*height*`x`*rate*, the symbolic name may be supplied as the argument to `-res`. If the symbolic name is `none`, the screen resolution will be the video mode that is currently programmed in the device when the window system is run.

`svga`	1024x768x60
`1152`	1152x900x76
`1280`	1280x1024x76
`vga`	640x480x60

`none`	default console resolution

The `-res` option also accepts additional, optional arguments immediately following the video mode specification. Any or all of these may be present.

`noconfirm`	Using the `-res` option, the user could put the system into an unusable state, a state where there is no video output. This can happen if there is ambiguity in the monitor sense codes for the particular code read. To reduce the chance of this occurring, the default behavior of `pgxconfig` is to print a warning message to this effect and to prompt the user to find out if it is okay to continue. The `noconfirm` option instructs `pgxconfig` to bypass this confirmation and to program the requested video mode anyway. This option is useful when `pgxconfig` is being run from a shell script.
`nocheck`	If present, normal error checking based on the monitor sense code is suspended. The video mode specified by the user will be accepted regardless of whether it is appropriate for the currently attached monitor. (This option is useful if a different monitor is to be connected to the PGX32 (Raptor GFX) device). Use of this option implies `noconfirm` as well.
`try`	This option allows the user to test the specified resolution before committing it. It displays a pattern on the screen with the specified resolution. If the test pattern appears correctly, the user may answer "y" to the query. The other permissable answer is "n".

`-res ?`
Print the list of possible resolutions supported by the PGX32 and the monitor.

`-24only`
Force the PGX32 (Raptor GFX) device to use 24 bit only when running Openwindows.

DEFAULTS

For a given invocation of `pgxconfig`, if an option does not appear on the command line, the corresponding OWconfig option is not updated; it retains its previous value, except for `-depth` and `-24only`.

A default value is used if a PGX32 (Raptor GFX) option has not been specified with `pgxconfig` when the window system is run. The option defaults are as follows:

`-dev`	`/dev/fbs/gfxp0`
`-file`	`system`
`-res`	none

The default of none for the `-res` option indicates that when the window system is run, the screen resolution will be the video mode that is currently programmed in the device.

EXAMPLES

EXAMPLE 1 Switching the Resolution on the Monitor Type

The following example switches the monitor type to the resolution of `1280` x `1024` at `76` Hz:

```
example# /usr/sbin/pgxconfig -res 1280x1024x76
```

FILES

`/dev/fbs/gfxp0`
device special file

`/usr/openwin/server/etc/OWconfig`
system configuration file

`/etc/openwin/server/etc/OWconfig`
machine configuration file

SEE ALSO

PGX32 Installation Manual

NAME

picld – PICL daemon

SYNOPSIS

/usr/lib/picl/picld

DESCRIPTION

The Platform Information and Control Library (PICL) provides a mechanism to publish platform-specific information for clients to access in a platform-independent way. `picld` maintains and controls access to the PICL information from clients and plug-in modules. The daemon is started in both single-user and multi-user boot mode.

Upon startup, the PICL daemon loads and initializes the plug-in modules. These modules use the `libpicltree`(3PICLTREE) interface to create nodes and properties in the PICL tree to publish platform configuration information. After the plug-in modules are initialized, the daemon opens the PICL daemon door to service client requests to access information in the PICL tree.

PICL Tree

The PICL tree is the repository of all the nodes and properties created by the plug-in modules to represent the platform configuration. Every node in the PICL tree is an instance of a well-defined PICL class. The name of the base PICL class is `picl`, which defines a basic set of properties that all nodes in the tree must possess. Two of those properties are `name` and `_class`, where `name` contains the name of the node, and the `_class` contains the PICL class name of the node. Certain nodes in the PICL tree have well-known names. For example, the name of the root node of the PICL tree is `/` and the name of the root node of the sub-tree containing platform device nodes is `platform`.

PICL plug–in Modules

The PICL plug-in modules are shared objects that publish platform-specific data in the PICL tree. They are located in well-known directories so that the daemon can locate and load them.

Plug-in modules are located in one of the following plug-in directories depending on the plaform-specific nature of the data that they collect and publish:

```
/usr/platform/'uname -i'/lib/picl/plugins
/usr/platform/'uname -m'/lib/picl/plugins
```

A plug-in module can specify its dependency on another plug-in module using the `-l` or `-R` linker option. The plug-ins are loaded by the daemon using `dlopen`(3DL) according to the specified dependencies. Each plug-in module must define a `.init` section, which is executed when the plug-in module is loaded, to register themselves with the daemon. See `picld_plugin_register`(3PICLTREE) for additional information on plug-in registration.

The plug-in modules use the `libpicltree`(3PICLTREE) interface to publish nodes and properties in the PICL tree so that clients can access them.

When the PICL daemon invokes the initialization routine of the plug-in module, the plug-in collects the platform information and creates nodes and/or properties to represent the configuration in the PICL tree. A plug-in can create additional threads to monitor the platform configuration and update the PICL tree with any changes. This enables a PICL plug-in to operate as a daemon within the PICL framework.

An environmental monitor is an example of a plug-in module that uses a thread to monitor the temperatures and fan speeds of the platform, then publishes the environmental information in the PICL tree so clients can access them.

Clients use the `libpicl`(3PICL) interface to send requests to `picld` for accessing the PICL tree.

EXIT STATUS

`picld` does not return an exit status.

FILES

`/var/run/picld_door`	`PICL` daemon door
`/usr/lib/picl/picld`	`PICL` daemon
`/etc/init.d/picld`	Start/stop script

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpiclu

SEE ALSO

`dlopen`(3DL), `libpicl`(3PICL), `libpicltree`(3PICLTREE), `picld_log`(3PICLTREE), `picld_plugin_register`(3PICLTREE), `attributes`(5)

NAME

ping – send ICMP (ICMP6) ECHO_REQUEST packets to network hosts

SYNOPSIS

/usr/sbin/ping *host* [*timeout*]

/usr/sbin/ping -s [-l | -U] [-adlLnrRv] [-A *addr_family*] [-c *traffic_class*] [-g *gateway* [-g *gateway...*]] [-F *flow_label*] [-I *interval*] [-i *interface*] [-P *tos*] [-p *port*] [-t *ttl*] *host* [*data_size*] [*npackets*]

DESCRIPTION

The utility ping utilizes the ICMP (ICMP6 in IPv6) protocol's ECHO_REQUEST datagram to elicit an ICMP (ICMP6) ECHO_RESPONSE from the specified *host* or network *gateway*. If *host* responds, ping will print:

host is alive

on the standard output and exit. Otherwise, after *timeout* seconds, it will write:

no answer from *host*

The default value of *timeout* is 20 seconds.

When you specify the s flag, sends one datagram per second (adjust with -I) and prints one line of output for every ECHO_RESPONSE that it receives. ping produces no output if there is no response. In this second form, ping computes round trip times and packet loss statistics; it displays a summary of this information upon termination or timeout. The default *data_size* is 56 bytes, or you can specify a size with the *data_size* command-line argument. If you specify the optional *npackets*, ping sends ping requests until it either sends *npackets* requests or receives *npackets* replies.

When using ping for fault isolation, first ping the local host to verify that the local network interface is running.

OPTIONS

The following options are supported:

-A *addr_family*

Specify the address family of the target host. *addr_family* can be either inet or inet6. Address family determines which protocol to use. For an argument of inet, IPv4 is used. For inet6, IPv6 is used.

By default, if the name of a host is provided, not the literal IP address, and a valid IPv6 address exists in the name service database, ping will use this address. Otherwise, if the name service database contains an IPv4 address, it will try the IPv4 address.

Specify the address family inet or inet6 to override the default behavior. If the argument specified is inet, ping will use the IPv4 address associated with the host

name. If none exists, `ping` will state that the host is unknown and exit. It does not try to determine if an IPv6 address exists in the name service database.

If the specified argument is `inet6`, `ping` uses the IPv6 address that is associated with the host name. If none exists, `ping` states that the host is unknown and exits.

`-F` *flow_label*
: Specify the flow label of probe packets. The value must be an integer in the range from 0 to 1048575. This option is valid only on IPv6.

`-I` *interval*
: Turn on the statistics mode and specify the interval between successive transmissions. The default is one second. See the discussion of the `-s` option.

`-L`
: Turn off loopback of multicast packets. Normally, members are in the host group on the outgoing interface, a copy of the multicast packets will be delivered to the local machine.

`-P` *tos*
: Set the type of service (*tos*) in probe packets to the specified value. The default is zero. The value must be an integer in the range from 0 to 255. Gateways also in the path can route the probe packet differently, depending upon the value of *tos* that is set in the probe packet. This option is valid only on IPv4.

`-R`
: Record route. Sets the IPv4 record route option, which stores the route of the packet inside the IPv4 header. The contents of the record route are only printed if the `-v` and `-s` options are given. They are only set on return packets if the target host preserves the record route option across echos, or the `-l` option is given. This option is valid only on IPv4.

`-U`
: Send UDP packets instead of ICMP (ICMP6) packets. `ping` sends UDP packets to consecutive ports expecting to receive back ICMP (ICMP6) `PORT_UNREACHABLE` from the target host.

`-a`
: `ping` all addresses, both IPv4 and IPv6, of the multihomed destination. The output appears as if `ping` has been run once for each IP address of the destination. If this option is used together with `-A`, `ping` probes only the addresses that are of the specified address family. When used with the `-s` option and *npackets* is not specified, `ping` continuously probes the destination addresses in a round robin

	fashion. If *npackets* is specified, `ping` sends *npackets* number of probes to each IP address of the destination and then exits.
`-c` *traffic_class*	Specify the traffic class of probe packets. The value must be an integer in the range from 0 to 255. Gateways along the path can route the probe packet differently, depending upon the value of *traffic_class* set in the probe packet. This option is valid only on IPv6.
`-d`	Set the `SO_DEBUG` socket option.
`-g` *gateway*	Specify a loose source route gateway so that the probe packet goes through the specified host along the path to the target host. The maximum number of gateways is 8 for IPv4 and 127 for IPv6. Note that some factors such as the link MTU can further limit the number of gateways for IPv6.
`-i` *interface_address*	Specify the outgoing interface address to use for multicast packets for IPv4 and both multicast and unicast packets for IPv6. The default interface address for multicast packets is determined from the (unicast) routing tables. *interface_address* can be a literal IP address, for example, `10.123.100.99`, or an interface name, for example, `le0`, or an interface index, for example `2`.
`-l`	Use to send the probe packet to the given host and back again using loose source routing. Usually specified with the `-R` option. If any gateways are specified using `-g`, they are visited twice, both to and from the destination. This option is ignored if the `-U` option is used.
`-n`	Show network addresses as numbers. `ping` normally does a reverse name lookup on the IP addresses it extracts from the packets received. The `-n` option blocks the reverse lookup, so `ping` prints IP addresses instead of host names.
`-p` *port*	Set the base UDP *port* number used in probes. This option is used with the `-U` option. The default base *port* number is 33434. The `ping` utility starts setting the destination port number of UDP packets to this base and increments it by one at each probe.
`-r`	Bypass the normal routing tables and send directly to a host on an attached network. If the host is not on a directly attached network, an error is returned. This

option can be used to `ping` a local host through an interface that has been dropped by the router daemon. See `in.routed`(1M).

`-s` Send one datagram per second and collect statistics.

`-t` *ttl* Specify the IPv4 time to live, or IPv6 hop limit, for unicast and multicast packets. The default time to live (hop limit) for unicast packets can be set with `ndd`(1M) using the `icmp_ipv4_ttl` variable for IPv4 and the `icmp_ipv6_ttl` variable for IPv6. The default time to live (hop limit) for multicast is one hop.

`-v` Verbose output. List any ICMP (ICMP6) packets, other than replies from the target host.

OPERANDS

host The network host

EXAMPLES

EXAMPLE 1 Using `ping` With IPv6

This example shows ping sending probe packets to all the IPv6 addresses of the host london, one at a time. It sends an ICMP6 `ECHO_REQUEST` every second until the user interrupts it.

```
istanbul% ping -s -A inet6 -a london
PING london: 56 data bytes
64 bytes from london (4::114:a00:20ff:ab3d:83ed): icmp_seq=0. time=2. ms
64 bytes from london (fec0::114:a00:20ff:ab3d:83ed): icmp_seq=1. time=1. ms
64 bytes from london (4::114:a00:20ff:ab3d:83ed): icmp_seq=2. time=1. ms
64 bytes from london (fec0::114:a00:20ff:ab3d:83ed): icmp_seq=3. time=1. ms
64 bytes from london (4::114:a00:20ff:ab3d:83ed): icmp_seq=4. time=1. ms
64 bytes from london (fec0::114:a00:20ff:ab3d:83ed): icmp_seq=5. time=1. ms
^C
----london PING Statistics----
6 packets transmitted, 6 packets received, 0% packet loss
round-trip (ms)  min/avg/max = 1/1/2
```

EXIT STATUS

The following exit values are returned:

`0` Successful operation; the machine is alive.

non-zero An error has occurred. Either a malformed argument has been specified, or the machine was not alive.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWbip

SEE ALSO | `ifconfig`(1M), `in.routed`(1M), `ndd`(1M), `netstat`(1M), `rpcinfo`(1M), `traceroute`(1M), `attributes`(5), `icmp`(7P), `icmp6`(7P)

NAME

pkgadd – Transfer software packages to the system

SYNOPSIS

pkgadd [-nv] [-a *admin*] [-d *device*] [[-M] -R *root_path*] [-r *response*] [-V *fs_file*] [pkginst... | -Y *category*[,*category*...]]

pkgadd -s *spool* [-d *device*] [pkginst... | -Y *category*[,*category*...]]

DESCRIPTION

pkgadd transfers the contents of a software package from the distribution medium or directory to install it onto the system. Used without the -d option, pkgadd looks in the default spool directory for the package (var/spool/pkg). Used with the -s option, it writes the package to a spool directory instead of installing it.

Certain unbundled and third-party packages are no longer entirely compatible with the latest version of pkgadd. These packages require user interaction throughout the installation and not just at the very beginning.

To install these older packages (released prior to Solaris 2.4), set the following environment variable: NONABI_SCRIPTS=TRUE

pkgadd permits keyboard interaction throughout the installation as long as this environment variable is set.

OPTIONS

The following options are supported:

-a *admin* — Define an installation administration file, *admin*, to be used in place of the default administration file. The token none overrides the use of any *admin* file, and thus forces interaction with the user. Unless a full path name is given, pkgadd first looks in the current working directory for the administration file. If the specified administration file is not in the current working directory, pkgadd looks in the /var/sadm/install/admin directory for the administration file.

-d *device* — Install or copy a package from *device*. *device* can be a full path name to a directory or the identifiers for tape, floppy disk, or removable disk (for example, /var/tmp or /floppy/*floppy_name*). It can also be a device alias (for example, /floppy/floppy0) or a datastream created by pkgtrans (see pkgtrans(1)).

-M — Instruct pkgadd not to use the $*root_path*/etc/vfstab file for determining the client's mount points. This option assumes the mount points are correct on the server and it behaves consistently with Solaris 2.5 and earlier releases.

-n — Installation occurs in non-interactive mode. The default mode is interactive.

-r *response* — Identify a file or directory which contains output from a previous pkgask(1M) session. This file supplies the interaction responses that would be requested by the package in interactive mode. *response* must be a full pathname.

`-R` *root_path*	Define the full path name of a directory to use as the *root_path*. All files, including package system information files, are relocated to a directory tree starting in the specified *root_path*. The *root_path* may be specified when installing to a client from a server (for example, `/export/root/client1`).
`-s` *spool*	Write the package into the directory *spool* instead of installing it.
`-v`	Trace all of the scripts that get executed by `pkgadd`, located in the *pkginst*`/install` directory. This option is used for debugging the procedural and non-procedural scripts.
`-V` *fs_file*	Specify an alternative *fs_file* to map the client's file systems. For example, used in situations where the $*root_path*`/etc/vfstab` file is non-existent or unreliable.
`-Y` *category*	Install packages based on the value of the `CATEGORY` parameter stored in the package's `pkginfo`(4) file.

When executed without options or operands, `pkgadd` uses `/var/spool/pkg` (the default spool directory).

OPERANDS

The following operands are supported:

pkginst	The package instance or list of instances to be installed. The token `all` may be used to refer to all packages available on the source medium. The format *pkginst*`.*` can be used to indicate all instances of a package. The asterisk character (`*`) is a special character to some shells and may need to be escaped. In the C-Shell, "*" must be surrounded by single quotes (`'`) or preceded by a backslash (`\`).

EXAMPLES

EXAMPLE 1 Installing a Package from a Solaris CD-ROM.

The following example installs a package from a Solaris CD-ROM. You are prompted for the name of the package you want to install.

```
example% pkgadd -d /cdrom/cdrom0/s0/Solaris_2.6
```

EXIT STATUS

`0`	Successful completion
`1`	Fatal error.
`2`	Warning.
`3`	Interruption.
`4`	Administration.
`5`	Administration. Interaction is required. Do not use `pkgadd -n`.
`10`	Reboot after installation of all packages.

20 Reboot after installation of this package.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

pkginfo(1), pkgmk(1), pkgparam(1), pkgproto(1), pkgtrans(1), installf(1M), pkgask(1M), pkgrm(1M), removef(1M), admin(4), pkginfo(4), attributes(5)

Application Packaging Developer's Guide

NOTES

When transferring a package to a spool directory, the -r, -n, and -a options cannot be used.

The -r option can be used to indicate a directory name as well as a filename. The directory can contain numerous response files, each sharing the name of the package with which it should be associated. This would be used, for example, when adding multiple interactive packages with one invocation of pkgadd. In this situation, each package would need a response file. If you create response files with the same name as the package (for example, pkinst1 and pkinst2), then name the directory in which these files reside after the -r.

The -n option causes the installation to halt if any interaction is needed to complete it.

If the default *admin* file is too restrictive, the administration file may need to be modified to allow for total non-interaction during a package installation. See admin(4) for details.

NAME

pkgask – stores answers to a request script

SYNOPSIS

pkgask [-d *device*] [-R *root_path*] -r *response pkginst*...

DESCRIPTION

pkgask allows the administrator to store answers to an interactive package (one with a request script, that is, a user-created file that must be named request). Invoking this command generates a response file that is then used as input at installation time. The use of this response file prevents any interaction from occurring during installation since the file already contains all of the information the package needs.

OPTIONS

The following options are supported

-d *device* Run the request script for a package on *device*. *device* can be a directory pathname or the identifiers for tape, floppy disk or removable disk (for example, /var/tmp, /dev/diskette, and /dev/dsk/c1d0s0). The default device is the installation spool directory.

-R *root_path* Define the full path name of a directory to use as the *root_path*. All files, including package system information files, are relocated to a directory tree starting in the specified *root_path*.

-r *response* Identify a file or directory which should be created to contain the responses to interaction with the package. The name must be a full pathname. The file, or directory of files, can later be used as input to the pkgadd(1M) command.

OPERANDS

The following operands are supported:

pkginst Specify the package instance, or list of instances for which request scripts will be created. The token all may be used to refer to all packages available on the source medium.

EXIT STATUS

0 Successful completion.

>0 An error occurred.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

pkginfo(1), pkgmk(1), pkgparam(1), pkgproto(1), pkgtrans(1), installf(1M), pkgadd(1M), pkgchk(1M), pkgrm(1M), removef(1M), admin(4),attributes(5)

Application Packaging Developer's Guide

NOTES

The -r option can be used to indicate a directory name as well as a filename. The directory name is used to create numerous response files, each sharing the name of the package with which it should be associated. This would be used, for example, when

you will be adding multiple interactive packages with one invocation of `pkgadd`(1M). Each package would need a response file. To create multiple response files with the same name as the package instance, name the directory in which the files should be created and supply multiple instance names with the `pkgask` command. When installing the packages, you will be able to identify this directory to the `pkgadd`(1M) command.

If the default `admin` file is too restrictive, the administration file may need to be modified to allow for total non-interaction during a package installation. See `admin`(4) for details.

NAME

pkgchk – check package installation accuracy

SYNOPSIS

```
pkgchk [-l | -acfnqvx] [-i file] [-p path...] [-R root_path] [ [-m pkgmap
     [-e envfile]] | pkginst... | -Y category,category...]

pkgchk -d device [-l | -fv] [-i file] [-M] [-p path...] [-V fs_file]
     [pkginst... | -Y category[,category...]]
```

DESCRIPTION

`pkgchk` checks the accuracy of installed files or, by using the `-l` option, displays information about package files. `pkgchk` checks the integrity of directory structures and files. Discrepancies are written to standard error along with a detailed explanation of the problem.

The first synopsis defined above is used to list or check the contents and/or attributes of objects that are currently installed on the system, or in the indicated `pkgmap`. Package names may be listed on the command line, or by default, the entire contents of a machine will be checked.

The second synopsis is used to list or check the contents of a package which has been spooled on the specified device, but not installed. Note that attributes cannot be checked for spooled packages.

OPTIONS

The following options are supported:

`-a` — Audit the file attributes only and do not check file contents. Default is to check both.

`-c` — Audit the file contents only and do not check file attributes. Default is to check both.

`-d` *device* — Specify the device on which a spooled package resides. *device* can be a directory path name or the identifiers for tape, floppy disk, or removable disk (for example, `/var/tmp` or `/dev/diskette`).

`-e` *envfile* — Request that the package information file named as *envfile* be used to resolve parameters noted in the specified `pkgmap` file.

`-f` — Correct file attributes if possible. If used with the `-x` option, this option removes hidden files. When `pkgchk` is invoked with this option, it creates directories, named pipes, links, and special devices if they do not already exist. If the `-d` option calls out an uninstalled package, the `-f` option will only take effect if the package is in directory (not stream) format. All file attributes will be set to agree with the entries in the `pkgmap` file except that setuid, setgid, and sticky bits will not be set in the mode.

`-i file` — Read a list of path names from `file` and compare this list against the installation software database or the indicated `pkgmap` file. Path names which are not contained in `file` are not checked.

`-l` — List information on the selected files that make up a package. This option is not compatible with the `-a`, `-c`, `-f`, `-g`, and `-v` options.

`-m pkgmap`	Check the package against the package map file, `pkgmap`.
`-M`	Instruct `pkgchk` not to use the `$`*root_path*`/etc/vfstab` file for determining the client's mount points. This option assumes the mount points are correct on the server and it behaves consistently with Solaris 2.5 and earlier releases.
`-n`	Do not check volatile or editable files' contents. This should be used for most post-installation checking.
`-p` *path*	Only check the accuracy of the path name or path names listed. *path* can be one or more path names separated by commas (or by white space, if the list is quoted).
`-q`	Quiet mode. Do not give messages about missing files.
`-R` *root_path*	Define the full name of a directory to use as the *root_path*. All files, including package system information files, are relocated to a directory tree starting in the specified *root_path*. The *root_path* may be specified when installing to a client from a server (for example, `/export/root/client1`).
`-v`	Verbose mode. Files are listed as processed.
`-V` *fs_file*	Specify an alternative *fs_file* to map the client's file systems. For example, used in situations where the `$`*root_path*`/etc/vfstab` file is non-existent or unreliable.
`-x`	Search exclusive directories, looking for files which exist that are not in the installation software database or the indicated `pkgmap` file.
`-Y` *category*	Check packages based on the value of the `CATEGORY` parameter stored in the installed or spooled package's `pkginfo`(4) file.

OPERANDS

pkginst	The package instance or instances to be checked. The format *pkginst* `.*` can be used to check all instances of a package. The default is to display all information about all installed packages. The asterisk character (`*`) is a special character to some shells and may need to be escaped. In the C-Shell, "*" must be surrounded by single quotes (`'`) or preceded by a backslash (`\`);

EXAMPLES

EXAMPLE 1 Using `pkgchk` for Displaying Package Installation Information

The following example displays package installation information for `/usr/bin/ls`:

```
example% pkgchk -l -p /usr/bin/ls
```

EXIT STATUS

`0`	Successful completion.
`>0`	An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`pkginfo`(1), `pkgtrans`(1), `pkgadd`(1M), `pkgask`(1M), `pkgrm`(1M), `pkginfo`(4), `attributes`(5)

Application Packaging Developer's Guide

NAME

pkgrm – remove a package from the system

SYNOPSIS

pkgrm [-nv] [-a *admin*] [[-A | -M]-R *root_path*] [-V *fs_file*] [pkginst… | -Y *category*[,*category*…]]

pkgrm -s *spool* [pkginst… | -Y *category*[,*category*…]]

DESCRIPTION

pkgrm will remove a previously installed or partially installed package from the system. A check is made to determine if any other packages depend on the one being removed. If a dependency exists, the action taken is defined in the admin file.

The default state for the command is in interactive mode, meaning that prompt messages are given during processing to allow the administrator to confirm the actions being taken. Non-interactive mode can be requested with the -n option.

The -s option can be used to specify the directory from which spooled packages should be removed.

Certain unbundled and third-party packages are no longer entirely compatible with the latest version of pkgrm. These packages require user interaction throughout the removal and not just at the very beginning.

To remove these older packages (released prior to Solaris 2.4), set the following environment variable:NONABI_SCRIPTS=TRUE pkgrm permits keyboard interaction throughout the removal as long as this environment variable is set.

OPTIONS

The following options are supported:

-a *admin*
: Use the installation administration file, admin, in place of the default admin file. pkgrm first looks in the current working directory for the administration file. If the specified administration file is not in the current working directory, pkgrm looks in the /var/sadm/install/admin directory for the administration file.

-A
: Remove the package files from the client's file system, absolutely. If a file is shared with other packages, the default behavior is to not remove the file from the client's file system.

-M
: Instruct pkgrm not to use the $*root_path*/etc/vfstab file for determining the client's mount points. This option assumes the mount points are correct on the server and it behaves consistently with Solaris 2.5 and earlier releases.

-n
: Non-interactive mode. If there is a need for interaction, the command will exit.

  Use of this option requires that at least one package instance be named upon invocation of the command. Certain conditions must exist for a package to be removed non-interactively or a non-restrictive admin file needs to be used.

`-R` *root_path*	Defines the full path name of a directory to use as the *root_path*. All files, including package system information files, are relocated to a directory tree starting in the specified *root_path*.
`-s` *spool*	Remove the specified package(s) from the directory *spool*. The default directory for spooled packages is `/var/sadm/pkg`.
`-v`	Trace all of the scripts that get executed by `pkgrm`, located in the *pkginst*`/install` directory. This option is used for debugging the procedural and non-procedural scripts.
`-V` *fs_file*	Specify an alternative *fs_file* to map the client's file systems. Used in situations where the $*root_path*`/etc/vfstab` file is non-existent or unreliable.
`-Y` *category*	Remove packages based on the value of the `CATEGORY` parameter stored in the installed or spooled package's `pkginfo`(4) file. No package with the `CATEGORY` value of `system` can removed from the file system with this option.

OPERANDS

The following operand is supported:

pkginst	Specifies the package to be removed. The format *pkginst*`.*` can be used to remove all instances of a package. The asterisk character (`*`) is a special character to some shells and may need to be escaped. In the C-Shell, "*" must be surrounded by single quotes (`'`) or preceded by a backslash (\).

EXAMPLES

EXAMPLE 1 Removing All instances of `SUNWjunk` From `client1`

The following example removes all instances of `SUNWjunk` from `client1`:

```
example% pkgrm -R /export/root/client1 SUNWjunk*
```

EXIT STATUS

The following exit values are returned:

`0`	Successful completion.
`1`	Fatal error.
`2`	Warning.
`3`	Interruption.
`4`	Administration.
`10`	Reboot after removal of all packages.
`20`	Reboot after removal of this package.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`pkginfo`(1), `pkgmk`(1), `pkgparam`(1), `pkgproto`(1), `pkgtrans`(1), `installf`(1M), `pkgadd`(1M), `pkgask`(1M), `pkgchk`(1M), `removef`(1M), `admin`(4),`pkginfo`(4), `attributes`(5)

Application Packaging Developer's Guide

NAME

pmadm – port monitor administration

SYNOPSIS

pmadm -a [-p *pmtag* | -t *type*] -s *svctag* -i *id* -m *pmspecific* -v *ver* [-f xu] [-y *comment*] [-z *script*]

pmadm -r -p *pmtag* -s *svctag*

pmadm -e -p *pmtag* -s *svctag*

pmadm -d -p *pmtag* -s *svctag*

pmadm -l [-t *type* | -p *pmtag*] [-s *svctag*]

pmadm -L [-t *type* | -p *pmtag*] [-s *svctag*]

pmadm -g -p *pmtag* -s *svctag* [-z *script*]

pmadm -g -s *svctag* -t *type* -z *script*

DESCRIPTION

`pmadm` is the administrative command for the lower level of the Service Access Facility hierarchy, that is, for service administration. A port may have only one service associated with it although the same service may be available through more than one port. In order to uniquely identify an instance of a service, the `pmadm` command must identify both the port monitor or port monitors through which the service is available (`-p` or `-t`) and the service (`-s`). See OPTIONS.

`pmadm` performs the following functions:

- adds or removes a service
- enables or disables a service
- installs or replaces a per-service configuration script
- prints requested service information

Any user on the system may invoke `pmadm` to request service status (`-l` or `-L`) or to print per-service configuration scripts (`-g` without the `-z` option). `pmadm` with other options may be executed only by a privileged user.

OPTIONS

The following options are supported:

`-a` Add a service. `pmadm` adds an entry for the new service to the port monitor's administrative file. Because of the complexity of the options and arguments that follow the `-a` option, it may be convenient to use a command script or the menu system to add services.

`-d` Disable a service. Add `x` to the flag field in the entry for the service *svctag* in the port monitor's administrative file. This is the entry used by port monitor *pmtag*. See the `-f` option, below, for a description of the flags available.

`-e` Enable a service. Remove `x` from the flag field in the entry for the service *svctag* in the port monitor administrative file. This is the entry used by port monitor *pmtag*. See the `-f` option, below, for a description of the flags available.

`-f xu`

The `-f` option specifies one or both of the following two flags which are then included in the flag field of the entry for the new service in the port monitor's administrative file. If the `-f` option is not included, no flags are set and the default conditions prevail. By default, a new service is enabled and no `utmpx` entry is created for it. An `-f` option without a following argument is illegal.

`x` Do not enable the service *svctag* available through port monitor *pmtag*.

`u` Create a `utmpx` entry for service *svctag* available through port monitor *pmtag*.

`-g`

Print, install, or replace a per-service configuration script. The `-g` option with a `-p` option and a `-s` option prints the per-service configuration script for service *svctag* available through port monitor *pmtag*. The `-g` option with a `-p` option, a `-s` option, and a `-z` option installs the per-service configuration script contained in the file `script` as the per-service configuration script for service *svctag* available through port monitor *pmtag*. The `-g` option with a `-s` option, a `-t` option, and a `-z` option installs the file `script` as the per-service configuration script for service *svctag* available through any port monitor of type `type`. Other combinations of options with `-g` are invalid.

`-i` *id*

`id` is the identity that is to be assigned to service *svctag* when it is started. `id` must be an entry in `/etc/passwd`.

`-l`

The `-l` option requests service information. Used by itself and with the options described below, it provides a filter for extracting information in several different groupings.

`-l`	By itself, the `-l` option lists all services on the system.
`-l -p` *pmtag*	Lists all services available through port monitor *pmtag*.
`-l -s` *svctag*	Lists all services with tag *svctag*.
`-l -p` *pmtag*`-s`*svctag*	Lists service *svctag*.
`-l -t` *type*	Lists all services available through port monitors of type `type`.
`-l -t` *type*`-s`*svctag*	Lists all services with tag *svctag* available through a port monitor of type `type`.

Other combinations of options with `-l` are invalid.

`-L`

The `-L` option is identical to the `-l` option except that output is printed in a condensed format.

`-m` *pmspecific*	*pmspecific* is the port monitor-specific portion of the port monitor administrative file entry for the service.
`-p` *pmtag*	Specifies the tag associated with the port monitor through which a service (specified as `-s` *svctag*) is available.
`-r`	Remove a service. When `pmadm` removes a service, the entry for the service is removed from the port monitor's administrative file.
`-s` *svctag*	Specifies the service tag associated with a given service. The service tag is assigned by the system administrator and is part of the entry for the service in the port monitor's administrative file.
`-t` *type*	Specifies the the port monitor type.
`-v` *ver*	Specifies the version number of the port monitor administrative file. The version number may be given as `-v '`*pmspec* `-V‘`where *pmspec* is the special administrative command for port monitor *pmtag*. This special command is `ttyadm` for `ttymon` and `nlsadmin` for `listen`. The version stamp of the port monitor is known by the command and is returned when *pmspec* is invoked with a `-V` option.
`-y` *comment*	Associate *comment* with the service entry in the port monitor administrative file.
`-z` *script*	Used with the `-g` option to specify the name of the file that contains the per-service configuration script. Modifying a configuration script is a three-step procedure. First a copy of the existing script is made (`-g` alone). Then the copy is edited. Finally, the copy is put in place over the existing script (`-g` with `-z`).

Options that request information write the requested information to the standard output. A request for information using the `-l` option prints column headers and aligns the information under the appropriate headings. In this format, a missing field is indicated by a hyphen. A request for information in the condensed format using the `-L` option prints the information in colon-separated fields; missing fields are indicated by two successive colons. # is the comment character.

EXAMPLES

EXAMPLE 1 Adding a Service to a Port Monitor with the Tag `pmtag`

The following command adds a service to a port monitor with tag `pmtag` and gives the service the tag `svctag`. The port monitor-specific information is generated by `specpm`. The service defined by `svctag` will be invoked with identity `root`.

```
pmadm -a -p pmtag -s svctag -i root -m ‘specpm -a arg1 -b arg2‘-v ‘specpm -V‘
```

EXAMPLE 2 Adding a Service with Service Tab `svctag`

The following command adds a service with service tag `svctag`, identity `guest`, and port monitor-specific information generated by `specpm` to all port monitors of type `type`:

```
pmadm -a -s svctag -i guest -t type -m ‘specpm -a arg1 -b arg2‘-v ‘specpm -V‘
```

EXAMPLE 3 Removing a Service

The following command removes the service `svctag` from port monitor `pmtag`:

```
pmadm -r -p pmtag -s svctag
```

EXAMPLE 4 Enabling a Service

The following command enables the service `svctag` available through port monitor `pmtag`:

```
pmadm -e -p pmtag -s svctag
```

EXAMPLE 5 Disabling a Service

The following command disables the service `svctag` available through port monitor `pmtag`:

```
pmadm -d -p pmtag -s svctag
```

EXAMPLE 6 Listing Status Information

The following command lists status information for all services:

```
pmadm -l
```

EXAMPLE 7 Listing Status Information

The following command lists status information for all services available through the port monitor with tag `ports`:

```
pmadm -l -p ports
```

EXAMPLE 8 Listing Status Information in Condensed Format

The following command lists the status information for all services available through the port monitor with tag `ports` in condensed format:

```
pmadm -L -p ports
```

EXAMPLE 9

List status information for all services available through port monitors of type `listen`:

```
pmadm -l -t listen
```

EXAMPLE 10 Printing the per-service Configuration

The following command prints the per-service configuration script associated with the service `svctag` available through port monitor `pmtag`:

```
pmadm -g -p pmtag -s svctag
```

EXIT STATUS

The following exit values are returned:

`0` Successful operation.

`>0` Operation failed.

FILES

`/etc/saf/`*pmtag*`/_config`

`/etc/saf/`*pmtag*`/`*svctag*

`/var/saf/`*pmtag*`/*`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`sac`(1M), `sacadm`(1M), `doconfig`(3NSL), `attributes`(5)

NAME

pmconfig – Configure the Power Management system

SYNOPSIS

/usr/sbin/pmconfig [-r]

DESCRIPTION

The pmconfig utility sets the Power Management and suspend-resume configuration. User has permission to change Power Management configuration using pmconfig only if he is allowed to do so according to PMCHANGEPERM keyword of /etc/default/power. User has permission to change the suspend-resume configuration using pmconfig only if he is allowed to do so according to the CPRCHANGEPERM keyword of /etc/default/power. See FILES section below for a description of the PMCHANGEPERM and CPRCHANGEPERM keywords of /etc/default/power.

Based on user permissions, pmconfig first resets the Power Management and/or suspend-resume state back to its default and then reads the new Power Management and/or suspend-resume configuration from /etc/power.conf and issues the commands to activiate the new configuration. The pmconfig utility is run at system boot. This utility can also be run from the command line after manual changes have been made to the /etc/power.conf file. For editing changes made to the /etc/power.conf file to take effect, users must run pmconfig.

The preferred interface for changing Power Management and suspend-resume configuration is dtpower(1M).

OPTIONS

The following options are supported:

-r Reset Power Management and suspend-resume state to default and exit. User must have both Power Management and suspend-resume configuration permission for this option.

EXIT STATUS

The following exit values are returned:

0 Upon successful completion

>0 An error occurred

FILES

/etc/power.conf System Power Management configuration file

/etc/default/power File that controls permissions for system's Power Management and suspend-resume features. The PMCHANGEPERM keyboard controls the Power Management configuration permissions, while the CPRCHANGEPERM keyword controls the suspend-resume configuration permissions.

Allowed values are:

all Any user can change the configuration.

- No one except super-user can change the configuration.

`<user1, user2,...>`	A user in this user list or a super-user can change the configuration. The user list is a space and/or comma (,) separated list. You must enclose the list in < and > characters.
`console-owner`	A user who owns the system console device node or a super-user can change the configuration.

The default values are `PMCHANGEPERM=console-owner` and `CPRCHANGEPERM=console-owner`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpmu
Interface stability	Unstable

SEE ALSO

`powerd`(1M), `power.conf`(4), `attributes`(5), `cpr`(7), `pm`(7D)

Using Power Management

DIAGNOSTICS

If the program cannot open the configuration file, it prints an error message to standard error. If the program encounters a syntax error in the configuration file, it prints an error message and the line number of the error in the configuration file. It then skips the rest of the information on that line and processes the next line. Any configuration information already processed on the line containing the error is used. If user does not have permission to change Power Management and/or suspend-resume configuration, and configuration file has entries for which user doesn't have permission, it process the entries for which user has permissions and prints error on rest.

NAME

pntadm – DHCP network table management utility

SYNOPSIS

pntadm -C [-r *resource*] [-p *path*] [-u *uninterpreted*] *network*

pntadm -A *name_IP_address* [-c *comment*] [-e *mm/dd/yyyy*] [-f *num* | *keywords*] [-h *client_hostname*] [-i [-a] *client_ID*] [-m [-y] *macro*] [-s *server*] [-r *resource*] [-p *path*] [-u *uninterpreted*] *network*

pntadm -M *name_IP_address* [-c *comment*] [-e *mm/dd/yyyy*] [-f *num* | *keywords*] [-h *client_hostname*] [-i [-a] *client ID*] [-m [-y] *macro*] [-n *new_client_IP_address*] [-s *server*] [-r *resource*] [-p *path*] [-u uninterpreted] *network*

pntadm -D *name_IP_address* [-y] [-r *resource*] [-p *path*] [-u uninterpreted] *network*

pntadm -P [-v] [-x] [-r *resource*] [-p *path*] [-u uninterpreted] *network*

pntadm -R [-r *resource*] [-p *path*] [-u uninterpreted] *network*

pntadm -L [-r *resource*] [-p *path*] [-u uninterpreted]

pntadm -B [-v] [*batchfile*]

DESCRIPTION

The pntadm command is used to manage the Dynamic Host Configuration Protocol (DHCP) network tables. It is used to add and remove networks under DHCP management, and add, delete, or modify IP address records within network tables, or to view tables. For a description of the format of DHCP network tables, see dhcp_network(4).

pntadm can be run as root or by other users assigned to the DHCP Management profile. See rbac(5) and user_attr(4).

If the networks you want to add are subnetted, you need to update the netmasks(4) table.

One of the following options (function flags) must be specified with the pntadm command: -A, -B, -C, -D, -L, -M, -P, or -R.

OPTIONS

The following options are supported:

-A *name_IP_address*
: Add a client entry with hostname or client IP address, *name_IP_address*, to the named DHCP network table.

  The following sub-options are optional:

  -c *comment*
  : Comment text. The default is NULL.

  -e *mm/dd/yyyy*
  : Absolute lease. The default is 0.

  -f *num* | *keywords*
  : Flag value. The default is 00.

The flag (`-f`) option can be specified either as a single number denoting the intended flag value, or as a series of the following keywords, combined using the plus (+) symbol:

`DYNAMIC` or `00`
Server manager's assignment.

`PERMANENT` or `01`
Lease on entry is permanent.

`MANUAL` or `02`
Administrator managed assignment.

`UNUSABLE` or `04`
Entry is not valid.

`BOOTP` or `08`
Entry reserved for `BOOTP` clients.

For a more detailed description of the flag values, see `dhcp_network`(4).

`-h` *client_hostname*
Client hostname. The default is NULL.

When the `-h` option is used in this mode, the *client_hostname* is added to the hosts table within the resource used for storing host names (files, NIS+ or DNS). The command will fail if this *client_hostname* is already present in the hosts table.

`-i` *client_ID* [`-a`]
Client identifier [`-a`]. The default is `00`.

The `-i` option modified with `-a` specifies that the client identifier is in ASCII format, and thus needs to be converted to hexadecimal format before insertion into the table.

`-m` *macro* [`-y`]
Macro name. Default is UNKNOWN.

The `-m` option modified with `-y` verifies the existence of the named macro in the `dhcptab` table before adding the entry.

`-s` *server*
Server IP or name. Default is system name (`uname -n`).

`-B`
Activate batch mode. `pntadm` will read from the specified file or from standard input a series of `pntadm`

commands and execute them within the same process. Processing many `pntadm` commands using this method is much faster than running an executable batchfile itself. Batch mode is recommended for using `pntadm` in scripts.

The following sub-option is optional:

`-v`
Display commands to standard output as they are processed.

`-C`
Create the DHCP network table for the network specified by *network*. See OPERANDS. For details, see `dhcp_network`(4) and `networks`(4).

`-D` *name_IP_address*
Delete the specified client entry with hostname or client IP address, *name_IP_address*, in the named DHCP network table. (See `dhcp_network`(4).)

The following sub-option is optional:

`-y`
Remove associated host table entry. The `-y` option requests that all hostnames associated with the IP address in the hosts table in the resource be removed.

`-L`
List the DHCP network tables presently configured, one per line, on standard output. If none are found, no output is printed and an exit status of `0` is returned.

`-M` *name_IP_address*
Modify the specified client entry with hostname or client IP address, *name_IP_address*, in the named DHCP network table. See `dhcp_network`(4). The default for the sub-options is what they currently are set to.

The following sub-options are optional.

`-c` *comment*
New comment text.

`-e` *mm/dd/yy*
New absolute lease expiration date. Time defaults to 12:00 AM of the day specified.

`-f` *num* | *keyboard*
New flag value, see explanation following the description of the `-A` option.

`-h` *host_name*
New client hostname.

The `-h` option allows you to change the current *hostname* associated with the IP address or to add a new *hostname* to the hosts table if an entry associated with this IP address does not exist.

`-i` *client_ID*
New client identifier [`-a`].

`-m` *macro* [`-y`]
Macro name defined in `dhcptab`.

`-n` *new_client_IP_address*
New IP address.

`-s` *server*
New server IP or name.

For more detailed description of the sub-options and flag values, see `dhcp_network`(4).

`-P` — Display the named DHCP network table. See `dhcp_network`(4).

The following sub-options are optional:

`-v` — Display lease time in full verbose format.

`-x` — Display lease time in raw format.

`-p` *path* — Override the `dhcpsvc.conf`(4) configuration value for data store resource path, *path* See `dhcpsvc.conf`(4)

`-R` — Remove the named DHCP network table. See `dhcp_network`(4).

`-r` *data_store_resource* — Override the `/etc/inet/dhcpsvc.conf` configuration value for `RESOURCE=` with the *data_store_resource* specified. See the `dhcpsvc.conf`(4) man page for more details on resource type, and the *Solaris DHCP Service Developer's Guide* for more information about adding support for other data stores.

`-u` uninterpreted — Data which will be ignored by `pntadm`, but passed to the currently configured public module to be interpreted by the data store. This might be used for a database account name or other authentication or authorization parameters required by a particular data store.

OPERANDS The following operand is supported:

network The network address or network name which corresponds to the `dhcp network` table. See `dhcp_network`(4).

EXAMPLES

EXAMPLE 1 Creating a Table for the `10.0.0.0` DHCP Network

The following command creates a table for the `10.0.0.0` (subnetted to class C) DHCP network table. Note that if you have an alias for this network in your `networks`(4) table, you can use that value rather than the dotted Internet Address notation.

```
example# pntadm -C 10.0.0.0
```

EXAMPLE 2 Adding an Entry to the `10.0.0.0` Table

The following command adds an entry to the `10.0.0.0` table in the `files` resource in the `/var/mydhcp` directory:

```
example# pntadm -r SUNWfiles -p /var/mydhcp -A 10.0.0.1 10.0.0.0
```

EXAMPLE 3 Modifying the `10.0.0.1` Entry of the `10.0.0.0` Table

The following command modifies the `10.0.0.1` entry of the `10.0.0.0` table, changing the macro name to `Green`, setting the flags field to `MANUAL` and `PERMANENT:`

```
example# pntadm -M 10.0.0.1 -m Green -f 'PERMANENT + MANUAL' 10.0.0.0
```

EXAMPLE 4 Changing the `10.0.0.1` Entry to `10.0.0.2`

The following command changes the `10.0.0.1` entry to `10.0.0.2`, making an entry in the `hosts`(4) table called `myclient`:

```
example# pntadm -M 10.0.0.1 -n 10.0.0.2 -h myclient 10.0.0.0
```

EXAMPLE 5 Setting the Client ID as ASCII

The following command sets the client ID as ASCII `aruba.foo.com` for the `myclient` entry:

```
example# pntadm -M myclient -i 'aruba.foo.com' -a 10.0.0.0
```

EXAMPLE 6 Deleting the `myclient`Entry from the `10.0.0.0` Table

The following command deletes the `myclient` (`10.0.0.2`) entry from the `10.0.0.0` table:

```
example# pntadm -D myclient 10.0.0.0
```

EXAMPLE 6 Deleting the `myclient`Entry from the `10.0.0.0` Table *(Continued)*

EXAMPLE 7 Removing the Named DHCP Network Table

The following command removes the named DHCP network table in the NIS+ directory specified:

```
example# pntadm -r SUNWnisplus -p Test.Nis.Plus. -R 10.0.0.0
```

EXAMPLE 8 Listing the Configured DHCP Network Tables

The following command lists the configured DHCP network tables:

```
example# pntadm -L
192.168.0.0
10.0.0.0
```

EXAMPLE 9 Executing pntadm Commands in Batch Mode

The following command runs a series of `pntadm` commands contained in a batch file:

```
example# pntadm -B addclients
```

EXIT STATUS

`0` Successful completion.

`1` Object already exists.

`2` Object does not exist.

`3` Non-critical error.

`4` Critical error.

FILES

`/etc/inet/dhcpsvc.conf`

`/etc/inet/hosts`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWdhcsu
Interface Stability	Evolving

SEE ALSO

`dhcpconfig`(1M), `dhcpmgr`(1M), `dhcp_network`(4), , `dhcpsvc.conf`(4), `dhcptab`(4), `hosts`(4), `netmasks`(4), `networks`(4), `user_attr`(4), `attributes`(5), `dhcp`(5), `dhcp_modules`(5), `rbac`(5)

Solaris DHCP Service Developer's Guide

System Administration Guide: IP Services

Alexander, S., and R. Droms, *DHCP Options and BOOTP Vendor Extensions*, RFC 1533, Lachman Technology, Inc., Bucknell University, October 1993.

Droms, R., *Interoperation Between DHCP and BOOTP*, RFC 1534, Bucknell University, October 1993.

Droms, R., *Dynamic Host Configuration Protocol*, RFC 1541, Bucknell University, October 1993.

Wimer, W., *Clarifications and Extensions for the Bootstrap Protocol*, RFC 1542, Carnegie Mellon University, October 1993.

NAME

pooladm – activate and deactivate the resource pools facility

SYNOPSIS

/usr/sbin/pooladm [-n] [-c [*filename*] | -x]

DESCRIPTION

The `pooladm` command provides administrative operations on pools and sets. `pooladm` reads the specified filename and attempts to activate the pool configuration contained in it.

Before updating the current pool run-time configuration, `pooladm` validates the configuration for correctness.

Without options, `pooladm` prints out the current running pools configuration.

OPTIONS

The following options are supported:

`-c` Instantiate the configuration at the given location. If a filename is not specified, it defaults to `/etc/pooladm.conf`.

`-n` Validate the configuration without actually updating the current active files.

`-x` Remove the currently active pool configuration. Destroy all defined resources, and return all formerly partitioned components to their default resources.

OPERANDS

The following operands are supported:

filename Use the configuration contained within this file.

EXAMPLES

EXAMPLE 1 Instantiating a Configuration

The following command instantiates the configuration contained at `/home/admin/newconfig`:

```
example# /usr/sbin/pooladm -c /home/admin/newconfig
```

EXAMPLE 2 Validating the Configuration Without Instantiating It

The following command attempts to instantiate the configuration contained at `/home/admin/newconfig`. It displays any error conditions that it encounters, but does not actually modify the active configuration.

```
example# /usr/sbin/pooladm -n -c /home/admin/newconfig
```

EXAMPLE 3 Removing the Current Configuration

The following command removes the current pool configuration:

```
example# /usr/sbin/pooladm -x
```

FILES

`/etc/pooladm.conf`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpool
Interface Stability Invocation Output	Evolving Unstable

SEE ALSO

`poolcfg`(1M), `poolbind`(1M), `libpool`(3LIB), `attributes`(5),

System Administration Guide: Resource Management and Network Services

NOTES

Resource bindings that are not presented in the form of a binding to a partitionable resource, such as the scheduling class, are not necessarily be modified in a `pooladm -x` operation.

NAME

poolbind – bind processes, tasks, or projects or query binding of processes to resource pools

SYNOPSIS

/usr/sbin/poolbind `-p` *poolname* `[-i` *idtype*`]` *id*...

/usr/sbin/poolbind `-q` *pid*...

/usr/sbin/poolbind `-Q` *pid*...

DESCRIPTION

The `poolbind` command allows an authorized user to bind projects, tasks, and processes to pools. It can also allow a user to query a process to determine which pool the process is bound to.

OPTIONS

The following options are supported:

`-i` *idtype*
: This option, together with the *idlist* arguments, specifies one or more processes to which the poolbind command is to apply. The interpretation of *idlist* depends on the value of *idtype*. The valid *idtype* arguments and corresponding interpretations of *idlist* are as follows:

  pid
  : *idlist* is a list of process IDs. Binds the specified processes to the specified pool. This is the default behavior if no idtype is specified.

  taskid
  : *idlist* is a list of task IDs. Bind all processes within the list of task IDs to the specified pool.

  projid
  : *idlist* is a list of project IDs. Bind all processes within the list of projects to the specified pool. Each project ID can be specified as either a project name or a numerical project ID. See `project`(4).

`-q` *pid* ...
: Queries the pool bindings for a given list of process IDs. If the collection of resources associated with the process does not correspond to any currently existing pool, or if there are multiple pools with the set of resources that the process is bound to, the query fails for that particular process ID.

`-Q` *pid* ...
: Queries the resource bindings for a given list of process IDs. The resource bindings are each reported on a separate line.

OPERANDS

The following operands are supported:

poolname
: The name of a pool to which the specified project, tasks or processes are to be bound.

EXAMPLES

EXAMPLE 1 Binding All Processes

The following command binds all processes in projects `5` and `7` to `pool web_app`:

```
example# /usr/sbin/poolbind -p web_app -i projid 5 7
```

EXAMPLE 2 Binding the Running Shell

The following command binds the running shell to `pool web_app`:

```
example# /usr/sbin/poolbind -p web_app $$
```

EXAMPLE 3 Querying the Pool Bindings

The following command queries the bindings to verify that the shell is bound to the given pool:

```
example# /usr/sbin/poolbind -q $$
```

EXAMPLE 4 Querying the Resource Bindings

The following command queries the bindings to verify that the shell is bound to the given resources:

```
example# /usr/sbin/poolbind -Q $$
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` Requested operation could not be completed.

`2` Invalid command line options were specified.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpool
Interface Stability Invocation Output	Evolving Unstable

SEE ALSO

`pooladm`(1M), `poolcfg`(1M), `libpool`(3LIB), `project`(4), `attributes`(5)

System Administration Guide: Resource Management and Network Services

NAME

poolcfg – create and modify resource pool configuration files

SYNOPSIS

/usr/sbin/poolcfg `-c` *command filename*

/usr/sbin/poolcfg `-f` *command_file filename*

/usr/sbin/poolcfg `-h`

DESCRIPTION

The `poolcfg` command provides configuration operations on pools and sets. These operations are performed upon an existing configuration and take the form of modifications to the specified configuration file. The special `discover` command does not require an existing configuration. Actual activation of the resulting configuration is achieved by way of the `pooladm`(1M) command.

Pools configuration files are structured files that must have been constructed using `poolcfg` itself or `libpool`(3LIB) directly.

The configurations which are created by this tool may be used by `pooladm` to instantiate the configuration upon a target host.

OPTIONS

The following options are supported:

`-c` *command*	Specify *command* as an editing command. See `USAGE`.
`-f` *command_file*	Take the commands from *command_file*. *command_file* consists of editing commands, one per line.
`-h`	Display extended information about the syntax of editing commands.

Scripts

A script consists of editing commands, one per line, of the following:

info [*entity-name*]	Display configuration (or specified portion) in human readable form to standard output. If no entity is specified, system information is displayed. Therefore, `poolcfg -c 'info' afile` is an equivalent invocation to `poolcfg -c 'info system name' afile`.
create *entity-name* [*property-list*]	Make an entity of the specified type and name.
destroy *entity-name*	Remove the specified entity.
modify *entity-name* [*property-list*]	Change the listed properties on the named entity.
associate *pool-name* [*resource-list*]	Connect one or more resources to a pool, or replace one or more existing connections.
discover	Create a system entity, with one pool entity and resources to match current system configuration. All discovered resources of each resource type are recorded in the file, with the single pool referring to the default resource for each resource type.

rename *entity-name* to *new-name* — Change the name of an entity on the system to its new name.

Property Lists

The property list is specified by:

```
( proptype name = value [ ; proptype name = value ]* )
```

where the last definition in the sequence for a given proptype, name pair is the one that holds. For property deletion, use ~ `proptype name`.

Resource Lists

A resource list is specified by

```
( resourcetype name [ ; resourcetype name ]* )
```

where the last specification in the sequence for a resource is the one that holds. There is no deletion syntax for resource lists.

Recognized Entities

system	Machine level entity
pool	Named collection of resource associations

Resource Types

pset	Processor set resource

Property Types

boolean	Takes one of two values true or false.
int	A 64–bit signed integer value.
uint	A 64–bit unsigned integer value.
string	Strings are delimited by quotes ("), and support the character escape sequences defined in `formats`(5).
float	Scientific notation is not supported.

EXAMPLES

EXAMPLE 1 Creating an Initial Configuration File

The following command creates an initial configuration file for this host. By not supplying a file name, `/etc/pooladm.conf` is assumed.

```
$ poolcfg -c discover
```

EXAMPLE 2 Creating an Initial Configuration File

The following command creates an initial configuration file for this host and writes it to `/home/admin/new_config`:

```
$ poolcfg -c discover /home/admin/new_config
```

EXAMPLE 3 Writing a poolcfg Script

The following `poolcfg` script creates a pool named `Accounting`, and a processor set, `small-1`. The processor set is created first, then the pool is created and associated with the set.

EXAMPLE 3 Writing a poolcfg Script *(Continued)*

```
create pset small-1 ( int pset.size = 4 )
create pool Accounting
associate pool Accounting ( pset small-1 )
```

EXAMPLE 4 Reporting on `pool_0`

The following command reports on `pool_0` to standard output in human readable form:

```
# poolcfg -c 'info pool pool_0' /etc/pooladm.conf
```

EXAMPLE 5 Destroying `pool_0` and Its Associations

The following command destroys `pool_0` and associations, but not the formerly associated resources:

```
# poolcfg -c 'destroy pool pool_0' /etc/pooladm.conf
```

EXAMPLE 6 Displaying the Current Configuration

The following command displays the current configuration:

```
$ poolcfg -c 'info' /etc/pooladm.conf
system muskoka
          int system.version 1
          boolean system.bind-default true
          string system.comment Discovered by libpool

          pool pool_default
               boolean pool.default true
               boolean pool.active true
               int pool.importance 5
               string pool.comment
               string.pool.scheduler FSS
               pset pset_default

          pset pset_default
               int pset.sys_id -1
               string pset.units population
               boolean pset.default true
               uint pset.max 4294967295
               uint pset.min 1
               string pset.comment
               boolean pset.escapable false
               uint pset.load 0
               uint pset.size 2

               cpu
                    int cpu.sys_id 0
                    string cpu.comment
```

EXAMPLE 6 Displaying the Current Configuration *(Continued)*

```
                cpu
                     int cpu.sys_id 2
                     string cpu.comment
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpool
Interface Stability Invocation Output	Evolving Unstable

SEE ALSO

`pooladm`(1M), `poolbind`(1M), `libpool`(3LIB), `attributes`(5), `formats`(5)

System Administration Guide: Resource Management and Network Services

NAME

ports – creates /dev entries and inittab entries for serial lines

SYNOPSIS

/usr/sbin/ports [-r *rootdir*]

DESCRIPTION

devfsadm(1M) is now the preferred command for /dev and /devices and should be used instead of ports.

The ports command creates symbolic links in the /dev/term and /dev/cua directories to the serial-port character device files in /devices and adds new entries in /etc/inittab for non-system ports found. System-board ports are given single lower-case letters for names (such as a and b) while other ports are named numerically.

ports searches the kernel device tree to find the serial devices attached to the system. It also checks /dev/term and /dev/cua to see what symbolic links to serial devices already exist. ports then performs the following:

1. Assigns new numbers (or letters for system-board ports) to ports that are attached to the system but do not have /dev/term and /dev/cua entries. The numbers or letters assigned are the lowest-unused numbers or letters.
2. Removes dangling links: links from /dev/term and /dev/cua pointing to no-longer-existing ports.
3. Creates new /dev/term and /dev/cua links for new serial devices.
4. Invokes sacadm(1M) to make new port monitor entries for the new devices. This is not done automatically for on-board ports; on workstations these ports are often not used for dial-in sessions, so a port-monitor for one of these ports must be created explicitly.

If the configuration has not changed, ports exits without doing anything.

ports is run each time a reconfiguration-boot is performed, or when add_drv(1M) is executed. When invoking ports manually, first run drvconfig(1M) to ensure /devices is consistent with the current device configuration.

Notice to Driver Writers

ports considers devices with a node type of DDI_NT_SERIAL, DDI_NT_SERIAL_MB, DDI_NT_SERIAL_DO, or DDI_NT_SERIAL_MB_DO to be serial port devices. Devices with one of these node types must create minor device names that obey the following conventions when calling ddi_create_minor_node(9F).

- The minor name for non-system port devices (DDI_NT_SERIAL) consists of an ASCII numeric string, where the first port on the device is named 0, the second named 1, the third named 2, up to the number of ports provided by the device.
- The minor name for non-system dialout devices (DDI_NT_SERIAL_DO) is the ASCII numeric port name, concatenated with ,cu. For example, the minor name for the first dialout port on the serial board is 0,cu.
- The minor name for system-board port devices (DDI_NT_SERIAL_MB) consists of a string containing a single ASCII lowercase character, where the first port on the device is named a, the second is named b, the third is named c, for all ports on the

device (or up through port `z`).

- The minor name for system-board dialout devices (`DDI_NT_SERIAL_MB_DO`) consists of the lowercase character port name, concatenated with `,cu`. For example, the minor name for the first dialout port on the on-board serial device is `a,cu`.

To prevent `disks` from attempting to automatically generate links for a device, drivers must specify a private node type and refrain from using one of the above node types when calling `ddi_create_minor_node`(9F).

OPTIONS

The following options are supported:

`-r` *rootdir* — Causes `ports` to presume that the `/dev/term`, `/dev/cua`, and `/devices` directories are found under *rootdir*, not directly under `/`. If this argument is specified, `sacadm`(1M) is not invoked, since it would update terminal administration files under `/etc` without regard to the *rootdir*.

EXAMPLES

EXAMPLE 1 Creating the Serial and Dialout Minor Device Nodes

The following example creates the serial and dialout minor device nodes from the `xkserial` driver's `attach`(9E) function:

```
/*
* Create the minor number by combining the instance number
* with the port number.
*/ #define XKNUMPORTS           8
#define XKMINORNUM(i, p)     ((i) << 4 | (p))
#define XKMINORNUM_DO(i, p)  ((i) << 4 | (p) | 0x80)
int
xkserialattach(dev_info_t *dip, ddi_attach_cmd_t cmd)
{
   int instance, portnum;
   char name[8];
       /* other stuff in attach... */
   instance = ddi_get_instance(dip);
   for (portnum = 0; portnum < XKNUMPORTS; portnum++) {
       /*
        * create the serial port device
        */
        sprintf(name, "%d", portnum);
        ddi_create_minor_node(dip, name, S_IFCHR,
           XKMINORNUM(instance, portnum), DDI_NT_SERIAL, 0);

        /*
         * create the dialout device
         */
         sprintf(name,"%d,cu", portnum);
         ddi_create_minor_node(dip, name, S_IFCHR,
             XKMINORNUM_DO(instance, portnum), DDI_NT_SERIAL_DO, 0);
     }
}
```

EXAMPLE 2 Installing the `xkserial` Port Driver on a SPARCstation 20

The following example installs the `xkserial` port driver on a SPARCstation 20 (with the driver controlling the fictional XKSerial 8 port serial board) and performs a reconfiguration-boot. It creates the following special files in `/devices`.

```
# ls -l /devices/iommu@f,e0000000/sbus@f,e0001000/xkserial@f,800000/
crw-r-----   1 root sys   32,  16 Aug 29 00:02 xkserial@2000:0
crw-r-----   1 root sys   32, 144 Aug 29 00:02 xkserial@2000:0,cu
crw-r-----   1 root sys   32,  17 Aug 29 00:02 xkserial@2000:1
crw-r-----   1 root sys   32, 145 Aug 29 00:02 xkserial@2000:1,cu
crw-r-----   1 root sys   32,  18 Aug 29 00:02 xkserial@2000:2
crw-r-----   1 root sys   32, 146 Aug 29 00:02 xkserial@2000:2,cu
crw-r-----   1 root sys   32,  19 Aug 29 00:02 xkserial@2000:3
crw-r-----   1 root sys   32, 147 Aug 29 00:02 xkserial@2000:3,cu
crw-r-----   1 root sys   32,  20 Aug 29 00:02 xkserial@2000:4
crw-r-----   1 root sys   32, 148 Aug 29 00:02 xkserial@2000:4,cu
crw-r-----   1 root sys   32,  21 Aug 29 00:02 xkserial@2000:5
crw-r-----   1 root sys   32, 149 Aug 29 00:02 xkserial@2000:5,cu
crw-r-----   1 root sys   32,  22 Aug 29 00:02 xkserial@2000:6
crw-r-----   1 root sys   32, 150 Aug 29 00:02 xkserial@2000:6,cu
crw-r-----   1 root sys   32,  23 Aug 29 00:02 xkserial@2000:7
crw-r-----   1 root sys   32, 151 Aug 29 00:02 xkserial@2000:7,cu
```

`/dev/term` contain symbolic links to the serial port device nodes in `/devices`

```
# ls -l /dev/term
/dev/term/0 -> ../../devices/[....]/xkserial@2000:0
/dev/term/1 -> ../../devices/[....]/xkserial@2000:1
/dev/term/2 -> ../../devices/[....]/xkserial@2000:2
/dev/term/3 -> ../../devices/[....]/xkserial@2000:3
/dev/term/4 -> ../../devices/[....]/xkserial@2000:4
/dev/term/5 -> ../../devices/[....]/xkserial@2000:5
/dev/term/6 -> ../../devices/[....]/xkserial@2000:6
/dev/term/7 -> ../../devices/[....]/xkserial@2000:7
```

and `/dev/cua` contain symbolic links to the dialout port device nodes in `/devices`

```
# ls -l /dev/cua

/dev/cua/0 -> ../../devices/[....]/xkserial@2000:0,cu
/dev/cua/1 -> ../../devices/[....]/xkserial@2000:1,cu
/dev/cua/2 -> ../../devices/[....]/xkserial@2000:2,cu
/dev/cua/3 -> ../../devices/[....]/xkserial@2000:3,cu
/dev/cua/4 -> ../../devices/[....]/xkserial@2000:4,cu
/dev/cua/5 -> ../../devices/[....]/xkserial@2000:5,cu
/dev/cua/6 -> ../../devices/[....]/xkserial@2000:6,cu
/dev/cua/7 -> ../../devices/[....]/xkserial@2000:7,cu
```

FILES

`/dev/term/`*n*	Logical serial port devices
`/dev/cua/`*n*	Logical dialout port devices

/etc/inittab

/etc/saf/*

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

add_drv(1M), devfsadm(1M), devlinks(1M), disks(1M), drvconfig(1M), pmadm(1M), sacadm(1M), tapes(1M), attributes(5), attach(9E), ddi_create_minor_node(9F)

Writing Device Drivers

NAME

powerd – Power manager daemon

SYNOPSIS

/usr/lib/power/powerd [-n]

DESCRIPTION

The powerd daemon is started by pmconfig(1M) to monitor system activity and perform an automatic shutdown using the suspend-resume feature. When the system is suspended, complete current state is saved on the disk before power is removed. On reboot, the system automatically starts a resume operation and the system is restored to the same state it was in immediately prior to suspend.

Immediately prior to system shutdown, the daemon notifies syslogd(1M) of the shutdown, which broadcasts a notification.

OPTIONS

The following option is supported:

-n No broadcast mode. The daemon silently shuts down the system without notifying syslogd(1M).

FILES

/etc/power.conf Power Management configuration information file

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpmu
Interface stability	Unstable

SEE ALSO

pmconfig(1M), dtpower(1M), syslogd(1M), power.conf(4), attributes(5), cpr(7), pm(7D)

Using Power Management

NAME

pppd – point to point protocol daemon

SYNOPSIS

pppd [*tty_name*] [*speed*] [*options*]

DESCRIPTION

The point-to-point protocol (PPP) provides a method for transmitting datagrams over serial point-to-point links. PPP is composed of three components: a facility for encapsulating datagrams over serial links, an extensible link control protocol (LCP), and a family of network control protocols (NCP) for establishing and configuring different network-layer protocols.

The encapsulation scheme is provided by driver code in the kernel. `pppd` provides the basic LCP authentication support and several NCPs for establishing and configuring the Internet Protocol (referred to as the IP Control Protocol or "IPCP") and IPv6 (IPV6CP).

OPTIONS

The following sections discuss the `pppd` options:

Options Files

Options are taken from files and the command line. `pppd` reads options from the files `/etc/ppp/options`, `$HOME/.ppprc` and `/etc/ppp/options.`*ttyname* (in that order) before processing the options on the command line. (Command-line options are scanned for the terminal name before the `options.`*ttyname* file is read.) To form the name of the `options.`*ttyname* file, the initial `/dev/` is removed from the terminal name, and any remaining forward slash characters (`/`) are replaced with dots. For example, with serial device `/dev/cua/a`, option file `/etc/ppp/options.cua.a` is read.

An options file is parsed into a series of words that are delimited by whitespace. Whitespace can be included in a word by enclosing the word in double-quotes ("). A backslash (\) quotes the succeeding character. A hash (#) starts a comment, which continues until the end of the line. There is no restriction on using the `file` or `call` options within an options file.

Frequently Used Options

`<tty_name>` — Communicate over the named device. The string `/dev/` is prepended if necessary. If no device name is given, or if the name of the terminal connected to the standard input is given, `pppd` uses that terminal and does not fork to put itself in the background. A value for this option from a privileged source cannot be overridden by a non-privileged user.

`<speed>` — Set the baud rate to `<speed>` (a decimal number). The default is to leave the baud rate unchanged. This option is normally needed for dial-out only.

`asyncmap` <*map*> — Set the `async` character map to <*map*>. The map describes which control characters cannot be successfully received over the serial line. `pppd` asks the peer to send these characters as a 2-byte escape sequence. The argument is a 32 bit hex number, with each bit representing a character to escape. Bit 0

	(00000001) represents the character 0x00; bit 31 (80000000) represents the character 0x1f or ^_. If multiple `asyncmap` options are given, the values are ORed together. If no `asyncmap` option is given, `pppd` attempts to negotiate a value of 0. If the peer agrees, this disables escaping of the standard control characters. Use the `default-asyncmap` option to disable negotiation and escape all control characters.
`auth`	Require the peer to authenticate itself before allowing network packets to be sent or received. This option is the default if the system has a default route. If the `auth` or the `noauth` option is not specified, `pppd` allows the peer to use only those IP addresses to which the system does not already have a route.
`call` *name*	Read options from the file `/etc/ppp/peers/`*name*. This file may contain privileged options, including `noauth`, even if `pppd` is not being run by root. The *name* string may not begin with a slash (“/”) or include consecutive periods `("..")` as a pathname component.
`callback` *number*	Request a callback to the given telephone number using Microsoft CBCP.
`connect` *script*	Use the executable or shell command specified by *script* to set up the serial line. This script would typically use the `chat`(1M) program to dial the modem and start the remote `PPP` session. A value for this option originating from a privileged source cannot be overridden by a non-privileged user.
`crtscts`	Use hardware flow control, that is, RTS/CTS, to control the flow of data on the serial port. If the `crtscts`, `nocrtscts`, `cdtrcts` or `nocdtrcts` option is not provided, the hardware flow control setting for the serial port is left unchanged. Some serial ports lack a true RTS output and use this mode to implement unidirectional flow control. The serial port suspends transmission when requested by the modem by means of CTS but cannot request the modem to stop sending to the computer. This mode allows the use of DTR as a modem control line.
`defaultroute`	Add a default route to the system routing tables when IPCP negotiation successfully completes, using the peer as the gateway. This entry is removed when the `PPP` connection is broken. This option is privileged if the `nodefaultroute` option is specified.

`disconnect` *script*
: Run the executable or shell command specified by *script* after `pppd` terminates the link. Typically, this script is used to command the modem to hang up if hardware modem control signals are not available. `disconnect` is not run if the modem has already hung up. A value for this option originating from a privileged source cannot be overridden by a non-privileged user.

`escape` *xx,yy,...*
: Specifies that certain characters be escaped on transmission regardless of whether the peer requests them to be escaped with its `async` control character map. The characters to be escaped are specified as a list of hex numbers separated by commas. Note that almost any character can be specified for the `escape` option, unlike the `asyncmap` option which allows only control characters to be specified. Characters that cannot be escaped are those containing hex values 0x20 through 0x3f and 0x5e.

`file` *name*
: Read options from file *name*. If this option is used on the command line or in `$HOME/.ppprc`, the file must be readable by the user invoking `pppd`. See Options Files for a list of files that `pppd` always reads, regardless of the use of this option.

`init` *script*
: Run the executable or shell command specified by *script* to initialize the serial line. This script would typically use the `chat`(1M) program to configure the modem to enable auto-answer. A value for this option from a privileged source cannot be overridden by a non-privileged user.

`lock`
: Directs `pppd` to create a UUCP-style lock file for the serial device to ensure exclusive access to the device.

`mru` *n*
: Set the Maximum Receive Unit (MRU) value to *n*. `pppd` asks the peer to send packets of no more than *n* bytes. Minimum MRU value is 128. Default MRU value is 1500. A value of 296 is recommended for slow links (40 bytes for TCP/IP header + 256 bytes of data). For IPv6, MRU must be at least 1280.

`mtu` *n*
: Set the Maximum Transmit Unit (MTU) value to *n*. Unless the peer requests a smaller value via MRU negotiation, `pppd` requests the kernel networking code to send data packets of no more than *n* bytes through the PPP network interface. For IPv6, MTU must be at least 1280.

`passive` Enables the "passive" option in the LCP. With this option, `pppd` attempts to initiate a connection; if no reply is received from the peer, `pppd` waits passively for a valid LCP packet instead of exiting, as it would without this option.

Options

`<local_IP_address>:<remote_IP_address>`
Set the local and/or remote interface IP addresses. Either one may be omitted, but the colon is required. The IP addresses are specified with a host name or in decimal dot notation, for example: `:10.1.2.3`. The default local address is the first IP address of the system unless the `noipdefault` option is provided. The remote address is obtained from the peer if not specified in any option. Thus, in simple cases, this option is not required. If a local and/or remote IP address is specified with this option, `pppd` will not accept a different value from the peer in the IPCP negotiation unless the `ipcp-accept-local` and/or `ipcp-accept-remote` options are given, respectively.

`allow-fcs` *fcs-type*
Set allowable FCS type(s) for data sent to the peer. The *fcs-type* is a comma-separated list of "crc16", "crc32", "null", or integers. By default, all known types are allowed. If this option is specified and the peer requests a type not listed, a LCP Configure-Nak is sent to request only the listed types.

`allow-ip` *address(es)*
Allow peers to use the given IP address or subnet without authenticating themselves. The parameter is parsed in the same manner as each element of the list of allowed IP addresses is parsed in the secrets files. See the Authentication section more more details.

`bsdcomp` *nr,nt*
Request that the peer compress packets that it sends using the BSD-Compress scheme, with a maximum code size of *nr* bits, and agree to compress packets sent to the peer with a maximum code size of *nt* bits. If *nt* is not specified, it defaults to the value given for *nr*. Values in the range 9 to 15 may be used for *nr* and *nt*; larger values provide better compression but consume more kernel memory for compression dictionaries. Alternatively, a value of 0 for *nr* or *nt* disables compression in the corresponding direction. Use `nobsdcomp` or `bsdcomp 0` to disable BSD-Compress compression entirely. If this option is read from a privileged source, a nonprivileged user may not specify a code size larger than the value from the privileged source.

`cdtrcts`
Use a non-standard hardware flow control such as DTR/CTS to control the flow of data on the serial port. If the `crtscts`, `nocrtscts`, `cdtrcts` or `nocdtrcts` option is not specified, the hardware flow control setting for the serial port is left unchanged. Some serial ports lack a true RTS output. Such serial ports use this mode to implement true bi-directional flow control. Note that this flow control mode does not permit using DTR as a modem control line.

`chap-interval` *n*
If this option is given, `pppd` will rechallenge the peer every *n* seconds.

`chap-max-challenge` *n*
Set the maximum number of CHAP challenge transmissions to *n* (default 10).

`chap-restart` *n*
Set the CHAP restart interval (retransmission timeout for challenges) to *n* seconds. The default is 3.

`connect-delay` *n*
Wait for up to *n* milliseconds after the connect script finishes for a valid PPP packet from the peer. When the wait period elapses or when a valid PPP packet is received from the peer, `pppd` begins negotiation by sending its first LCP packet. The default value is 1000 (1 second). A wait period applies only if the `connect` or `pty` option is used.

`datarate` *n*
Set maximum data rate to *n* (in bytes per second) when using the `pty`, `notty`, `record`, or `socket` options.

`debug`
Enables connection debugging facilities. If this option is given, `pppd` logs the contents of all control packets sent or received in a readable form. The packets are logged through syslog with facility `daemon` and level `debug`. This information can be directed to a file by configuring `/etc/syslog.conf` appropriately.

`default-asyncmap`
Disable `asyncmap` negotiation, forcing all control characters to be escaped for both the transmit and the receive direction.

`default-fcs`
Disable FCS Alternatives negotiation entirely. By default, no FCS Alternatives option is sent to the peer, but the option is accepted. If this option is specified by the peer, then LCP Configure-Reject is sent.

`default-mru`
Disable MRU [Maximum Receive Unit] negotiation. With this option, `pppd` uses the default MRU value of 1500 bytes for the transmit and receive directions.

`deflate` *nr,nt,e*
Request that the peer compress packets that it sends, using the `deflate` scheme, with a maximum window size of *2**nr* bytes, and agree to compress packets sent to the peer with a maximum window size of *2**nt* bytes and effort level of *e* (1 to 9). If *nt* is not specified, it defaults to the value given for *nr*. If *e* is not specified, it defaults to 6. Values in the range 9 to 15 may be used for *nr* and *nt*; larger values provide better compression but consume more kernel memory for compression dictionaries. (Value 8 is not permitted due to a zlib bug.) Alternatively, a value of 0 for *nr* or *nt* disables compression in the corresponding direction. Use `nodeflate` or `deflate 0` to disable `deflate` compression entirely. (Note: `pppd` requests deflate compression in preference to BSD-Compress if the peer can do either.) If this option is read from a privileged source, a nonprivileged user may not specify a code size larger than the value from the privileged source.

`demand`
Initiate the link only on demand, that is, when data traffic is present. With this option, the remote IP address must be specified by the user on the command line or in an options file. `pppd` initially configures and enables the interface for IP traffic without connecting to the peer. When traffic is available, `pppd` connects to the peer and performs negotiation, authentication and other actions. When completed, `pppd` passes data packets across the link. The `demand` option implies the `persist` option. If this behavior is not desired, use the `nopersist` option after the `demand` option. The `idle` and `holdoff` options can be used in conjunction with the `demand` option.

`domain` *d*
Append the domain name *d* to the local host name for authentication purposes. For example, if `gethostname()` returns the name `porsche`, but the fully qualified domain name is `porsche.Quotron.COM`, you could specify `domain Quotron.COM`. With this configuration, `pppd` uses the name `porsche.Quotron.COM` for accessing secrets in the secrets file and as the default name when authenticating to the peer. This option is privileged.

`endpoint` *endpoint-value*
Set the endpoint discriminator (normally used for RFC 1990 Multilink PPP operation). The *endpoint-value* consists of a class identifier and a class-dependent value. The class identifier is one of "null," "local," "IP," "MAC," "magic," "phone," or a decimal integer. If present, the class-dependent value is separated from the identifier by a colon (":") or period (".") . This value may be a standard dotted-decimal IP address for class "IP," an optionally colon-or-dot separated hex Ethernet address for class "MAC" (must have 6 numbers), or an arbitrary string of bytes specified in hex with optional colon or dot separators between bytes. Although this option is available, this implementation does not support multilink.

`fcs` *fcs-type*
Set FCS type(s) desired for data sent by the peer. The *fcs-type* is a comma-separated list of crc16, crc32, null, or integers. By default, an FCS Alternatives option is not specified, and the medium-dependent FCS type is used. If this option is specified and the peer sends an LCP Configure-Nak, only the listed types are used. If none are in common, the FCS Alternatives option is omitted from the next LCP Configure-Request to drop back to the default.

`hide-password`
When logging the contents of PAP packets, this option causes `pppd` to exclude the password string from the log. This is the default.

`holdoff` *n*
Specifies how many seconds to wait before re-initiating the link after it terminates. This option is effective only if the `persist` or `demand` option is used. The holdoff period is not applied if the link is terminated because it was idle.

`ident` *string*
Set the LCP Identification string. The default value is a version string similar to that displayed by the `--version` option.

`idle` *n*
Specifies that `pppd` must disconnect if the link is idle for *n* seconds. The link is idle when no data packets (i.e. IP packets) are being sent or received. Do not use this option with the `persist` option but without the `demand` option.

`ipcp-accept-local`
With this option, `pppd` accepts the peer's idea of the local IP address, even if the local IP address is specified in an option.

`ipcp-accept-remote`
With this option, `pppd` accepts the peer's idea of its remote IP address, even if the remote IP address is specified in an option.

`ipcp-max-configure` *n*
Set the maximum number of IPCP Configure-Request transmissions to *n* (default 10).

`ipcp-max-failure` *n*
Set the maximum number of IPCP Configure-NAKs sent before sending Configure-Rejects instead to *n* (default 10).

`ipcp-max-terminate` *n*
Set the maximum number of IPCP terminate-request transmissions to *n* (default 3).

`ipcp-restart` *n*
Set the IPCP restart interval (retransmission timeout) to *n* seconds (default 3).

`ipparam` *string*
Provides an extra parameter to the ip-up and ip-down scripts. When this option is given, the *string* supplied is given as the sixth parameter to those scripts. See the Scripts section.

`ipv6` <*local_interface_identifier*>,<*remote_interface_identifier*>
Set the local and/or remote 64-bit interface identifier. Either one may be omitted. The identifier must be specified in standard ASCII notation of IPv6 addresses (for example: `::dead:beef`). If the `ipv6cp-use-ipaddr` option is given, the local and remote identifiers are derived from the respective IPv4 addresses (see above). The `ipv6cp-use-persistent` option can be used instead of the `ipv6 <local>,<remote>` option.

`ipv6cp-accept-local`
Accept peer's interface identifier for the local link identifier.

`ipv6cp-max-configure` *n*
Set the maximum number of IPv6CP Configure-Request transmissions to *n* (default 10).

`ipv6cp-max-failure` *n*
Set the maximum number of IPv6CP Configure-NAKs sent before sending Configure-Rejects instead to *n* (default 10).

`ipv6cp-max-terminate` *n*
Set the maximum number of IPv6CP terminate-request transmissions to *n* (default 3).

`ipv6cp-restart` *n*
Set the IPv6CP restart interval (retransmission timeout) to *n* seconds (default 3).

`ipv6cp-use-ipaddr`
If either the local or remote IPv6 address is unspecified, use the corresponding configured IPv4 address as a default interface identifier. (This option uses the configured addresses, not the negotiated addresses. Do not use it with `ipcp-accept-local` if the local IPv6 identifier is unspecified or with `ipcp-accept-remote` if the remote IPv6 identifier is unspecified.)

`ipv6cp-use-persistent`
Use uniquely-available persistent value for link local address.

`kdebug` *n*
Enable debugging code in the kernel-level PPP driver. Argument *n* is the sum of the following values: `1` to enable general debug messages, `2` to request that contents of received packets be printed, and `4` to request contents of transmitted packets be printed. Messages printed by the kernel are logged by `syslogd`(1M) to a file directed in the `/etc/syslog.conf` configuration file. Do not use the `kdebug` option to debug failed links. Use the `debug` option instead.

`lcp-echo-failure` *n*
If this option is given, `pppd` presumes the peer to be dead if *n* LCP Echo-Requests are sent without receiving a valid LCP Echo-Reply. If this happens, `pppd` terminates the connection. This option requires a non-zero value for the `lcp-echo-interval` parameter. This option enables `pppd` to terminate after the physical connection is broken (for example, if the modem has hung up) in situations where no hardware modem control lines are available.

`lcp-echo-interval` *n*
If this option is given, `pppd` sends an LCP Echo-Request frame to the peer every *n* seconds. Normally the peer responds to the Echo-Request by sending an Echo-Reply. This option can be used with the `lcp-echo-failure` option to detect that the peer is no longer connected.

`lcp-max-configure` *n*
Set the maximum number of LCP Configure-Request transmissions to *n* (default 10).

`lcp-max-failure` *n*
Set the maximum number of LCP Configure-NAKs sent before starting to send Configure-Rejects instead to *n* (default 10).

`lcp-max-terminate` *n*
Set the maximum number of LCP Terminate-Request transmissions to *n* (default 3).

`lcp-restart` *n*
Set the LCP restart interval (retransmission timeout) to *n* seconds (default 3).

`linkname` *name*
Sets the logical name of the link to *name*. `pppd` creates a file named `ppp-`*name*`.pid` in `/var/run` containing its process ID. This is useful in determining which instance of `pppd` is responsible for the link to a given peer system. This is a privileged option.

`local`
Do not use modem control lines. With this option, `pppd` ignores the state of the CD (Carrier Detect) signal from the modem and does not change the state of the DTR (Data Terminal Ready) signal.

`logfd` *n*
Send log messages to file descriptor *n*. `pppd` sends log messages to (at most) one file or file descriptor (as well as sending the log messages to syslog), so this option and the `logfile` option are mutually exclusive. By default `pppd` sends log messages to `stdout` (file descriptor 1) unless the serial port is open on stdout.

`logfile` *filename*
Append log messages to the file *filename* (and send the log messages to syslog). The file is opened in append mode with the privileges of the user who invoked `pppd`.

`login`
Use the system password database for authenticating the peer using PAP, and record the user in the system `wtmp` file. Note that the peer must have an entry in the `/etc/ppp/pap-secrets` file and the system password database to be allowed access.

`maxconnect` *n*
Terminate the connection after it has been available for network traffic for *n* seconds (that is, *n* seconds after the first network control protocol starts). An LCP Time-Remaining message is sent when the first NCP starts, and again when 5, 2, and 0.5 minutes are remaining.

`maxfail` *n*
Terminate after *n* consecutive failed connection attempts. A value of 0 means no limit. The default value is 10.

`modem`
Use the modem control lines. This option is the default. With this option, `pppd` waits for the CD (Carrier Detect) signal from the modem to be asserted when opening the serial device (unless a connect script is specified), and drops the DTR (Data Terminal Ready) signal briefly when the connection is terminated and before executing the connect script.

`ms-dns` <*addr*>
If `pppd` is acting as a server for Microsoft Windows clients, this option allows `pppd` to supply one or two DNS (Domain Name Server) addresses to the clients. The first instance of this option specifies the primary DNS address; the second instance (if given) specifies the secondary DNS address. If the first instance specifies a name that resolves to multiple IP addresses, then the first two addresses are used. (This option is present in some older versions of `pppd` under the name `dns-addr`.)

`ms-lanman`
If `pppd` connects as a client to a Microsoft server and uses MS-CHAPv1 for authentication, this option selects the LAN Manager password style instead of Microsoft NT.

`ms-wins` <*addr*>
If `pppd` acts as a server for Microsoft Windows or Samba clients, this option allows `pppd` to supply one or two WINS (Windows Internet Name Services) server addresses to the clients. The first instance of this option specifies the primary WINS address; the second instance (if given) specifies the secondary WINS address. As with `ms-dns`, if the name specified resolves to multiple IP addresses, then the first two will be taken as primary and secondary.

`name` *name*
Set the name of the local system for authentication purposes to *name*. This is a privileged option. With this option, `pppd` uses lines in the secrets files that have *name* as the second field to look for a secret to use in authenticating the peer. In addition, unless overridden with the `user` option, *name* is used as the name to send to the peer when authenticating the local system. (Note that `pppd` does not append the domain name to *name*.)

`no-accm-test`
Disable use of `asyncmap` (ACCM) checking using LCP Echo-Request messages. If the `lcp-echo-failure` is used on an asynchronous line, `pppd` includes all control characters in the first *n* LCP Echo-Request messages. If the `asyncmap` is set incorrectly, the link drops rather than continue operation with random failures. This option disables that feature.

`noaccomp`
Disable HDLC Address/Control compression in both directions (send and receive).

`noauth`
Do not require the peer to authenticate itself. This option is privileged.

`nobsdcomp`
Disables BSD-Compress compression; `pppd` will not request or agree to compress packets using the BSD-Compress scheme. This option is not necessary if `noccp` is specified.

`noccp`
Disable CCP (Compression Control Protocol) negotiation. This option should only be required if the peer has bugs or becomes confused by requests from `pppd` for CCP negotiation. If CCP is disabled, then BSD and deflate compression do not need to be separately disabled.

`nocrtscts`
Disable hardware flow control (i.e. RTS/CTS) on the serial port. If the `crtscts`, `nocrtscts`, `cdtrcts` or `nocdtrcts` options are not given, the hardware flow control setting for the serial port is left unchanged.

`nocdtrcts`
This option is a synonym for `nocrtscts`. Either option will disable both forms of hardware flow control.

`nodefaultroute`
Disable the `defaultroute` option. You can prevent non-root users from creating default routes with `pppd` by placing this option in the `/etc/ppp/options` file.

`nodeflate`
Disables deflate compression; `pppd` will not request or agree to compress packets using the deflate scheme. This option is not necessary if `noccp` is specified.

`nodeflatedraft`
Do not use Internet Draft (incorrectly assigned) algorithm number for deflate compression. This option is not necessary if `noccp` is specified.

`nodetach`
Do not detach from the controlling terminal. Without this option, `pppd` forks to become a background process if a serial device other than the terminal on the standard input is specified.

`noendpoint`
Do not send or accept the Multilink Endpoint Discriminator option.

`noident`
Disable use of LCP Identification. LCP Identification messages will not be sent to the peer, but received messages will be logged. (Specify this option twice to completely disable LCP Identification. In this case, `pppd` sends LCP Code-Reject in response to received LCP Identification messages.)

`noip`
Disable IPCP negotiation and IP communication. Use this option only if the peer has bugs or becomes confused by requests from `pppd` for IPCP negotiation.

`noipv6`
Disable IPv6CP negotiation and IPv6 communication. IPv6 is not enabled by default.

`noipdefault`
Disables the default behavior when no local IP address is specified, which is to determine (if possible) the local IP address from the hostname. With this option, the peer must supply the local IP address during IPCP negotiation (unless it specified explicitly on the command line or in an options file).

`nolog`
Do not send log messages to a file or file descriptor. This option cancels the `logfd` and `logfile` options. `nologfd` acts as an alias for this option.

`nomagic`
Disable magic number negotiation. With this option, `pppd` cannot detect a looped-back line. Use this option only if the peer has bugs. Do not use this option to work around the “Serial line is looped back” error message.

`nopam`
This privileged option disables use of pluggable authentication modules. If this option is specified, `pppd` reverts to standard authentication mechanisms. The default is not to use PAM.

`nopcomp`
Disable protocol field compression negotiation in the receive and the transmit direction.

`nopersist`
Exit once a connection has been made and terminated. This is the default unless the `persist` or `demand` option is specified.

`noplink`
Cause `pppd` to use I_LINK instead of I_PLINK. This is the default. When I_LINK is used, the system cleans up terminated interfaces (even when SIGKILL is used) but does not allow `ifconfig`(1M) to unplumb PPP streams or insert or remove modules dynamically. Use the `plink` option if `ifconfig`(1M) modinsert, modremove or unplumb support is needed.

`nopredictor1`
Do not accept or agree to Predictor-1 compression. (This option is accepted for compatibility. The implementation does not support Predictor-1 compression.)

`noproxyarp`
Disable the `proxyarp` option. If you want to prevent users from creating proxy ARP entries with `pppd`, place this option in the `/etc/ppp/options` file.

`notty`
Normally, `pppd` requires a terminal device. With this option, `pppd` allocates itself a pseudo-tty master/slave pair and uses the slave as its terminal device. `pppd` creates a child process to act as a character shunt to transfer characters between the pseudo-tty master and its standard input and output. Thus, `pppd` transmits characters on its standard output and receives characters on its standard input even if they are not terminal devices. This option increases the latency and CPU overhead of transferring data over the ppp interface as all of the characters sent and received must flow through the character shunt process. An explicit device name may not be given if this option is used.

`novj`
Disable Van Jacobson style TCP/IP header compression in both the transmit and the receive direction.

`novjccomp`
Disable the connection-ID compression option in Van Jacobson style TCP/IP header compression. With this option, `pppd` does not omit the connection-ID byte from Van Jacobson compressed TCP/IP headers, nor does it ask the peer to do so. This option is unnecessary if `novj` is specified.

`pam`
This privileged option enables use of PAM. If this is specified, `pppd` uses the `pam`(3PAM) framework for user authentication with a service name of "ppp" if the `login` option and PAP authentication are used. The default is not to use PAM.

`papcrypt`
Indicates that `pppd` should not accept a password which, before encryption, is identical to the secret from the `/etc/ppp/pap-secrets` file. Use this option if the secrets in the `pap-secrets` file are in `crypt`(3C) format.

`pap-max-authreq` *n*
Set the maximum number of PAP authenticate-request transmissions to *n* (default 10).

`pap-restart` *n*
Set the PAP restart interval (retransmission timeout) to *n* seconds (default 3).

`pap-timeout` *n*
Set the maximum time that `pppd` waits for the peer to authenticate itself with PAP to *n* seconds (0= no limit). The default is 30 seconds.

`password` *string*
Password string for authentication to the peer.

`persist`
Do not exit after a connection is terminated; instead try to reopen the connection.

`plink`
Cause `pppd` to use I_PLINK instead of I_LINK. The default is to use I_LINK, which cleans up terminated interface (even if SIGKILL is used), but does not allow `ifconfig`(1M) to unplumb PPP streams or insert or remove modules dynamically. Use this option if `ifconfig`(1M) modinsert/modremove/unplumb support is needed. See also the `plumbed` option.

`plugin` *filename*
Load the shared library object file *filename* as a plugin. This is a privileged option. Unless the filename specifies an explicit path, `/etc/ppp/plugins` and `/usr/lib/inet/ppp` will be searched for the object to load in that order.

`plumbed`
This option indicates that `pppd` should find a plumbed interface and use that for the session. If IPv4 addresses or IPv6 interface IDs or link MTU are otherwise unspecified, they are copied from the interface selected. This mode mimics some of the functionality of the older `aspppd` implementation and may be helpful when `pppd` is used with external applications that use `ifconfig`(1M).

`pppmux` *timer*
Enable PPP Multiplexing option negotiation and set transmit multiplexing timeout to *timer* microseconds.

`privgroup` *group-name*
Allows members of group *group-name* to use privileged options. This is a privileged option. Because there is no guarantee that members of *group-name* cannot use `pppd`

to become root themselves, you should be careful using this option. Consider it equivalent to putting the members of *group-name* in the `root` or `sys` group.

`proxyarp`
Add an entry to the system's Address Resolution Protocol (ARP) table with the IP address of the peer and the Ethernet address of this system. When you use this option, the peer appears to other systems to be on the local Ethernet. The remote address on the PPP link must be in the same subnet as assigned to an Ethernet interface.

`pty` *script*
Specifies that the command *script*, and not a specific terminal device is used for serial communication. `pppd` allocates itself a pseudo-tty master/slave pair and uses the slave as its terminal device. *script* runs in a child process with the pseudo-tty master as its standard input and output. An explicit device name may not be given if this option is used. (Note: if the `record` option is used in conjunction with the `pty` option, the child process will have pipes on its standard input and output.)

`receive-all`
With this option, `pppd` accepts all control characters from the peer, including those marked in the receive `asyncmap`. Without this option, `pppd` discards those characters as specified in *RFC 1662*. This option should be used only if the peer has bugs, as is often found with dial-back implementations.

`record` *filename*
Directs `pppd` to record all characters sent and received to a file named *filename*. *filename* is opened in append mode, using the user's user-ID and permissions. Because this option uses a pseudo-tty and a process to transfer characters between the pseudo-tty and the real serial device, it increases the latency and CPU overhead of transferring data over the PPP interface. Characters are stored in a tagged format with timestamps that can be displayed in readable form using the `pppdump`(1M) program. This option is generally used when debugging the kernel portion of `pppd` (especially CCP compression algorithms) and not for debugging link configuration problems. See the `debug` option.

`remotename` *name*
Set the assumed name of the remote system for authentication purposes to *name*. Microsoft WindowsNT does not provide a system name in its CHAP Challenge messages, and this option is often used to work around this problem.

`refuse-chap`
With this option, `pppd` will not agree to authenticate itself to the peer using standard Challenge Handshake Authentication Protocol (CHAP). (MS-CHAP is not affected.)

`refuse-mschap`
Do not agree to authenticate to peer with MS-CHAPv1. If this option is specified, requests for MS-CHAPv1 authentication from the peer are declined with LCP Configure-Nak. That option does not disable any other form of CHAP.

`refuse-mschapv2`
Do not agree to authenticate to peer with MS-CHAPv2. If specified, this option requests that MS-CHAPv2 authentication from the peer be declined with LCP Configure-Nak. That option does not disable any other form of CHAP.

`refuse-pap`
With this option, `pppd` will not agree to authenticate itself to the peer using Password Authentication Protocol (PAP).

`require-chap`
Require the peer to authenticate itself using standard CHAP authentication. MS-CHAP is not affected.

`require-mschap`
Require the peer to authenticate itself using MS-CHAPv1 authentication.

`require-mschapv2`
Require the peer to authenticate itself using MS-CHAPv2 authentication.

`require-pap`
Require the peer to authenticate itself using PAP authentication.

`show-password`
When logging contents of PAP packets, this option causes `pppd` to show the password string in the log message.

`silent`
With this option, `pppd` will not transmit LCP packets to initiate a connection until a valid LCP packet is received from the peer. This is like the "passive" option with older versions of `pppd` and is retained for compatibility, but the current `passive` option is preferred.

`small-accm-test`
When checking the `asyncmap` (ACCM) setting, `pppd` uses all 256 possible values by default. See `no-accm-test`. This option restricts the test so that only the 32 values affected by standard ACCM negotiation are tested. This option is useful on very slow links.

`socket` *host*`:`*port*
Connect to given host and port using TCP and run PPP over this connection.

`sync`
Use synchronous HDLC serial encoding instead of asynchronous. The device used by `pppd` with this option must have sync support. Currently supports `zs`, `se`, and `hsi` drivers.

`unit` *n*
Set PPP interface unit number to *n*, if possible.

`updetach`
With this option, `pppd` detaches from its controlling terminal after establishing the PPP connection. When this is specified, messages sent to `stderr` by the connect script, usually `chat`(1M), and debugging messages from the debug option are directed to `pppd`'s standard output.

`usehostname`
Enforce the use of the hostname with domain name appended, if given, as the name of the local system for authentication purposes. This overrides the `name` option. Because the `name` option is privileged, this option is normally not needed.

`usepeerdns`
Ask the peer for up to two DNS server addresses. Addresses supplied by the peer, if any, are passed to the `/etc/ppp/ip-up` script in the environment variables DNS1 and DNS2. In addition, `pppd` creates an `/etc/ppp/resolv.conf` file containing one or two nameserver lines with the address(es) supplied by the peer.

`user` *name*
Sets the name used for authenticating the local system to the peer to *name*.

`vj-max-slots` *n*
Sets the number of connection slots to be used by the Van Jacobson TCP/IP header compression and decompression code to *n*, which must be between 2 and 16 (inclusive).

`welcome` *script*
Run the executable or shell command specified by *script* before initiating PPP negotiation, after the connect script, if any, has completed. A value for this option from a privileged source cannot be overridden by a non-privileged user.

`xonxoff`
Use software flow control, that is, XON/XOFF, to control the flow of data on the serial port.

Obsolete Options

The following options are obsolete:

`+ua` *name*
Read a PAP user name and password from the file *name*. This file must have two lines for name and password. Name and password are sent to the peer when the peer requests PAP authentication.

`+ipv6`
Enable IPv6 and IPv6CP without specifying interface identifiers.

`--version`
Show version number and exit.

`--help`
Show brief help message and exit.

EXTENDED DESCRIPTION

The following sections discuss miscellaneous features of `pppd`:

Security

`pppd` allows system administrators to provide legitimate users with PPP access to a server machine without fear of compromising the security of the server or the network it runs on. Access control is provided by restricting IP addresses the peer may use based on its authenticated identity (if any), and through restrictions on options a non-privileged user may use. Options that permit potentially insecure configurations are privileged. Privileged options are accepted only in files that are under the control of the system administrator or when `pppd` is being run by root.

By default, pppd allows an unauthenticated peer to use a given IP address only if the system does not already have a route to that IP address. For example, a system with a permanent connection to the wider Internet will normally have a default route, meaning all peers must authenticate themselves to set up a connection. On such a system, the auth option is the default. Conversely, a system with a PPP link that comprises the only connection to the Internet probably does not possess a default route, so the peer can use virtually any IP address without authenticating itself.

Security-sensitive options are privileged and cannot be accessed by a non-privileged user running pppd, either on the command line, in the user's $HOME/.pprc file, or in an options file read using the file option. Privileged options may be used in /etc/ppp/options file or in an options file read using the call option. If pppd is run by the root user, privileged options can be used without restriction. If the /etc/ppp/options file does not exist, then only root may invoke pppd. The /etc/ppp/options file must be created (but may be empty) to allow ordinary non-root users to access pppd.

When opening the device, pppd uses the invoking user's user ID or the root UID (that is, 0), depending if the device name was specified by the user or the system administrator. If the device name comes from a privileged source, that is, /etc/ppp/options or an options file read using the call option, pppd uses full root privileges when opening the device. Thus, by creating an appropriate file under /etc/ppp/peers, the system administrator can allow users to establish a PPP connection via a device that they would not normally have access to. Otherwise pppd uses the invoking user's real UID when opening the device.

Authentication

During the authentication process, one peer convinces the other of its identity by sending its name and some secret information to the other. During authentication, the first peer becomes the "client" and the second becomes the "server." Authentication names can (but are not required to) correspond to the peer's Internet hostnames.

pppd supports four authentication protocols: the Password Authentication Protocol (PAP) and three forms of the Challenge Handshake Authentication Protocol (CHAP). With the PAP protocol, the client sends its name and a cleartext password to the server to authenticate itself. With CHAP, the server initiates the authentication exchange by sending a challenge to the client who must respond with its name and a hash value derived from the shared secret and the challenge.

The PPP protocol is symmetrical, meaning that each peer may be required to authenticate itself to the other. Different authentication protocols and names can be used for each exchange.

By default, pppd authenticates if requested and does not require authentication from the peer. However, pppd does not authenticate itself with a specific protocol if it has no secrets that can do so.

pppd stores authentication secrets in the /etc/ppp/pap-secrets (for PAP), and /etc/ppp/chap-secrets (for CHAP) files. Both files use the same format. pppd uses secrets files to authenticate itself to other systems and to authenticate other systems to itself.

Secrets files contain one secret per line. Secrets are specific to a particular combination of client and server and can only be used by that client to authenticate itself to that server. Each line in a secrets file has a minimum of three fields that contain the client and server names followed by the secret. Often, these three fields are followed by IP addresses that are used by clients to connect to a server.

A secrets file is parsed into words, with client name, server name and secrets fields allocated one word each. Embedded spaces or other special characters within a word must be quoted or escaped. Case is significant in all three fields.

A secret beginning with an at sign (“@”) is followed by the name of a file containing the secret. An asterisk (*) as the client or server name matches any name. When choosing a match, pppd selects the one with the fewest wildcards. Succeeding words on a line are interpreted by pppd as acceptable IP addresses for that client. IP Addresses are disallowed if they appear in lines that contain only three words or lines whose first word begins with a hyphen (“-”). To allow any address, use "*". An address starting with an exclamation point (”!”) indicates that the specified address is not acceptable. An address may be followed by "/" and a number *n* to indicate a whole subnet (all addresses that have the same value in the most significant *n* bits). In this form, the address may be followed by a plus sign ("+") to indicate that one address from the subnet is authorized, based on the ppp network interface unit number in use. In this case, the host part of the address is set to the unit number, plus one.

When authenticating the peer, pppd chooses a secret with the peer’s name in the first field of the secrets file and the name of the local system in the second field. The local system name defaults to the hostname, with the domain name appended if the domain option is used. The default can be overridden with the name option unless the usehostname option is used.

When authenticating to the peer, pppd first determines the name it will use to identify itself to the peer. This name is specified with the user option. If the user option is not used, the name defaults to the host name of the local system. pppd then selects a secret from the secrets file by searching for an entry with a local name in the first field and the peer’s name in the second field. pppd will know the name of the peer if standard CHAP authentication is used because the peer will have sent it in the Challenge packet. However, if MS-CHAP or PAP is being used, pppd must determine the peer’s name from the options specified by the user. The user can specify the peer’s name directly with the remotename option. Otherwise, if the remote IP address was specified by a name, rather than in numeric form, that name will be used as the peer’s name. If that fails, pppd uses the null string as the peer’s name.

When authenticating the peer with PAP, the supplied password is compared with data in the secrets file. If the password and secret do not match, the password is encrypted using `crypt()` and checked against the secret again. If the `papcrypt` option is given, the first unencrypted comparison is omitted for better security, and entries must thus be in encrypted `crypt`(3C) form.

If the `login` option is specified, the username and password are also checked against the system password database. This allows you to set up the `pap-secrets` file to enable PPP access only to certain users, and to restrict the set of IP addresses available to users. Typically, when using the `login` option, the secret in `/etc/ppp/pap-secrets` would be "", which matches any password supplied by the peer. This makes having the same secret in two places unnecessary. When `login` is used, the `pam` option enables access control through `pam`(3PAM).

Authentication must be completed before IPCP (or other network protocol) can be started. If the peer is required to authenticate itself and fails, `pppd` closes LCP and terminates the link. If IPCP negotiates an unacceptable IP address for the remote host, IPCP is closed. IP packets are sent or received only when IPCP is open.

To allow hosts that cannot authenticate themselves to connect and use one of a restricted set of IP addresses, add a line to the `pap-secrets` file specifying the empty string for the client name and secret.

Additional `pppd` options for a given peer may be specified by placing them at the end of the secrets entry, separated by two dashes (—). For example

```
peername servername secret ip-address -- novj
```

Routing

When IPCP negotiation is complete, `pppd` informs the kernel of the local and remote IP addresses for the PPP interface and creates a host route to the remote end of the link that enables peers to exchange IP packets. Communication with other machines generally requires further modification to routing tables and/or Address Resolution Protocol (ARP) tables. In most cases the `defaultroute` and/or `proxyarp` options are sufficient for this, but further intervention may be necessary. If further intervention is required, use the `/etc/ppp/ip-up` script or a routing protocol daemon.

To add a default route through the remote host, use the `defaultroute` option. This option is typically used for “client” systems; that is, end-nodes that use the PPP link for access to the general Internet.

In some cases it is desirable to use proxy ARP, for example on a server machine connected to a LAN, to allow other hosts to communicate with the remote host. `proxyarp` instructs `pppd` to look for a network interface on the same subnet as the remote host. That is, an interface supporting broadcast and ARP that is not a point-to-point or loopback interface and that is currently up. If found, `pppd` creates a permanent, published ARP entry with the IP address of the remote host and the hardware address of the network interface.

When the `demand` option is used, the interface IP addresses are already set at the time when IPCP comes up. If `pppd` cannot negotiate the same addresses it used to configure the interface, it changes the interface IP addresses to the negotiated addresses. This may disrupt existing connections. Using demand dialing with peers that perform dynamic IP address assignment is not recommended.

Scripts

`pppd` invokes scripts at various stages during processing that are used to perform site-specific ancillary processing. These scripts may be shell scripts or executable programs. `pppd` does not wait for the scripts to finish. The scripts are executed as `root` (with the real and effective user-id set to 0), enabling them to update routing tables, run privileged daemons, or perform other tasks. Be sure that the contents of these scripts do not compromise your system's security. `pppd` runs the scripts with standard input, output and error redirected to `/dev/null`, and with an environment that is empty except for some environment variables that give information about the link. The `pppd` environment variables are:

`DEVICE`	Name of the serial tty device.
`IFNAME`	Name of the network interface.
`IPLOCAL`	IP address for the link's local end. This is set only when IPCP has started.
`IPREMOTE`	IP address for the link's remote end. This is set only when IPCP has started.
`PEERNAME`	Authenticated name of the peer. This is set only if the peer authenticates itself.
`SPEED`	Baud rate of the tty device.
`ORIG_UID`	Real user-id of user who invoked `pppd`.
`PPPLOGNAME`	Username of the real user-id who invoked `pppd`. This is always set.

`pppd` also sets the following variables for the ip-down and auth-down scripts:

`CONNECT_TIME`	Number of seconds between the start of PPP negotiation and connection termination.
`BYTES_SENT`	Number of bytes sent at the level of the serial port during the connection.
`BYTES_RCVD`	Number of bytes received at the level of the serial port during the connection.
`LINKNAME`	Logical name of the link, set with the `linkname` option.

If they exist, `pppd` invokes the following scripts. It is not an error if they do not exist.

`/etc/ppp/auth-up`	Program or script executed after the remote system successfully authenticates itself. It is executed with five command-line arguments: `interface-name`

	`peer-name user-name tty-device speed`. Note that this script is not executed if the peer does not authenticate itself, for example, when the `noauth` option is used.
`/etc/ppp/auth-down`	Program or script executed when the link goes down if `/etc/ppp/auth-up` was previously executed. It is executed in the same manner with the same parameters as `/etc/ppp/auth-up`.
`/etc/ppp/ip-up`	A program or script that is executed when the link is available for sending and receiving IP packets (that is, IPCP has come up). It is executed with six command-line arguments: `interface-name tty-device speed local-IP-address remote-IP-address ipparam`.
`/etc/ppp/ip-down`	A program or script which is executed when the link is no longer available for sending and receiving IP packets. This script can be used for undoing the effects of the `/etc/ppp/ip-up` script. It is invoked in the same manner and with the same parameters as the `ip-up` script.
`/etc/ppp/ipv6-up`	Similar to `/etc/ppp/ip-up`, except that it is executed when the link is available for sending and receiving IPv6 packets. Executed with six command-line arguments: `interface-name tty-device speed local-link-local-address remote-link-local-address ipparam`.
`/etc/ppp/ipv6-down`	Similar to `/etc/ppp/ip-down`, but executed when IPv6 packets can no longer be transmitted on the link. Executed with the same parameters as the ipv6-up script.

EXAMPLES

EXAMPLE 1 Using the `auth` Option

The following examples assume that the `/etc/ppp/options` file contains the `auth` option.

`pppd` is commonly used to dial out to an ISP. You can do this using the "`pppd call isp`" command where the `/etc/ppp/peers/isp` file is set up to contain a line similar to the following:

```
cua/a 19200 crtscts connect '/usr/bin/chat -f /etc/ppp/chat-isp' noauth
```

For this example, `chat`(1M) is used to dial the ISP's modem and process any login sequence required. The `/etc/ppp/chat-isp` file is used by `chat` and could contain the following:

EXAMPLE 1 Using the `auth` Option *(Continued)*

```
ABORT "NO CARRIER"
ABORT "NO DIALTONE"
ABORT "ERROR"
ABORT "NO ANSWER"
ABORT "BUSY"
ABORT "Username/Password Incorrect"
"" "at"
OK "at&f&d2&c1"
OK "atdt2468135"
"name:" "^Umyuserid"
"word:" "\qmypassword"
"ispts" "\q^Uppp"
"~-^Uppp-~"
```

See the `chat`(1M) man page for details of `chat` scripts.

EXAMPLE 2 Using `pppd` with `proxyarp`

`pppd` can also provide a dial-in ppp service for users. If the users already have login accounts, the simplest way to set up the ppp service is to let the users log in to their accounts and run `pppd` as shown in the following example:

```
example% pppd proxyarp
```

EXAMPLE 3 Providing a User with Access to PPP Facilities

To provide a user with access to the PPP facilities, allocate an IP address for the user's machine, create an entry in `/etc/ppp/pap-secrets` or `/etc/ppp/chap-secrets`. This enables the user's machine to authenticate itself. For example, to enable user "Joe" using machine "joespc" to dial in to machine "server" and use the IP address "joespc.my.net," add the following entry to the `/etc/ppp/pap-secrets` or `/etc/ppp/chap-secrets` files:

```
joespc    server    "joe's secret"    joespc.my.net
```

Alternatively, you can create another username, for example "ppp," whose login shell is `/usr/bin/pppd` and whose home directory is `/etc/ppp`. If you run `pppd` this way, add the options to the `/etc/ppp/.pppрc` file.

If your serial connection is complex, it may be useful to escape such control characters as XON (^Q) and XOFF (^S), using `asyncmap a0000`. If the path includes a telnet, escape ^] (`asyncmap 200a0000`). If the path includes a `rlogin` command, add `escape ff` option to the options, because `rlogin` removes the window-size-change sequence [0xff, 0xff, 0x73, 0x73, followed by any 8 bytes] from the stream.

EXIT STATUS

The `pppd` exit status indicates errors or specifies why a link was terminated. Exit status values are:

0	`pppd` has detached or the connection was successfully established and terminated at the peer's request.
1	An immediately fatal error occurred. For example, an essential system call failed.
2	An error was detected in the options given. For example, two mutually exclusive options were used, or `/etc/ppp/options` is missing and the user is not root.
3	`pppd` is not `setuid-root` and the invoking user is not root.
4	The kernel does not support PPP. For example, the PPP kernel driver is not included or cannot be loaded.
5	`pppd` terminated because it was sent a SIGINT, SIGTERM or SIGHUP signal.
6	The serial port could not be locked.
7	The serial port could not be opened.
8	The connect script failed and returned a non-zero exit status.
9	The command specified as the argument to the `pty` option could not be run.
10	The PPP negotiation failed because no network protocols were able to run.
11	The peer system failed or refused to authenticate itself.
12	The link was established successfully, but terminated because it was idle.
13	The link was established successfully, but terminated because the connect time limit was reached.
14	Callback was negotiated and an incoming call should arrive shortly.
15	The link was terminated because the peer is not responding to echo requests.
16	The link was terminated by the modem hanging up.
17	The PPP negotiation failed because serial loopback was detected.
18	The init script failed because a non-zero exit status was returned.
19	Authentication to the peer failed.

FILES

`/var/run/spppn.pid`	Process-ID for `pppd` process on PPP interface unit *n*.
`/var/run/ppp-name.pid`	Process-ID for `pppd` process for logical link name (see the `linkname` option).
`/etc/ppp/pap-secrets`	Usernames, passwords and IP addresses for PAP authentication. This file should be

	owned by root and not readable or writable by any other user, otherwise `pppd` will log a warning.
`/etc/ppp/chap-secrets`	Names, secrets and IP addresses for all forms of CHAP authentication. The `/etc/ppp/pap-secrets` file should be owned by `root` should not readable or writable by any other user, otherwise, `pppd` will log a warning.
`/etc/ppp/options`	System default options for `pppd`, read before user default options or command-line options.
`$HOME/.ppprc`	User default options, read before `/etc/ppp/options.`*ttyname*.
`/etc/ppp/options.`*ttyname*	System default options for the serial port in use; read after `$HOME/.ppprc`. The *ttyname* component of this filename is formed when the initial `/dev/` is stripped from the port name (if present), and slashes (if any) are converted to dots.
`/etc/ppp/peers`	Directory with options files that may contain privileged options, even if `pppd` was invoked by a user other than `root`. The system administrator can create options files in this directory to permit non-privileged users to dial out without requiring the peer to authenticate, but only to certain trusted peers.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpppdu
Interface Stability	Evolving

SEE ALSO

`chat`(1M), `ifconfig`(1M), `crypt`(3C), `pam`(3PAM), `attributes`(5)

Haskin, D., Allen, E. *RFC 2472 – IP Version 6 Over PPP*. Network Working Group. December 1998.

Jacobson, V. *RFC 1144, Compressing TCP/IP Headers for Low-Speed Serial Links*. Network Working Group. February, 1990

Lloyd, B., Simpson, W. *RFC 1334, PPP Authentication Protocols*. Network Working Group. October 1992.

McGregor, G. *RFC 1332, The PPP Internet Protocol Control Protocol (IPCP)*. Network Working Group. May 1992.

Rivest, R. *RFC 1321, The MD5 Message-Digest Algorithm*. Network Working Group. April 1992

Simpson, W. *RFC 1661, The Point-to-Point Protocol (PPP)*. Network Working Group. July 1994.

Simpson, W. *RFC 1662, HDLC-like Framing* . Network Working Group. July 1994.

NOTES

These signals affect `pppd` behavior:

`SIGINT, SIGTERM`	Terminate the link, restore the serial device settings and exit.
`SIGHUP`	Terminate the link, restore the serial device settings and close the serial device. If the `persist` or `demand` option is specified, `pppd` attempts to reopen the serial device and start another connection after the holdoff period. Otherwise `pppd` exits. If received during the holdoff period, `SIGHUP` causes `pppd` to end the holdoff period immediately.
`SIGUSR1`	Toggles the state of the `debug` option and prints link status information to the log.
`SIGUSR2`	Causes `pppd` to renegotiate compression. This is useful to re-enable compression after it has been disabled as a result of a fatal decompression error. (Fatal decompression errors generally indicate a bug in an implementation.)

DIAGNOSTICS

Messages are sent to the syslog daemon using facility `LOG_DAEMON`. To see error and debug messages, edit the `/etc/syslog.conf` file to direct the messages to the desired output device or file, or use the `updetach` or `logfile` options.

The `debug` option causes the contents of all LCP, PAP, CHAP or IPCP control packets sent or received to be logged. This is useful if PPP negotiation does not succeed or if authentication fails.

Debugging can also be enabled or disabled by sending a `SIGUSR1` signal, which acts as a toggle to the `pppd` process.

NAME

pppoec – PPPoE chat utility

SYNOPSIS

```
pppoec [-omillisecs] [-smillisecs] [-v] device [service [ [except]server...
     [only]]]

pppoec [-omillisecs] [-v] -i [device]
```

DESCRIPTION

The `pppoec` utility implements the client-side negotiation of PPPoE. It is intended to be used with the `pppd`(1M) `connect` option, in the same manner as the `chat`(1M) utility is used for asynchronous dial-up PPP.

When given with the `-i` flag, `pppoec` sends out a broadcast query on the given interface named by the *device* parameter. You can specify no other arguments in this mode. All responding PPPoE servers and the offered services are displayed on standard output.

Otherwise, when given without the `-i` flag, `pppoec` does the full PPPoE client-side negotiation. The *device* parameter is the intended Ethernet interface, and must already be plumbed with `sppptun`(1M). The optional *service* parameter specifies a particular service desired; other offered services will be ignored. The optional *server* parameter specifies a specific server desired. You can specify *server* as an Ethernet address in the usual x:x:x:x:x:x format (with "`*`" in any of the six byte positions interpreted to mean "any"), or as a symbolic name resolved through `/etc/ethers` (or NIS), or as a PPPoE access concentrator name. The sense of the match (true or false) can be inverted by specifying the keyword `except` before this string. This parameter can be specified more than once, and the first match is taken.

If you specify the *server* parameter, then the selected servers become "preferred." If no preferred server responds, then the first responding server is used instead. To exclude non-matching servers entirely, append the keyword `only`.

OPTIONS

`-i`	Sends out broadcast query over interface specified by *device*.
`-o`	Sets the initial wait time in milliseconds for PADO from the server before PADI is retried. The default is 500 milliseconds for normal operation, or 3000 milliseconds (3 seconds) for inquiry (`-i`) mode.
`-s`	Sets the initial wait time in milliseconds for PADS from the server before PADR is retried. The default is 2000 milliseconds (2 seconds).
`-v`	Displays verbose progress messages, including all PPPoE messages sent, and all state machine transitions.

You normally do not need to adjust the parameters set with `-o` and `-s`. They are provided for coping with unusually slow servers.

OPERANDS

The following operands are supported:

device	plumbed Ethernet interface

server	preferred server or, if you specify `only`, the specified server
service	desired service; other available services are ignored

EXAMPLES

EXAMPLE 1 Connecting to Any Service on `hme0`

The following command enables you to connect to any PPPoE service on `hme0`:

```
# /usr/bin/pppd sppptun plugin pppoe.so \
connect "/usr/lib/inet/pppoec hme0" debug
```

Often, a command such as the preceding is specified in an `/etc/ppp/peers` file instead. For example, enter the following in `/etc/ppp/peers/myisp`:

```
sppptun
plugin pppoe.so
connect "/usr/lib/inet/pppoec hme0"
debug
```

To invoke the PPP connection described in the file, enter:

```
% /usr/bin/pppd call myisp
```

Note that, because the `/etc/ppp/peers` files are considered privileged by `pppd`, you need not be root to invoke the preceding command.

EXAMPLE 2 Connecting to a Particular Service

A more complex example: on `hme0`, connect to only the `internet` service offered by PPPoE servers with access concentrator name `isp`, but not to any Ethernet addresses starting with `40:0:1a`.

```
# /usr/lib/inet/pppoec hme0 internet except 40:0:1a:*:*:* isp only
```

Note that the `except 40:0:1a:*:*:*` filter must come before `isp`, because the filters are first-match.

EXIT STATUS

The following exit values are returned:

`0`	Successful completion.
`>0`	An error occurred.

FILES

`/usr/lib/inet/pppoec`
executable command

`/dev/sppptun`
Solaris PPP tunneling device driver.

`/etc/ppp/connect-errors`
usual location of error output (see DIAGNOSTICS, below)

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpppdt

SEE ALSO

`pppd`(1M), `sppptun`(1M), `pppoed`(1M), `sppptun`(7M)

RFC 2516, Method for Transmitting PPP Over Ethernet (PPPoE), Mamakos et al, February 1999

DIAGNOSTICS

Error messages are written to standard error, which is normally redirected by `pppd` to `/etc/ppp/connect-errors`. The errors can also be redirected to `pppd`'s standard output by using the `updetach` option.

If you specify the `-v`, verbose progress messages are displayed, including all PPPoE messages sent, and all state machine transitions. Specifying the `updetach` or `nodetach` `pppd` option is helpful when using verbose mode.

NAME

pppoed – PPPoE server daemon

SYNOPSIS

ppoed [*options*]

DESCRIPTION

The pppoed daemon implements the server-side negotiation of PPPoE. When a client requests service from this daemon, a copy of pppd(1M) is invoked to handle the actual PPP communication.

At startup, options are read from the command line and the /etc/ppp/pppoe file. After these options have been read, options in the per-device /etc/ppp/pppoe.*device* files are read, using the device names specified on the command line or in /etc/ppp/pppoe. (Device names are not permitted in the per-device files.) It is not an error if any of these files are absent; missing files are ignored.

Options are reread in the same order on SIGHUP. Except for the possibility of short delays due to the processing time, SIGHUP does not interfere with any client operations. Current status, including options read, is dumped to /tmp/pppoed.*pid* on SIGINT.

The options are used to set up a list of services to be offered to PPPoE clients on the broadcast domains (Ethernet subnets) specified by the named devices. Option parsing is always in one of two modes, either global mode or service mode. The initial mode at the beginning of each file (and the command line) is global mode. Options specified in global mode serve as default values for subsequently defined services. Service mode is entered by the service *name* option. In this mode, the named option is defined. Options that appear in this mode override any global mode definitions for the current service.

The option parsing follows standard shell tokenizing rules, using whitespace to delimit tokens, quotes to enclose strings that can contain whitespace, and escape sequences for special characters. Environment variables are substituted using familiar $VAR and ${VAR} syntax and set using NEWVAR=*string*. Variables are both usable in subsequent options and provided to the pppd(1M) processes spawned for each client, but they are interpreted as they are encountered during option processing. Thus, all set variables are seen by all processes spawned; position in the configuration files has no effect on this.

OPTIONS

The pppoed daemon supports the following options:

client [except] *client-list*

This option restricts the clients that may receive the service. If the except keyword is given, then the clients on the list cannot access the service, but others can. If this keyword is not given, then only the listed clients can access the service.

This option can be specified more than once for a given service. For a given client, first match among all listed options encountered specifies the handling. If it

matches an option with `except` specified, then access is denied. Otherwise, it is granted. The `client` list within a service is prepended to any list specified in the global context.

If no `client` options are given or if all options are specified with `except`, then all clients are permitted by default. If any `client` options without `except` are specified, then no clients are permitted by default.

The *client-list* is a comma-separated list of client identifiers. The match is made if any client on the list matches; thus, these are logically "ORed" together. Each client identifier can be either a symbolic name (resolved through `/etc/ethers` or NIS, as defined by `/etc/nsswitch.conf`) or a hexadecimal Ethernet address in the format `x:x:x:x:x:x`. In the latter case, any byte of the address can be "`*`", which matches any value in that position. For example, `40:0:1a:*:*:*` matches Ethernet adapters from the manufacturer assigned block `40:0:1a`.

`debug` — Increase debug logging detail level by one. The detail levels are 0 (no logging), 1 (errors only; the default), 2 (warnings), 3 (informational messages), and 4 (debug messages). Log messages are written by default to `syslog`(3C) using facility *daemon* (see the `log` option below). When specified on the command line or in the global context of the `/etc/ppp/pppoe` file, this option also sets the daemon's default (non-service-related) detail level.

`device` *device-list* — Specify the devices on which the service is available. The *device-list* is a comma-separated list of logical device names (without the leading `/dev/`), such as `hme0`. This option is ignored if encountered in the per-device `/etc/ppp/pppoe.`*device* files.

`extra` *string* — Specifies extra options to `pppd`(1M). It defaults to "`plugin pppoe.so directtty`" and usually does not need to be overridden.

`file` *path* — Suspends parsing of the current file, returns to global mode, and reads options from *path*. This file must be present and readable; if it is not, an error is logged. When the end of that file is reached, processing returns to the current file and the mode is reset to global again.

The global mode options specified in files read by this command use the options set in the current file's global mode; this condition extends to any file included by those files. All files read are parsed as though the command line had specified this option, and thus inherit the command line's global modes.

This option can be used to revert to global mode at any point in an option file by specifying `file /dev/null`.

`group` *name*
: Specifies the group ID (symbolic or numeric) under which `pppd` is executed. If `pppoed` is not run as root, this option is ignored.

`log` *path*
: Specifies an alternate debug logging file. Debug messages are sent to this file instead of `syslog`. The special name `syslog` is recognized to switch logging back to `syslog`. When specified on the command line or in the global context of the `/etc/ppp/pppoe` file, this option also sets the daemon's default (non-service-related) log file.

`nodebug`
: Set debug logging detail level to 0 (no logging). When specified on the command line or in the global context of the `/etc/ppp/pppoe` file, this option also sets the daemon's default (non-service-related) detail level.

`nowildcard`
: Specifies that the current service should not be included in response to clients requesting "any" service. The client must ask for this service by name. When specified on the command line or in the global context of the `/etc/ppp/pppoe` file, this option causes `pppoed` to ignore all wildcard service requests.

`path` *path*
: Specifies the path to the `pppd` executable. Defaults to `/usr/bin/pppd`.

`pppd` *string*
: Passes command-line arguments to `pppd`. It can be used to set the IP addresses or configure security for the session. The default value is the empty string.

`server` *string*
: Specifies the PPPoE Access Concentrator name to be sent to the client. It defaults to "Solaris PPPoE".

`service` *name*
: Closes any service being defined and begins definition of a new service. The same service name can be used without conflict on multiple devices. If the same service name is used on a single device, then the last definition encountered during parsing overrides all previous definitions.

`user` *name*	Specifies the user ID (symbolic or numeric) under which `pppd` is executed. If `pppoed` is not run as root, this option is ignored.
`wildcard`	Specifies that the service should be included in responses to client queries that request "any" service (which is done by requesting a service name of length zero). When specified on the command line or in the global context of the `/etc/ppp/pppoe` file, this option causes `pppoed` to ignore all wildcard service requests. This is the default.

EXAMPLES

EXAMPLE 1 Configuring for Particular Services

In the `/etc/ppp/pppoe` file:

```
service internet
   device $DEV
   pppd "proxyarp 192.168.1.1:"
service debugging
   device hme0,$DEV
   pppd "debug proxyarp 192.168.1.1:"
```

You then invoke the daemon with:

```
% /usr/lib/inet/pppoed DEV=le0
```

The lines in `/etc/ppp/pppoe` and the preceding command result in offering services "internet" and "debugging" (and responding to wildcard queries) on interface `le0`, and offering only service "debugging" on interface `hme0`.

SIGNALS

The `pppoed` daemon responds to the following signals:

`SIGHUP`	Causes `pppoed` to reparse the original command line and all configuration files, and close and reopen any log files.
`SIGINT`	Causes a snapshot of the state of the `pppoed` daemon to be written to `/tmp/pppoed.`*pid* (where *pid* is the decimal process ID of the daemon).

FILES

`/usr/lib/inet/pppoed`	executable command
`/dev/sppptun`	Solaris PPP tunneling device driver
`/etc/ppp/pppoe`	main configuration option file
`/etc/ppp/pppoe.`*device*	per–device configuration option file
`/etc/ppp/pppoe-errors`	location of output from `pppd`'s stderr
`/etc/ppp/pppoe.if`	list of Ethernet interfaces to be plumbed at boot time
`/tmp/pppoed.`*pid*	ASCII text file containing dumped `pppoed` state information

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpppdt

SEE ALSO

`pppd`(1M), `pppoec`(1M), `sppptun`(1M), `sppptun`(7M)

RFC 2516, Method for Transmitting PPP Over Ethernet (PPPoE), Mamakos et al, February 1999

NOTES

Because `pppd` is installed setuid root, this daemon need not be run as root. However, if it is not run as root, the `user` and `group` options are ignored.

The Ethernet interfaces to be used must be plumbed for PPPoE using the `sppptun`(1M) utility before services can be offered.

The daemon operate runs even if there are no services to offer. If you want to modify a configuration, it is not necessary to terminate the daemon. Simply use `pkill -HUP pppoed` after updating the configuration files.

The PPPoE protocol is far from perfect. Because it runs directly over Ethernet, there is no possibility of security and the MTU is limited to 1492 (violating RFC 1661's default value of 1500). It is also not possible to run the client and the server of a given session on a single machine with a single Ethernet interface for testing purposes. The client and server portions of a single session must be run on separate Ethernet interfaces with different MAC addresses.

NAME

pppstats – print PPP statistics

SYNOPSIS

pppstats [-a] [-v] [-r] [-z] [-c <*count*>] [-w <*secs*>] [*interface*]

DESCRIPTION

The pppstats utility reports PPP-related statistics at regular intervals for the specified PPP interface. If the interface is unspecified, pppstats defaults to sppp0. The display is split horizontally into input and output sections containing columns of statistics describing the properties and volume of packets received and transmitted by the interface.

OPTIONS

The pppstats options are:

-a	Display absolute values rather than deltas. With this option, all reports show statistics for the time elapsed since the link was initiated. Without this option, the second and subsequent reports show statistics for the time since the last report.
-c *count*	Repeat the display *count* times. If this option is not specified, the default repeat count is 1 if the -w option is not specified, otherwise infinity.
-r	Display additional statistics summarizing the compression ratio achieved by the packet compression algorithm in use.
-v	Display additional statistics relating to the performance of the Van Jacobson TCP header compression algorithm.
-w *wait*	Pause *wait* seconds between each display. If this option is not specified, the default interval is five seconds.
-z	Instead of the standard display, show statistics indicating the performance of the packet compression algorithm in use.

EXTENDED DESCRIPTION

The following fields are printed on the input side when the -z option is not used:

IN	Total number of bytes received by this interface.
PACK	Total number of packets received by this interface.
VJCOMP	Number of header-compressed TCP packets received by this interface.
VJUNC	Number of header-uncompressed TCP packets received by this interface. Not reported when the -r option is specified.
VJERR	Number of corrupted or bogus header-compressed TCP packets received by this interface. Not reported when the -r option is specified.
VJTOSS	Number of VJ header-compressed TCP packets dropped on reception by this interface because of preceding errors. Only reported when the -v option is specified.

NON-VJ — Total number of non-TCP packets received by this interface. Only reported when the -v option is specified.

RATIO — Compression ratio achieved for received packets by the packet compression scheme in use, defined as the uncompressed size divided by the compressed size. Only reported when the -r option is specified.

UBYTE — Total number of bytes received, after decompression of compressed packets. Only reported when the -r option is specified.

The following fields are printed on the output side:

OUT — Total number of bytes transmitted from this interface.

PACK — Total number of packets transmitted from this interface.

VJCOMP — Number of TCP packets transmitted from this interface with VJ-compressed TCP headers.

VJUNC — Number of TCP packets transmitted from this interface with VJ-uncompressed TCP headers. Not reported when the -r option is specified.

NON-VJ — Total number of non-TCP packets transmitted from this interface. Not reported when the -r option is specified.

VJSRCH — Number of searches for the cached header entry for a VJ header compressed TCP packet. Only reported when the -v option is specified.

VJMISS — Number of failed searches for the cached header entry for a VJ header compressed TCP packet. Only reported when the -v option is specified.

RATIO — Compression ratio achieved for transmitted packets by the packet compression scheme in use, defined as the size before compression divided by the compressed size. Only reported when the -r option is specified.

UBYTE — Total number of bytes to be transmitted before packet compression is applied. Only reported when the -r option is specified.

When the -z option is specified, pppstats displays the following fields relating to the packet compression algorithm currently in use. If packet compression is not in use, these fields display zeroes. The fields displayed on the input side are:

COMPRESSED BYTE — Number of bytes of compressed packets received.

COMPRESSED PACK — Number of compressed packets received.

`INCOMPRESSIBLE BYTE`	Number of bytes of incompressible packets (that is, those which were transmitted in uncompressed form) received.
`INCOMPRESSIBLE PACK`	Number of incompressible packets received.
`COMP RATIO`	Recent compression ratio for incoming packets, defined as the uncompressed size divided by the compressed size (including both compressible and incompressible packets).

The fields displayed on the output side are:

`COMPRESSED BYTE`	Number of bytes of compressed packets transmitted.
`COMPRESSED PACK`	Number of compressed packets transmitted.
`INCOMPRESSIBLE BYTE`	Number of bytes of incompressible packets received; that is, those that were transmitted by the peer in uncompressed form.
`INCOMPRESSIBLE PACK`	Number of incompressible packets transmitted.
`COMP RATIO`	Recent compression ratio for outgoing packets.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpppdu
Interface Stability	Evolving

SEE ALSO

`pppd`(1M), `attributes`(5)

NAME

praudit – print contents of an audit trail file

SYNOPSIS

praudit [-lrs] [-d*del*] [*filename*...]

DESCRIPTION

praudit reads the listed *filename*s (or standard input, if no *filename* is specified) and interprets the data as audit trail records as defined in audit.log(4). By default, times, user and group IDs (UIDs and GIDs, respectively) are converted to their ASCII representation. Record type and event fields are converted to their ASCII representation. A maximum of 100 audit files can be specified on the command line.

OPTIONS

-l Prints one line per record. The record type and event fields are always converted to their short ASCII representation as is done for the -s option.

-r Print records in their raw form. Times, UIDs, GIDs, record types, and events are displayed as integers. This option and the -s option are exclusive. If both are used, a format usage error message is output.

-s Print records in their short form. All numeric fields are converted to ASCII and displayed. The short ASCII representations for the record type and event fields are used. This option and the -r option are exclusive. If both are used, a format usage error message is output.

-d*del* Use *del* as the field delimiter instead of the default delimiter, which is the comma. If *del* has special meaning for the shell, it must be quoted. The maximum size of a delimiter is four characters.

FILES

/etc/security/audit_event

/etc/security/audit_class

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

bsmconv(1M), audit(2), getauditflags(3BSM), audit.log(4), audit_class(4), audit_event(4), group(4), passwd(4), attributes(5)

NOTES

The functionality described in this man page is available only if the Basic Security Module (BSM) has been enabled. See bsmconv(1M) for more information.

NAME

printmgr – Solaris Print Manager is a graphical user interface for managing printers in a network

SYNOPSIS

`/usr/sadm/admin/bin/printmgr`

DESCRIPTION

Solaris Print Manager is a Java-based graphical user interface that enables you to manage local and remote printer access. This tool can be used in the following name service environments: LDAP, NIS, NIS+, NIS+ with Federated Naming Service (FNS), and files. You must be logged in as superuser to use this tool.

Using Solaris Printer Manager is the preferred method for managing printer access instead of `admintool:printers` because Solaris Print Manager centralizes printer information when it is used in a name service environment.

Adding printer information to a name service makes access to printers available to all systems on the network and generally makes printer administration easier because all the information about printers is centralized.

Solaris Print Manager may be run on a remote system with the display sent to the local system. See the*System Administration Guide: Advanced Administration* for instructions on setting the `DISPLAY` environment variable.

Using Solaris Print Manager to perform printer-related tasks automatically updates the appropriate printer databases. Solaris Print Manager also includes a command-line console that displays the `lp` command line for the add, modify, and delete printer operations. Errors and warnings may also be displayed when Printer Manager operations are performed.

Help is available by clicking the Help button.

USAGE

Solaris Print Manager enables you to do the following tasks:

Select a Name Service
: Select a name service for retrieving or changing printer information.

Add Access to a Printer
: Add printer access on a printer client using Solaris Print Manager.

Add an Attached Printer
: After physically attaching the printer to a system, use Solaris Print Manager to install a local printer and make it available for printing.

Add a Network Printer
: After physically attaching the printer to a system, use Solaris Print Manager to install a local printer and make it available for printing.

Modify Printer Properties
: After adding access to a printer or adding an attached or network printer, you can modify certain printer attributes.

Delete a Printer
: Delete access to a printer from the print client or delete a printer from the print server or from the name service environment.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWppm

SEE ALSO

`ldap`(1), `lpget`(1M), `lpset`(1M), `attributes`(5)

System Administration Guide: Advanced Administration

NOTES

Be mindful of the following if the LDAP database is used as the name service.

1. Because the domain information for the printer being updated is extracted from the `ldapclient`(1M) configuration, the LDAP server being updated must host the same domain that is used by the current `ldapclient`(1M) server.
2. If the LDAP server being updated is a replica LDAP server, the updates will be referred to the master LDAP server and completed there. The updates might be out of sync and not appear immediatedly, as the replica server may not have been updated by the master server. For example, a printer that you deleted by using `lpset` may still appear in the printer list you display with `lpget` until the replica is updated from the master. Replica servers vary as to how often they are updated from the master. See *System Administration Guide: Advanced Administration* for information on LDAP server replication.
3. Although users can use the LDAP command line utilities `ldapadd`(1) and `ldapmodify`(1)to update printer entries in the directory, the preferred method is to use `lpset`. Otherwise, if the `lpadd` and `lpmodify` utilities are used, the administrator must ensure that the `printer-name` attribute value is unique within the `ou=printers` container on the LDAP server. If the value is not unique, the result of modifications done using `lpset` or the Solaris Print Manager, `printmgr`(1M) may be unpredictable.

NAME

privatepw – administer FTP Server enhanced group access file

SYNOPSIS

privatepw [-c] [-f *ftpgroups*] [-g *real_group_name*] *accessgroup*

privatepw -d [-f *ftpgroups*] *accessgroup*

privatepw -l [-f *ftpgroups*]

privatepw -V

DESCRIPTION

The privatepw utility is an administrative tool to add, delete and list enhanced access group information in the ftpgroups file. See ftpgroups(4).

When privatepw is used without options, the help usage message is displayed. The privatepw utility prompts for a password when adding an enhanced access group entry or modifiying an existing one.

OPTIONS

The following options are supported by the privatepw utility:

-c	Create a new ftpgroups file for the specified *accessgroup*.
-d	Delete information about the specified *accessgroup* from the ftpgroups file.
-f *ftpgroups*	Use the specifed ftpgroups file for all updates.
-g *group*	Set the real system group to the *group* specified. *group* is a valid group name returned by getgrnam(3C). If the real system group is not supplied with the -g option when adding an ehanced access group entry, the privatepw utility prompts for it.
-l	List the contents of the ftpgroups file.
-V	Display program copyright and version information, then terminate.

OPERANDS

The following operands are supported:

accessgroup	The name of the enhanced access group to create or update It consists of an arbitrary string of alphanumeric and punctuation characters. See ftpgroups(4).

EXIT STATUS

The following exit values are returned:

0	Successful completion.
>0	An error occurred.

FILES

`/etc/ftpd/ftpgroups`

`/etc/group`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWftpu

SEE ALSO

`in.ftpd`(1M), `getgrnam`(3C), `ftpgroups`(4), `attributes`(5)

NAME

prodreg – GUI viewer for Solaris Product Registry

SYNOPSIS

`/usr/bin/prodreg` [*subcommand*]

DESCRIPTION

The `prodreg` command is the viewer for the Solaris Product Registry (ProdReg), a system for maintaining records of the software products installed on a given Solaris system.

OPERANDS

The following operands are supported:

subcommand A ProdReg subcommand name. The subcommand name is optional; if omitted, it is equivalent to the `swing` subcommand.

USAGE

The following subcommands are supported:

`swing` Starts up the viewer using the default Java 2, SDK, Standard Edition, Swing GUI. This is the same as `prodreg` with no arguments.

`awt` Starts up the viewer using a Java AWT GUI.

`help` Prints a usage message.

`version` Prints the current version of the `prodreg` program.

ENVIRONMENT VARIABLES

The following environment variable affects the operation of `prodreg`:

`PKG_INSTALL_ROOT` If present, defines the full path name of a directory to use as the system's `PKG_INSTALL_ROOT` path. All product and package information files are then looked for in the directory tree, starting with the specified `PKG_INSTALL_ROOT` path. If not present, the default system path of "/" is used.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWwsrv

SEE ALSO

`installer`(1M), `libwsreg`(3LIB), `attributes`(5)

NAME

projadd – administer a new project on the system

SYNOPSIS

projadd [-c *comment*] [-U *user* [,*user*...]] [-G *group* [,*group*...]] [-p *projid* [*projid*...]] *project*

DESCRIPTION

`projadd` adds a new project entry to the `/etc/project` file. If the files backend is being used for the project database, the new project is available for use immediately upon the completion of the `projadd` command.

OPTIONS

The following options are supported:

`-c` *comment*
: Add a project comment. Comments are stored in the project's entry in the `/etc/project` file. Generally, comments contain a short description of the project and are used as the field for the project's full name.

  Specify *comment* as a text string. *comment* cannot contain a colon (`:`) or NEWLINE.

`-G` *group*
: Make *group* a member group of project.

  Specify *group* as a group ID or name.

`-o`
: Enable duplication of a project ID.

`-p` *projid*
: Set the project ID of the new project.

  Specify *projid* as a non-negative decimal integer below `UID_MAX` as defined in `limits.h`. *projid* defaults to the next available unique number above the highest number currently assigned. For example, if *projid*s `100`, `105`, and `200` are assigned, the next default *projid* is `201`. *projid*s between `0-99` are reserved by SunOS.

`-U`
: Add the existing users (as specified either by user ID or name), to member users of project.

OPERANDS

The following operands are supported:

project
: The name of the project to create. The *project* operand is a string consisting of characters from the set of alphabetic characters, numeric characters, underline (_), and hyphen (-). The period ('.') is reserved for projects with special meaning to the operating system. The first character of the project name must be a letter. A warning message is displayed if these restrictions are not met.

EXIT STATUS

The following exit values are returned:

`0`
: Successful completion.

`2`
: The command syntax was invalid. A usage message for `projadd` is displayed.

`3`
: An invalid argument was provided to an option.

4	The *projid* given with the `-p` option is already in use.
5	The project files contain an error. See `project`(4).
6	The project to be modified, group, or user does not exist.
9	The project is already in use.
10	Cannot update the `/etc/project` file.

FILES

`/etc/project` System project file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWesu

SEE ALSO

`projects`(1), `groupadd`(1M), `groupdel`(1M), `groupmod`(1M), `grpchk`(1M), `projdel`(1M), `projmod`(1M), `useradd`(1M), `userdel`(1M), `usermod`(1M), `project`(4), `attributes`(5)

NOTES

In case of an error, `projadd` prints an error message and exits with a non-zero status.

`projadd` adds a project definition only on the local system. If a network name service such as NIS or LDAP is being used to supplement the local `/etc/project` file with additional entries, `projadd` cannot change information supplied by the network name service.

NAME

projdel – delete a project from the system

SYNOPSIS

`projdel` *project*

DESCRIPTION

The `projdel` utility deletes a project from the system and makes the appropriate changes to the system file.

OPERANDS

The following operands are supported:

project The name of the project to be deleted.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`2` The command syntax was invalid. A usage message for `projdel` is displayed.

`3` An invalid argument was provided to an option.

`4` The *projid* given with the `-p` option is already in use.

`5` The project files contain an error. See `project`(4).

`6` The project to be modified, group, or user does not exist.

`9` The project is already in use.

`10` Cannot update the `/etc/project` file.

FILES

`/etc/project` System project file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWesu

SEE ALSO

`projects`(1), `groupadd`(1M), `groupdel`(1M), `groupmod`(1M), `grpchk`(1M), `logins`(1M), `projadd`(1M), `projmod`(1M), `useradd`(1M), `userdel`(1M), `usermod`(1M), `project`(4), `attributes`(5)

DIAGNOSTICS

In case of an error, `projdel` prints an error message and exits with a non-zero status.

NOTES

`projdel` deletes a project definition only on the local system. If a network name service such as NIS or LDAP is being used to supplement the local `/etc/project` file with additional entries, `projdel` cannot change information supplied by the network name service.

NAME

projmod – modify a project's information on the system

SYNOPSIS

projmod [-p *projid* [-o] [-a | -r]] [-U *user* [,*user*...]] [-G *group* [,*group*...]] [-c *comment*] [-l *new_projectname*] *project*

DESCRIPTION

The `projmod` utility modifies a project's definition on the system. `projmod` changes the definition of the specified project and makes the appropriate project-related system file and file system changes.

OPTIONS

The following options are supported:

`-c` *comment*	Specify *comment* as a text string. Generally, *comment* contains a short description of the project. This information is stored in the project's `/etc/project` entry.
`-G` *group* [,*group*...]	Specify a replacement list of member groups of the project, or with `-a` or `-r`, a list of groups to be added or removed from the project.
`-l` *new_projectname*	Specify the new project name for the project. The *new_projectname* argument is a string consisting of characters from the set of alphabetic characters, numeric characters, period (`.`), underline (_), and hyphen (`-`). The first character should be alphabetic. A warning message is written if these restrictions are not met.
`-o`	This option allows the specified project ID to be duplicated (non-unique).
`-a`	This option specifies that the users or groups specified by the `-U` or `-G` flags should be added to the project, rather than replacing the existing member list.
`-r`	This option specifies that the users or groups specified by the `-U` or `-G` flags should be removed from the project, rather than replacing the existing member list.
`-p` *projid*	Specify a new project ID for the project. It must be a non-negative decimal integer less than MAXUID as defined in `param.h`.
`-U` *user* [,*user*...]	Specify a replacement list of member users of the project, or with `-a` or `-r`, a list of users to be added or removed from the project.

OPERANDS

The following operands are supported:

project	An existing project name to be modified.

EXIT STATUS

In case of an error, `projmod` prints an error message and exits with one of the following values:

The following exit values are returned:

0 Successful completion.

2 The command syntax was invalid. A usage message for `projmod` is displayed.

3 An invalid argument was provided to an option.

4 The *projid* given with the `-p` option is already in use.

5 The project files contain an error. See `project`(4).

6 The project to be modified, group, or user does not exist.

9 The project is already in use.

10 Cannot update the `/etc/project` file.

FILES

`/etc/group`	System file containing group definitions
`/etc/project`	System project file
`/etc/passwd`	System password file
`/etc/shadow`	System file containing users' encrypted passwords and related information

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWesu

SEE ALSO

`groupadd`(1M), `groupdel`(1M), `groupmod`(1M), `projadd`(1M), `projdel`(1M), `useradd`(1M), `userdel`(1M), `usermod`(1M), `passwd`(4), `project`(4), `attributes`(5)

NOTES

The `projmod` utility modifies project definitions only in the local `/etc/project` file. If a network name service such as NIS or LDAP is being used to supplement the local files with additional entries, *projmod* cannot change information supplied by the network name service. However `projmod` verifies the uniqueness of project name and project ID against the external name service.

NAME

prstat – report active process statistics

SYNOPSIS

```
prstat [-acJLmRtTv] [-C psrsetlist] [-j projlist] [-k tasklist] [-n
     ntop[,nbottom]] [-p pidlist] [-P cpulist] [-s key | -S key ] [-u euidlist]
     [-U uidlist] [interval [count]]
```

DESCRIPTION

The `prstat` utility iteratively examines all active processes on the system and reports statistics based on the selected output mode and sort order. `prstat` provides options to examine only processes matching specified PIDs, UIDs, CPU IDs, and processor set IDs.

The `-j`, `-k`, `-C`, `-p`, `-P`, `-u`, and `-U` options accept lists as arguments. Items in a list can be either separated by commas or enclosed in quotes and separated by commas or spaces.

If you do not specify an option, `prstat` examines all processes and reports statistics sorted by CPU usage.

OPTIONS

The following options are supported:

`-a` — Report information about processes and users. In this mode `prstat` displays separate reports about processes and users at the same time.

`-c` — Print new reports below previous reports instead of overprinting them.

`-C` *psrsetlist* — Report only processes or lwps that are bound to processor sets in the given list. Each processor set is identified by an integer as reported by `psrset`(1M). The load averages displayed are the sum of the load averages of the specified processor sets (see `pset_getloadavg`(3C)).

`-j` *projlist* — Report only processes or lwps whose project ID is in the given list. Each project ID can be specified as either a project name or a numerical project ID. See `project`(4).

`-J` — Report information about processes and projects. In this mode `prstat` displays separate reports about processes and projects at the same time.

`-k` *tasklist* — Report only processes or lwps whose task ID is in *tasklist*.

`-L` — Report statistics for each light-weight process (LWP). By default, `prstat` reports only the number of LWPs for each process.

`-m` — Report microstate process accounting information. In addition to all fields listed in `-v` mode, this mode also

includes the percentage of time the process has spent processing system traps, text page faults, data page faults, waiting for user locks and waiting for CPU (latency time).

`-n` *ntop*[,*nbottom*]
: Restrict number of output lines. The *ntop* argument determines how many lines of process or `lwp` statistics are reported, and the *nbottom* argument determines how many lines of user, task, or projects statistics are reported if the `-a`, `-t`, `-T`, or `-J` options are specified. By default, `prstat` can display as many lines of output as will fit within a window or terminal.

`-p` *pidlist*
: Report only processes whose process ID is in the given list.

`-P` *cpulist*
: Report only processes or lwps which have most recently executed on a CPU in the given list. Each CPU is identified by an integer as reported by `psrinfo`(1M).

`-R`
: Put `prstat` in the real time scheduling class. When this option is used, `prstat` is given priority over time-sharing and interactive processes. This option is available only for superuser.

`-s` *key*
: Sort output lines (that is, processes, `lwps`, or users) by *key* in descending order. Only one *key* can be used as an argument.

  There are five possible key values:

  cpu
  : Sort by process CPU usage. This is the default.

  pri
  : Sort by process priority.

  rss
  : Sort by resident set size.

  size
  : Sort by size of process image.

  time
  : Sort by process execution time.

`-S` *key*
: Sort output lines by *key* in ascending order. Possible *key* values are the same as for the `-s` option. See `-s`.

`-t`
: Report total usage summary for each user. The summary includes the total number of processes or LWPs owned by the user, total size of process images,

	total resident set size, total cpu time, and percentages of recent cpu time and system memory.
`-T`	Report information about processes and tasks. In this mode `prstat` displays separate reports about processes and tasks at the same time.
`-u` *euidlist*	Report only processes whose effective user ID is in the given list. Each user ID may be specified as either a login name or a numerical user ID.
`-U` *uidlis*t	Report only processes whose real user ID is in the given list. Each user ID may be specified as either a login name or a numerical user ID.
`-v`	Report verbose process usage. This output format includes the percentage of time the process has spent in user mode, in system mode, and sleeping. It also includes the number of voluntary and involuntary context switches, system calls and the number of signals received.

OUTPUT

The following list defines the column headings and the meanings of a `prstat` report:

PID
: The process ID of the process.

USERNAME
: The real user (login) name or real user ID.

SIZE
: The total virtual memory size of the process, including all mapped files and devices, in kilobytes (`K`), megabytes (`M`), or gigabytes (`G`). The resident set size of the process (RSS), in kilobytes (K), megabytes (M), or gigabytes (G).

STATE
: The state of the process:

  cpu*N*
  : Process is running on CPU *N*.

  sleep
  : Sleeping: process is waiting for an event to complete.

  run
  : Runnable: process in on run queue.

  zombie
  : Zombie state: process terminated and parent not waiting.

  stop
  : Process is stopped.

PRI
: The priority of the process. Larger numbers mean higher priority.

NICE
: Nice value used in priority computation. Only processes in certain scheduling classes have a nice value.

TIME	The cumulative execution time for the process.
CPU	The percentage of recent CPU time used by the process.
PROCESS	The name of the process (name of executed file).
LWPID	The `lwp` ID of the `lwp` being reported.
NLWP	The number of `lwp`s in the process.

The following columns are displayed when the `-v` or `-m` option is specified

USR	The percentage of time the process has spent in user mode.
SYS	The percentage of time the process has spent in system mode.
TRP	The percentage of time the process has spent in processing system traps.
TFL	The percentage of time the process has spent processing text page faults.
DFL	The percentage of time the process has spent processing data page faults.
LCK	The percentage of time the process has spent waiting for user locks.
SLP	The percentage of time the process has spent sleeping.
LAT	The percentage of time the process has spent waiting for CPU.
VCX	The number of voluntary context switches.
ICX	The number of involuntary context switches.
SCL	The number of system calls.
SIG	The number of signals received.

Under the `-l` option, one line is printed for each `lwp` in the process and some reporting fields show the values for the `lwp`, not the process.

OPERANDS

The following operands are supported:

count	Specifies the number of times that the statistics are repeated. By default, `prstat` reports statistics until a termination signal is received.
interval	Specifies the sampling interval in seconds; the default interval is `5` seconds.

EXAMPLES

EXAMPLE 1 Reporting the Five Most Active Super-User Processes

The following command reports the five most active super-user processes running on `CPU1` and `CPU2`:

```
example% prstat -u root -n 5 -P 1,2 1 1

PID   USERNAME  SIZE   RSS STATE  PRI  NICE      TIME  CPU PROCESS/LWP
 306   root     3024K 1448K sleep   58    0   0:00.00 0.3% sendmail/1
 102   root     1600K  592K sleep   59    0   0:00.00 0.1% in.rdisc/1
```

EXAMPLE 1 Reporting the Five Most Active Super-User Processes *(Continued)*

```
 250   root     1000K  552K sleep   58    0   0:00.00 0.0% utmpd/1
 288   root     1720K 1032K sleep   58    0   0:00.00 0.0% sac/1
   1   root      744K  168K sleep   58    0   0:00.00 0.0% init/1
TOTAL:       25, load averages:  0.05, 0.08, 0.12
```

EXAMPLE 2 Displaying Verbose Process Usage Information

The following command displays verbose process usage information about processes with lowest resident set sizes owned by users `root` and `john`.

```
example% prstat -S rss -n 5 -vc -u root,john

 PID USERNAME USR SYS TRP TFL DFL LCK SLP LAT VCX ICX SCL SIG PROCESS/LWP
   1 root     0.0 0.0  -   -   -   -  100  -    0   0   0   0 init/1
 102 root     0.0 0.0  -   -   -   -  100  -    0   0   3   0 in.rdisc/1
 250 root     0.0 0.0  -   -   -   -  100  -    0   0   0   0 utmpd/1
1185 john     0.0 0.0  -   -   -   -  100  -    0   0   0   0 csh/1
 240 root     0.0 0.0  -   -   -   -  100  -    0   0   0   0 powerd/4
 TOTAL:       71, load averages:  0.02, 0.04, 0.08
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`proc`(1), `psrinfo`(1M), `psrset`(1M), `sar`(1M), `pset_getloadavg`(3C), `proc`(4), `project`(4), `attributes`(5)

NOTES

The snapshot of system usage displayed by `prstat` is true only for a split-second, and it may not be accurate by the time it is displayed. When the `-m` option is specified, `prstat` tries to turn on microstate accounting for each process; the original state is restored when `prstat` exits. See `proc`(4) for additional information about the microstate accounting facility.

NAME

prtconf – print system configuration

SPARC

/usr/sbin/prtconf [-V] | [-F] | [-x] | [-vpPD]

IA

/usr/sbin/prtconf [-V] | [-x] | [-vpPD]

DESCRIPTION

The `prtconf` command prints the system configuration information. The output includes the total amount of memory, and the configuration of system peripherals formatted as a device tree.

OPTIONS

The following options are supported:

`-D` For each system peripheral in the device tree, displays the name of the device driver used to manage the peripheral.

`-F` Returns the device path name of the console frame buffer, if one exists. If there is no frame buffer, `prtconf` returns a non-zero exit code. This flag must be used by itself. It returns only the name of the console, frame buffer device or a non-zero exit code. For example, if the console frame buffer on a SPARCstation 1 is `cgthree` in SBus slot #3, the command returns: `/sbus@1,f80000000/cgthree@3,0`. This option could be used to create a symlink for `/dev/fb` to the actual console device.

`-p` Displays information derived from the device tree provided by the firmware (PROM) on SPARC platforms or the booting system on IA platforms.The device tree information displayed using this option is a snapshot of the initial configuration and may not accurately reflect reconfiguration events that occur later.

`-P` Includes information about pseudo devices. By default, information regarding pseudo devices is omitted.

`-v` Specifies verbose mode.

`-V` Displays platform-dependent PROM (on SPARC platforms) or booting system (on IA platforms) version information. This flag must be used by itself. The output is a string. The format of the string is arbitrary and platform-dependent.

`-x` Reports if the firmware on this system is 64-bit ready. Some existing platforms may need a firmware upgrade in order to run the 64-bit kernel. If the operation is not applicable to this platform or the firmware is already 64-bit ready, it exits silently with a return code of zero. If the operation is applicable to this platform and the firmware is not 64-bit ready, it displays a descriptive message on the standard output and exits with a non-zero return code. The hardware platform documentation contains more information about the platforms that may need a firmware upgrade in order to run the 64-bit kernel.

This flag overrides all other flags and must be used by itself.

EXIT STATUS

The following exit values are returned:

`0`	No error occurred.
`non-zero`	With the `-F` option (SPARC only), a non-zero return value means that the output device is not a frame buffer. With the `-x` option, a non-zero return value means that the firmware is not 64–bit ready. In all other cases, a non-zero return value means that an error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWesu (32-bit) SUNWesxu (64-bit)

SEE ALSO

`modinfo`(1M), `sysdef`(1M), `attributes`(5)

Sun Hardware Platform Guide

SPARC Only

`openprom`(7D)

NOTES

The output of the `prtconf` command is highly dependent on the version of the PROM installed in the system. The output will be affected in potentially all circumstances.

The `driver not attached` message means that no driver is currently attached to that instance of the device. In general, drivers are loaded and installed (and attached to hardware instances) on demand, and when needed, and may be uninstalled and unloaded when the device is not in use.

NAME

prtdiag – display system diagnostic information

SYNOPSIS

/usr/platform/*platform-name***/sbin/prtdiag** [-v] [-l]

DESCRIPTION

prtdiag displays system configuration and diagnostic information on sun4u systems.

The diagnostic information lists any failed field replaceable units (FRUs) in the system.

The interface, output, and location in the directory hierarchy for prtdiag are uncommitted and subject to change in future releases.

platform-name is the name of the platform implementation and can be found using the -i option of uname(1).

Note – prtdiag does not display diagnostic information and environmental status when executed on the Sun Enterprise 10000 server. See the /var/opt/SUNWssp/adm/${SUNW_HOSTNAME}/messages file on the system service processor (SSP) to obtain such information for this server.

OPTIONS

The following options are supported:

-l Log output. If failures or errors exist in the system, output this information to syslogd(1M) only.

-v Verbose mode. Displays the time of the most recent AC Power failure, and the most recent hardware fatal error information, and (if applicable) environmental status. The hardware fatal error information is useful to repair and manufacturing for detailed diagnostics of FRUs.

EXIT STATUS

The following exit values are returned:

0 No failures or errors are detected in the system.

1 Failures or errors are detected in the system.

2 An internal prtdiag error occurred, for example, out of memory.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWkvm

SEE ALSO

uname(1), modinfo(1M), prtconf(1M), psrinfo(1M), sysdef(1M), syslogd(1M), attributes(5), openprom(7D)

NAME

prtfru – print FRUID-specific information about the FRUs on a system or domain

SYNOPSIS

prtfru [-h] | [-d] | [-clx]

prtfru [*container*]

DESCRIPTION

The prtfru utility is used to obtain FRUID data from the system or domain. Its output is that of a tree structure echoing the path in the FRU (Field-Replaceable Unit) tree to each container. When a container is found, the data from that container is printed in a tree-like structure as well.

prtfru without any arguments will print the FRU hierarchy and all of the FRUID container data. prtfru prints to stdout which may be redirected to a file.

OPTIONS

The following options are supported:

-c Prints *only* the containers and their data. This option does not print the FRU tree hierarchy.

-d Prints a DTD for the current registry to stdout.

-h Displays a help message and exits.

-l Prints *only* the FRU tree hierarchy. This option does not print the container data.

-x Prints in XML format with a system identifier (SYSTEM) of prtfrureg.dtd.

Options -c and -l can be used together to obtain a list of the containers.

OPERANDS

The folowing operand is supported:

container The name of a particular container in the FRU hierarchy, that is, either the name or path/name of a container as displayed in the -l option.

EXIT STATUS

The following exit values are returned:

0 All information was found and printed successfully.

>0 An error has occurred.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWfruid

SEE ALSO

attributes(5)

NAME

prtpicl – print PICL tree

SYNOPSIS

`prtpicl` `[-c` *picl_class*`] [-v]`

DESCRIPTION

The `prtpicl` command prints the PICL tree maintained by the PICL daemon. The output of `prtpicl` includes the name and PICL class of the nodes.

OPTIONS

The following options are supported:

`-c` *picl_class* — Print only the nodes of the named PICL class.

`-v` — Print in verbose mode. In verbose mode, `prtpicl` prints a list of properties and values for each node. Verbose mode is disabled by default.

EXIT STATUS

The following exit values are returned:

`0` — Successful completion.

`non-zero` — An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpiclu

SEE ALSO

`picld`(1M), `attributes`(5)

NAME

prtvtoc – report information about a disk geometry and partitioning

SYNOPSIS

prtvtoc [-fhs] [-t *vfstab*] [-m *mnttab*] *device*

DESCRIPTION

The prtvtoc command allows the contents of the VTOC (volume table of contents) to be viewed. The command can be used only by the super-user.

The *device* name can be the file name of a raw device in the form of /dev/rdsk/c?t?d?s2 or can be the file name of a block device in the form of /dev/dsk/c?t?d?s2.

OPTIONS

The following options are supported:

-f	Report on the disk free space, including the starting block address of the free space, number of blocks, and unused partitions.
-h	Omit the headers from the normal output.
-s	Omit all headers but the column header from the normal output.
-t *vfstab*	Use *vfstab* as the list of filesystem defaults, in place of /etc/vfstab.
-m *mnttab*	Use *mnttab* as the list of mounted filesystems, in place of /etc/mnttab.

EXAMPLES

EXAMPLE 1 The prtvtoc command.

The command line entry and system response shown below are for a 424-megabyte hard disk:

```
example# prtvtoc /dev/rdsk/c0t3d0s2
* /dev/rdsk/c0t3d0s2  partition map
*
* Dimension:
*     512 bytes/sector
*      80 sectors/track
*       9 tracks/cylinder
*     720 sectors/cylinder
*    2500 cylinders
*    1151 accessible cylinders
*
* Flags:
*   1: unmountable
*  10: read-only
*
```

The data in the Tag column above indicates the type of partition, as follows:

Name	*Number*
UNASSIGNED	0x00

EXAMPLE 1 The `prtvtoc` command. *(Continued)*

BOOT	0x01
ROOT	0x02
SWAP	0x03
USR	0x04
BACKUP	0x05
STAND	0x06
VAR	0x07
HOME	0x08
ALTSCTR	0x09
CACHE	0x0a

The data in the `Flags` column above indicates how the partition is to be mounted, as follows:

Name	*Number*
MOUNTABLE, READ AND WRITE	0x00
NOT MOUNTABLE	0x01
MOUNTABLE, READ ONLY	0x10

EXAMPLE 2 Output for the `-f` option.

The following example shows output for the `-f` option for the same disk as above.

```
example# prtvtoc -f /dev/rdsk/c0t3d0s2
FREE_START=0 FREE_SIZE=0 FREE_COUNT=0 FREE_PART=34
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`devinfo`(1M), `fmthard`(1M), `format`(1M), `mount`(1M), `attributes`(5)

WARNINGS

The `mount` command does not check the "not mountable" bit.

NAME

psradm – change processor operational status

SYNOPSIS

psradm -f | -i | -n [-v] *processor_id*...

psradm -a-f | -i | -n [-v]

DESCRIPTION

The psradm utility changes the operational status of processors. The legal states for the processor are on-line, off-line, and no-intr.

An on-line processor processes LWPs (lightweight processes) and may be interrupted by I/O devices in the system.

An off-line processor does not process any LWPs. Usually, an off-line processor is not interruptible by I/O devices in the system. On some processors or under certain conditions, it may not be possible to disable interrupts for an off-line processor. Thus, the actual effect of being off-line may vary from machine to machine.

A no-intr processor processes LWPs but is not interruptible by I/O devices.

A processor may not be taken off-line if there are LWPs that are bound to the processor. On some architectures, it might not be possible to take certain processors off-line if, for example, the system depends on some resource provided by the processor.

At least one processor in the system must be able to process LWPs. At least one processor must also be able to be interrupted. Since an off-line processor may be interruptible, it is possible to have an operational system with one processor no-intr and all other processors off-line but with one or more accepting interrupts.

If any of the specified processors are powered off, psradm may power on one or more processors.

Only superusers can use the psradm utility.

OPTIONS

The following options are supported:

-a Perform the action on all processors, or as many as possible.

-f Take the specified processors off-line.

-i Set the specified processors no-intr.

-n Bring the specified processors on-line.

-v Output a message giving the results of each attempted operation.

OPERANDS

The following operands are supported:

processor_id The processor ID of the processor to be set on-line or off-line or no-intr.

Specify *processor_id* as an individual processor number (for example, 3), multiple processor numbers separated by spaces (for example, 1 2 3), or a range of processor numbers (for example,

`1-4`). It is also possible to combine ranges and (individual or multiple) *processor_id*s (for example, `13 5 78 9`).

EXAMPLES

EXAMPLE 1 Examples of `psradm`.

The following example sets processors 2 and 3 `off-line`.

```
psradm -f 2 3
```

The following example sets processors 1 and 2 `no-intr`.

```
psradm -i 1 2
```

The following example sets all processors `on-line`.

```
psradm -a -n
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

FILES

`/etc/wtmpx` records logging processor status changes

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`psrinfo`(1M), `psrset`(1M), `p_online`(2), `attributes`(5)

DIAGNOSTICS

`psradm: processor 4: Invalid argument`
The specified processor does not exist in the configuration.

`psradm: processor 3: Device busy`
The specified processor could not be taken `off-line` because it either has LWPs bound to it, is the last `on-line` processor in the system, or is needed by the system because it provides some essential service.

`psradm: processor 3: Device busy`
The specified processor could not be set `no-intr` because it is the last interruptible processor in the system, or or it is the only processor in the system that can service interrupts needed by the system.

`psradm: processor 3: Device busy`
The specified processor is powered off, and it cannot be powered on because some platform-specific resource is unavailable.

`psradm: processor 0: Not owner`
The user does not have permission to change processor status.

`psradm: processor 2: Operation not supported`
The specified processor is powered off, and the platform does not support power on of individual processors.

NAME

psrinfo – displays information about processors

SYNOPSIS

psrinfo [-v] [*processor_id*...]

psrinfo -s *processor_id*

DESCRIPTION

psrinfo displays information about processors.

Without the *processor_id* operand, psrinfo displays one line for each configured processor, displaying whether it is on-line, non-interruptible (designated by no-intr), off-line, or powered off, and when that status last changed. Use the *processor_id* operand to display information about a specific processor. See OPERANDS.

OPTIONS

The following options are supported:

-s *processor_id* Silent mode. Displays 1 if the specified processor is fully on-line, and 0 if the specified processor is non-interruptible, off-line, or powered off.

Use silent mode when using psrinfo in shell scripts.

-v Verbose mode. Displays additional information about the specified processors, including: processor type, floating point unit type and clock speed. If any of this information cannot be determined, psrinfo displays unknown.

OPERANDS

The following operands are supported:

processor_id The processor ID of the processor about which information is to be displayed.

Specify *processor_id* as an individual processor number (for example, 3), multiple processor numbers separated by spaces (for example, 1 2 3), or a range of processor numbers (for example, 1-4). It is also possible to combine ranges and (individual or multiple) *processor_id*s (for example, 13 5 78 9).

EXAMPLES

EXAMPLE 1 Displaying Information About All Configured Processors in Verbose Mode

The following example displays information about all configured processors in verbose mode.

```
psrinfo -v
```

EXAMPLE 2 Determining If a Processor is On-line

The following example uses psrinfo in a shell script to determine if a processor is on-line.

```
if [ "`psrinfo -s 3 2> /dev/null`" –eq 1 ]
then
    echo "processor 3 is up"
fi
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`psradm`(1M), `p_online`(2), `processor_info`(2), `attributes`(5)

DIAGNOSTICS

`psrinfo: processor 9: Invalid argument`
The specified processor does not exist.

NAME

psrset – creation and management of processor sets

SYNOPSIS

psrset -a *processor_set_id processor_id...*

psrset -b *processor_set_id pid...*

psrset -c [*processor_id...*]

psrset -d *processor_set_id*

psrset -e *processor_set_id command* [*argument(s)*]

psrset -f *processor_set_id*

psrset [-i] [*processor_set_id...*]

psrset -n *processor_set_id*

psrset -p [*processor_id...*]

psrset -q [*pid...*]

psrset -r *processor_id...*

psrset -u *pid...*

DESCRIPTION

The `psrset` utility controls the management of processor sets. Processor sets allow the binding of processes to groups of processors, rather than just a single processor. There are two types of processor sets, those created by the user using the `psrset` command or the `pset_create`(2) system call, and those automatically created by the system. Processors assigned to user-created processor sets will run only LWPs that have been bound to that processor set, but system processor sets may run other LWPs as well.

System-created processor sets will not always exist on a given machine. When they exist, they will generally represent particular characteristics of the underlying machine, such as groups of processors that can communicate more quickly with each other than with other processors in the system. These processor sets cannot be modified or removed, but processes may be bound to them.

OPTIONS

The following options are supported:

`-a` Assigns the specified processors to the specified processor set.

Processor sets automatically created by the system cannot have processors assigned to them. However, processors belonging to system processor sets may be assigned to user-created processor sets. This option is restricted to use by the super-user.

`-b` Binds all the LWPs of the specified processes to the specified processor set.

LWPs bound to a processor set will be restricted to run only on the processors in that set unless they require resources available only on another processor. Processes may only be bound to non-empty processor sets, that is, processor sets that have had processors assigned to them.

Bindings are inherited, so new LWPs and processes created by a bound LWP will have the same binding. Binding an interactive shell to a processor, for example, binds all commands executed by the shell.

`-c` Creates a new processor set and displays the new processor set ID.

If a list of processors is given, it also attempts to assign those processors to the processor set. If this succeeds, the processors will be idle until LWPs are bound to the processor set. This option is restricted to use by the super-user.

Only a limited number of processor sets may be active (created and not destroyed) at a given time. This limit will always be greater than the number of processors in the system. If the `-c` option is used when the maximum number of processor sets is already active, the command will fail.

The following format will be used for the first line of output of the `-c` option when the `LC_MESSAGES` locale category specifies the "C" locale. In other locales, the strings `created`, `processor`, and `set` may be replaced with more appropriate strings corresponding to the locale.

`"created processor set %d\n"` *processor set ID*

`-d` Removes the specified processor set, releasing all processors and processes associated with it.

Processor sets automatically created by the system cannot be removed. This option is restricted to use by the super-user.

`-e` Executes a command (with optional arguments) in the specified processor set.

The command process and any child processes are executed only by processors in the processor set.

The super-user may execute a command in any active processor set. Other users may only execute commands in system processor sets.

`-f` Disables interrupts for all processors within the specified processor set.

See `psradm`(1M). If some processors in the set cannot have their interrupts disabled, the other processors will still have their interrupts disabled, and the command will report an error and return non-zero exit status. This option is restricted to use by the super-user.

`-i` Displays a list of processors assigned to each named processor set. If no argument is given, a list of all processor sets and the processors assigned to them is displayed. This is also the default operation if the `psrset` command is not given an option.

`-n` Enables interrupts for all processors within the specified processor set.

See `psradm`(1M). This option is restricted to use by the super-user.

`-p` Displays the processor set assignments for the specified list of processors. If no argument is given, the processor set assignments for all processors in the system is given.

`-q` Displays the processor set bindings of the specified processes. If a process is composed of multiple LWPs, which have different bindings, the bindings of only one of the bound LWPs will be shown. If no argument is given, the processor set bindings of all processes in the system is displayed.

`-r` Removes a list of processors from their current processor sets. Processors that are removed will return to either the system processor set to which they previously belonged, or to the general pool of processors if they did not belong to a system processor set. This option is restricted to use by the super-user.

Processors with LWPs bound to them using `pbind`(1M) cannot be assigned to or removed from processor sets.

`-u` Removes the processor set bindings from all the LWPs of the specified processes, allowing them to be executed on any on-line processor if they are not bound to individual processors through `pbind`.

The super-user can bind or unbind any process to any active processor set. Other users can only bind or unbind processes to system processor sets, and unbind only from processor sets that do not have the `PSET_NOESCAPE` attribute set. Furthermore, they can only bind or unbind processes for which they have permission to signal, that is, any process that has the same effective user ID as the user.

OPERANDS

The following operands are supported:

pid Specify *pid* as a process ID.

processor_id Specify *processor_id* as an individual processor number (for example, `3`), multiple processor numbers separated by spaces (for example, `1 2 3`), or a range of processor numbers (for example, `1-4`). It is also possible to combine ranges and (individual or multiple) *processor_id*s (for example, `13 5 78 9`).

processor_set_id Specify *processor_set_id* as a processor set ID.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

non-0 An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu
Stability Level	Stable

SEE ALSO

pbind(1M), psradm(1M), psrinfo(1M), processor_bind(2), processor_info(2), pset_bind(2), pset_create(2), pset_info(2), sysconf(3C), attributes(5)

DIAGNOSTICS

`psrset: cannot query pid 31: No such process`
The process specified did not exist or has exited.

`psrset: cannot bind pid 31: Not owner`
The user does not have permission to bind the process.

`psrset: cannot assign processor 4: Not owner`
The user does not have permission to assign the processor.

`psrset: cannot assign processor 8: Invalid argument`
The specified processor is not on-line, or the specified processor does not exist.

`psrset: cannot bind pid 67: Device busy`
An LWP in the specified process is bound to a processor and cannot be bound to a processor set that does not include that processor.

`psrset: cannot assign processor 7: Device busy`
The specified processor could not be added to the processor set. This may be due to bound LWPs on that processor, or because that processor cannot be combined in the same processor set with other processors in that set, or because the processor is the last one in its current processor set.

`psrset: cannot execute in processor set 8: Invalid argument`
The specified processor set does not exist.

`psrset: cannot create processor set: Not enough space`
The maximum number of processor sets allowed in the system is already active.

NAME

putdev – edits device table

SYNOPSIS

putdev -a *alias* [*attribute=value* [...]]

putdev -m *device attribute=value* [*attribute* = *value* [...]]

putdev -d *device* [*attribute* [...]]

DESCRIPTION

putdev adds a new device to the device table, modifies an existing device description or removes a device entry from the table. The first synopsis is used to add a device. The second synopsis is used to modify existing entries by adding or changing attributes. If a specified attribute is not defined, this option adds that attribute to the device definition. If a specified attribute is already defined, it modifies the attribute definition. The third synopsis is used to delete either an entire device entry or, if the attribute argument is used, to delete an attribute assignment for a device.

OPTIONS

The following options are supported:

-a — Add a device to the device table using the specified attributes. The device must be referenced by its alias.

-d — Remove a device from the device table, when executed without the *attributes* argument. Used with the *attribute* argument, it deletes the given attribute specification for *device* from the table.

-m — Modify a device entry in the device table. If an entry already exists, it adds any specified attributes that are not defined. It also modifies any attributes which already have a value with the value specified by this command.

OPERANDS

The following operands are supported:

alias — Designates the alias of the device to be added.

attribute — Designates a device attribute to be added, modified, or deleted. Can be any of the device attributes described under DEVICE ATTRIBUTES except alias. This prevents an accidental modification or deletion of a device's alias from the table.

device — Designates the pathname or alias of the device whose attribute is to be added, modified, or removed.

value — Designates the value to be assigned to a device's attribute.

DEVICE ATTRIBUTES

The following list shows the standard device attributes, used by applications such as ufsdump(1M) and ufsrestore(1M), which can be defined for a device. You are not limited to this list, you can define any attribute you like.

alias — The unique name by which a device is known. No two devices in the database may share the same alias name. The name is limited in length to 14 characters and should contain only alphanumeric characters and the following special characters if they are escaped with a backslash: underscore (_), dollar sign ($), hyphen (–), and period (.).

`bdevice`	The pathname to the block special device node associated with the device, if any. The associated major/minor combination should be unique within the database and should match that associated with the `cdevice` field, if any. (It is the administrator's responsibility to ensure that these `major/minor` numbers are unique in the database.)
`capacity`	The capacity of the device or of the typical volume, if removable.
`cdevice`	The pathname to the character special device node associated with the device, if any. The associated major/minor combination should be unique within the database and should match that associated with the `bdevice` field, if any. (It is the administrator's responsibility to ensure that these `major/minor` numbers are unique in the database.)
`cyl`	Used by the command specified in the `mkfscmd` attribute.
`desc`	A description of any instance of a volume associated with this device (such as floppy diskette).
`dpartlist`	The list of disk partitions associated with this device. Used only if `type=disk`. The list should contain device aliases, each of which must have `type=dpart`.
`dparttype`	The type of disk partition represented by this device. Used only if `type=dpart`. It should be either `fs` (for file system) or `dp` (for data partition).
`erasecmd`	The command string that, when executed, erases the device.
`fmtcmd`	The command string that, when executed, formats the device.
`fsname`	The file system name on the file system administered on this partition, as supplied to the `/usr/sbin/labelit` command. This attribute is specified only if `type=dpart` and `dparttype=fs`.
`gap`	Used by the command specified in the `mkfscmd` attribute.
`mkfscmd`	The command string that, when executed, places a file system on a previously formatted device.
`mountpt`	The default mount point to use for the device. Used only if the device is mountable. For disk partitions where `type=dpart` and `dparttype=fs`, this attribute should specify the location where the partition is normally mounted.
`nblocks`	The number of blocks in the file system administered on this partition. Used only if `type=dpart` and `dparttype=fs`.
`ninodes`	The number of inodes in the file system administered on this partition. Used only if `type=dpart` and `dparttype=fs`.

`norewind` The name of the character special device node that allows access to the serial device without rewinding when the device is closed.

`pathname` Defines the pathname to an i-node describing the device (used for non-block or character device pathnames, such as directories).

`type` A token that represents inherent qualities of the device. Standard types include: 9-track, ctape, disk, directory, diskette, dpart, and qtape.

`volname` The volume name on the file system administered on this partition, as supplied to the `/usr/sbin/labelit` command. Used only if `type=dpart` and `dparttype=fs`.

`volume` A text string used to describe any instance of a volume associated with this device. This attribute should not be defined for devices which are not removable.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` Command syntax was incorrect, an invalid option was used, or an internal error occurred.

`2` The device table could not be opened for reading, or a new device table could not be created.

`3` If executed with the `-a` option, indicates that an entry in the device table with the alias `alias` already exits. If executed with the `-m` or `-d` options, indicates that no entry exists for device *device*.

`4` Indicates that `-d` was requested and one or more of the specified attributes were not defined for the device.

FILES

`/etc/device.tab`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`devattr`(1M), `putdgrp`(1M), `ufsdump`(1M), `ufsrestore`(1M), `attributes`(5)

System Administration Guide: Basic Administration

NAME

putdgrp – edits device group table

SYNOPSIS

putdgrp [-d] *dgroup* [*device*...]

DESCRIPTION

putdgrp modifies the device group table. It performs two kinds of modification. It can modify the table by creating a new device group or removing a device group. It can also change group definitions by adding or removing a device from the group definition.

When the command is invoked with only a *dgroup* specification, the command adds the specified group name to the device group table if it does not already exist. If the -d option is also used with only the *dgroup* specification, the command deletes the group from the table.

When the command is invoked with both a *dgroup* and a *device* specification, it adds the given device name(s) to the group definition. When invoked with both arguments and the -d option, the command deletes the device name(s) from the group definition.

When the command is invoked with both a *dgroup* and a *device* specification and the device group does not exist, it creates the group and adds the specified devices to that new group.

OPTIONS

The following options are supported:

-d Delete the group or, if used with *device*, delete the device from a group definition.

OPERANDS

The following operands are supported:

dgroup Specify a device group name.

device Specify the pathname or alias of the device that is to be added to, or deleted from, the device group.

EXIT STATUS

The following exit values are returned:

0 Successful completion.

1 Command syntax was incorrect, an invalid option was used, or an internal error occurred.

2 Device group table could not be opened for reading or a new device group table could not be created.

3 If executed with the -d option, indicates that an entry in the device group table for the device group *dgroup* does not exist and so cannot be deleted. Otherwise, indicates that the device group *dgroup* already exists and cannot be added.

4 If executed with the -d option, indicates that the device group *dgroup* does not have as members one or more of the specified devices. Otherwise, indicates that the device group *dgroup* already has one or more of the specified devices as members.

EXAMPLES

EXAMPLE 1 Adding a new device group.

The following example adds a new device group:

```
example#  putdgrp floppies
```

EXAMPLE 2 Adding a device to a device group.

The following example adds a device to a device group:

```
example#  putdgrp floppies diskette2
```

EXAMPLE 3 Deleting a device group.

The following example deletes a device group:

```
example#  putdgrp -d floppies
```

EXAMPLE 4 Deleting a device from a device group.

The following example deletes a device from a device group:

```
example#  putdgrp -d floppies diskette2
```

FILES

`/etc/dgroup.tab`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`listdgrp`(1M), `putdev`(1M), `attributes`(5)

NAME

pwck, grpck – password/group file checkers

SYNOPSIS

/usr/sbin/pwck [*filename*]

/usr/sbin/grpck [*filename*]

DESCRIPTION

`pwck` scans the password file and notes any inconsistencies. The checks include validation of the number of fields, login name, user ID, group ID, and whether the login directory and the program-to-use-as-shell exist. The default password file is `/etc/passwd`.

`grpck` verifies all entries in the group file. This verification includes a check of the number of fields, group name, group ID, whether any login names belong to more than NGROUPS_MAX groups and that all login names appear in the password file. The default group file is `/etc/group`.

FILES

`/etc/group`

`/etc/passwd`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`getpwent`(3C), `group`(4), `passwd`(4), `attributes`(5)

DIAGNOSTICS

Group entries in `/etc/group` with no login names are flagged.

`Group file '`*filename*`' is empty`
The `/etc/passwd` or `/etc/group` file is an empty file.

`cannot open file` *filename*`: No such file or directory`
The `/etc/passwd` or `/etc/group` file does not exist.

NOTES

If no filename argument is given, `grpck` checks the local group file, `/etc/group`, and also makes sure that all login names encountered in the checked group file are known to the system `getpwent`(3C) routine. This means that the login names may be supplied by a network name service.

NAME

pwconv – installs and updates /etc/shadow with information from /etc/passwd

SYNOPSIS

pwconv

DESCRIPTION

The pwconv command creates and updates /etc/shadow with information from /etc/passwd.

pwconv relies on a special value of 'x' in the password field of /etc/passwd. This value of 'x' indicates that the password for the user is already in /etc/shadow and should not be modified.

If the /etc/shadow file does not exist, this command will create /etc/shadow with information from /etc/passwd. The command populates /etc/shadow with the user's login name, password, and password aging information. If password aging information does not exist in /etc/passwd for a given user, none will be added to /etc/shadow. However, the last changed information will always be updated.

If the /etc/shadow file does exist, the following tasks will be performed:

Entries that are in the /etc/passwd file and not in the /etc/shadow file will be added to the /etc/shadow file.

Entries that are in the /etc/shadow file and not in the /etc/passwd file will be removed from /etc/shadow.

Password attributes (for example, password and aging information) that exist in an /etc/passwd entry will be moved to the corresponding entry in /etc/shadow.

The pwconv command can only be used by the super-user.

FILES

/etc/opasswd

/etc/oshadow

/etc/passwd

/etc/shadow

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

passwd(1), passmgmt(1M), usermod(1M), passwd(4), attributes(5)

DIAGNOSTICS

pwconv exits with one of the following values:

0 SUCCESS.

1 Permission denied.

2	Invalid command syntax.
3	Unexpected failure. Conversion not done.
4	Unexpected failure. Password file(s) missing.
5	Password file(s) busy. Try again later.
6	Bad entry in `/etc/shadow` file.

NAME

quot – summarize file system ownership

SYNOPSIS

quot [-acfhnv] *filesystem*...

quot -a [-cfhnv]

DESCRIPTION

quot displays the number of blocks (1024 bytes) in the named *filesystem* (one or more) currently owned by each user. There is a limit of 2048 blocks. Files larger than this will be counted as a 2048 block file, but the total block count will be correct.

OPTIONS

The following options are supported:

-a Generate a report for all mounted file systems.

-c Display three columns giving a file size in blocks, the number of files of that size, and a cumulative total of blocks containing files of that size or a smaller size.

-f Display three columns giving, for each user, the number of blocks owned, the count of number of files, and the user name. This option is incompatible with the -c and -v options.

-h Estimate the number of blocks in the file. This does not account for files with holes in them.

-n Attach names to the list of files read from standard input. quot -n cannot be used alone, because it expects data from standard input. For example, the pipeline

```
ncheck myfilesystem | sort +0n | quot -n myfilesystem
```

will produce a list of all files and their owners. This option is incompatible with all other options.

-v In addition to the default output, display three columns containing the number of blocks not accessed in the last 30, 60, and 90 days.

OPERANDS

filesystem mount-point of the filesystem(s) being checked

USAGE

See largefile(5) for the description of the behavior of quot when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXIT STATUS

0 Successful operation.

32 Error condition (bad or missing argument, bad path, or other error).

FILES

/etc/mnttab Lists mounted file systems.

/etc/passwd Used to obtain user names

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO du(1), mnttab(4), passwd(4), attributes(5), largefile(5)

NOTES This command can only be used by the super-user.

NAME

quota – display a user's ufs file system disk quota and usage

SYNOPSIS

quota [-v] [*username*]

DESCRIPTION

quota displays users' ufs disk usage and limits. Only the super-user may use the optional *username* argument to view the limits of other users.

quota without options only display warnings about mounted file systems where usage is over quota. Remotely mounted file systems which do not have quotas turned on are ignored.

username can be the numeric UID of a user.

OPTIONS

-v Display user's quota on all mounted file systems where quotas exist.

USAGE

See largefile(5) for the description of the behavior of quota when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

FILES

/etc/mnttab list of currently mounted filesystems

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

edquota(1M), quotaon(1M), quotacheck(1M), repquota(1M), rquotad(1M), attributes(5), largefile(5)

NOTES

quota will also display quotas for NFS mounted ufs-based file systems if the rquotad daemon is running. See rquotad(1M).

quota may display entries for the same file system multiple times for multiple mount points. For example,

```
quota -v user1
```

may display identical quota information for user1 at the mount points /home/user1, /home/user2, and /home/user, if all three mount points are mounted from the same file system with quotas turned on.

NAME

quotacheck – ufs file system quota consistency checker

SYNOPSIS

quotacheck [-fp] [-v] *filesystem*...

quotacheck -a [-fpv]

DESCRIPTION

quotacheck examines each mounted ufs file system, builds a table of current disk usage, and compares this table against the information stored in the file system's disk quota file. If any inconsistencies are detected, both the quota file and the current system copy of the incorrect quotas are updated.

filesystem is either a file system mount point or the block device on which the file system resides.

quotacheck expects each file system to be checked to have a quota file named quotas in the root directory. If none is present, quotacheck will not check the file system.

quotacheck accesses the character special device in calculating the actual disk usage for each user. Thus, the file systems that are checked should be quiescent while quotacheck is running.

OPTIONS

The following options are supported:

-a Check the file systems which /etc/mnttab indicates are ufs file systems. These file systems must be read-write mounted with disk quotas enabled, and have an rq entry in the mntopts field in /etc/vfstab.

-f Force check on file systems with logging enabled. Use in combination with the -p option.

-p Check quotas of file systems in parallel. For file systems with logging enabled, no check is performed unless the -f option is also specified.

-v Indicate the calculated disk quotas for each user on a particular file system. quotacheck normally reports only those quotas modified.

USAGE

See largefile(5) for the description of the behavior of quotacheck when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

FILES

/etc/mnttab Mounted file systems

/etc/vfstab List of default parameters for each file system

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

edquota(1M), quota(1M), quotaon(1M), repquota(1M), attributes(5), largefile(5), quotactl(7I), mount_ufs(1M)

NAME

quotaon, quotaoff – turn ufs file system quotas on and off

SYNOPSIS

quotaon [-v] *filesystem*...

quotaon -a [-v]

quotaoff [-v] *filesystem*...

quotaoff -a [-v]

DESCRIPTION

quotaon turns on disk quotas for one or more ufs file systems.

Before a file system may have quotas enabled, a file named quotas, owned by root, must exist in the root directory of the file system. See edquota(1M) for details on how to modify the contents of this file.

quotaoff turns off disk quotas for one or more ufs file systems.

The file systems specified must already be mounted.

These commands update the mntopts field of the appropriate entries in /etc/mnttab to indicate when quotas are on or off for each file system. If quotas are on, the string quota will be added to mntopts; if quotas are off, the quota string is not present.

filesystem must be either the mount point of a file system, or the block device on which the file system resides.

quotaon

-a This option is normally used at boot time to enable quotas. It applies only to those file systems in /etc/vfstab which have "rq" in the mntopts field, are currently mounted "rw", and have a quotas file in the root directory.

-v Display a message for each file system after quotas are turned on.

quotaoff

-a Force all file systems in /etc/mnttab to have their quotas disabled.

-v Display a message for each file system affected.

USAGE

See largefile(5) for the description of the behavior of quotaon and quotaoff when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

FILES

/etc/mnttab mounted file systems

/etc/vfstab list of default parameters for each file system

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO | `edquota`(1M), `quota`(1M), `quotacheck`(1M), `repquota`(1M), `mnttab`(4), `vfstab`(4), `attributes`(5), `largefile`(5), `quotactl`(7I)

NAME

rctladm – display and/or modify global state of system resource controls

SYNOPSIS

rctladm [-lu] [-e *action*] [-d *action*] [*name*...]

DESCRIPTION

The rctladm command allows the examination and modification of active resource controls on the running system. An instance of a resource control is referred to as an *rctl*. See setrctl(2) for further description of an rctl. Logging of rctl violations can be activated or deactivated system-wide and active rctls (and their state) can be listed.

OPTIONS

The rctladm command supports the following options:

-l
: List available rctls with event status. This option displays the global event actions available for each rctl, and by the action name used with the enable (-e) and disable (-d) options below. The global flag values for the control are also displayed. If one or more name operands are specified, only those rctls matching the given names will be displayed.

  This is the default action if no options are specified.

-e *action*
-d *action*
: Enable or disable, respectively, the global action on the specified rctl(s). If no rctl is specified, no action is taken and an error status is returned. You can use the special token all with the disable option to deactivate all global actions on a resource control.

  You can set the syslog action to a specific degree by assigning a severity level. To do this, specify syslog=*level*, where *level* is one of the string tokens given as valid severity levels in syslog(3C). You can omit the common LOG_ prefix on the severity level.

-u
: Configure resource controls based on the contents of /etc/rctladm.conf. Any name operands are ignored.

OPERANDS

The following operands are supported:

name
: The name of the rctl to operate on. You can specify multiple rctl names may be specified. If no names are specified, and the list action has been specified, then all rctls are listed. If the enable or disable action is specified, one or more rctl names must be specified.

EXAMPLES

EXAMPLE 1 Activating System Logging for Specific Violations

The following command activates system logging of all violations of task.max-lwps.

```
# rctladm -e syslog task.max-lwps
#
```

EXAMPLE 1 Activating System Logging for Specific Violations *(Continued)*

EXAMPLE 2 Examining Current Status of a Specific Resource

The following command examines the current status of the `task.max-lwps` resource.

```
$ rctladm -l task.max-lwps
task.max-lwps               syslog=DEBUG
$
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` Fatal error occurred.

`2` Invalid command line options were specified

FILES

`/etc/rctladm.conf` Each time `rctladm` is executed, it updates the contents of `rctladm.conf` with the current configuration.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWesu

SEE ALSO

`setrctl`(2), `getrctl`(2), `prctl`(1), `rctlblk_get_global_flags`(3C), `rctlblk_get_global_action`(3C), `attributes`(5)

NOTES

By default, the base operating system provides a variety of controls:

```
project.cpu-shares

task.max-lwps
task.max-cpu-time

process.max-cpu-time
process.max-file-size
process.max-data-size
process.max-stack-size
process.max-core-size
process.max-file-descriptor
process.max-address-space
```

The default properties of the process resource controls are described on `setrlimit`(2). Task and project resource controls are unenforced by default.

NAME

rdate – set system date from a remote host

SYNOPSIS

`rdate` *hostname*

DESCRIPTION

`rdate` sets the local date and time from the *hostname* given as an argument. You must have the authorization `solaris.system.date` on the local system. Typically, `rdate` is used in a startup script.

The `inetd` daemon responds to `rdate` requests. To enable `inetd` response, the lines invoking the `time` command in `inetd.conf` must not be commented out.

USAGE

The `rdate` command is IPv6–enabled. See `ip6`(7P).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWrcmdc

SEE ALSO

`inetd`(1M), `inetd.conf`(4), `attributes`(5), `ip6`(7P)

NAME

reboot – restart the operating system

SYNOPSIS

/usr/sbin/reboot [-dlnq] *boot_arguments*

DESCRIPTION

The reboot utility restarts the kernel. The kernel is loaded into memory by the PROM monitor, which transfers control to the loaded kernel.

Although reboot can be run by the super-user at any time, shutdown(1M) is normally used first to warn all users logged in of the impending loss of service. See shutdown(1M) for details.

The reboot utility performs a sync(1M) operation on the disks, and then a multi-user reboot is initiated. See init(1M) for details.

The reboot utility normally logs the reboot to the system log daemon, syslogd(1M), and places a shutdown record in the login accounting file /var/adm/wtmpx. These actions are inhibited if the -n or -q options are present.

Normally, the system reboots itself at power-up or after crashes.

OPTIONS

-d	Force a system crash dump before rebooting. See dumpadm(1M) for information on configuring system crash dumps.
-l	Suppress sending a message to the system log daemon, syslogd(1M) about who executed reboot.
-n	Avoid calling sync(2) and do not log the reboot to syslogd(1M) or to /var/adm/wtmpx. The kernel still attempts to sync filesystems prior to reboot, except if the -d option is also present. If -d is used with -n, the kernel does not attempt to sync filesystems.
-q	Quick. Reboot quickly and ungracefully, without shutting down running processes first.

OPERANDS

boot_arguments	An optional *boot_arguments* string can be used to specify arguments to the uadmin(2) function that will be passed to the boot program and kernel upon restart. The form and list of arguments is described in the boot(1M) and kernel(1M) man pages. If multiple arguments are specified, they must be quoted for the shell, as shown in the EXAMPLES section. If the *boot_arguments* begin with a hyphen, they must be preceded by the – delimiter (two hyphens) to denote the end of the reboot argument list.

EXAMPLES

EXAMPLE 1 Passing the -r and -v Arguments to boot

In the following example, the delimiter '––' (two hyphens) must be used to separate the options of reboot from the arguments of boot(1M).

```
example# reboot -dl –– -rv
```

EXAMPLE 1 Passing the `-r` and `-v` Arguments to `boot` *(Continued)*

EXAMPLE 2 Rebooting Using a Specific Disk and Kernel

The following example reboots using a specific disk and kernel. Note that multiple boot arguments are quoted.

```
example# reboot "disk1 kernel.test/unix"
```

FILES

`/var/adm/wtmpx` login accounting file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`mdb`(1), `boot`(1M), `dumpadm`(1M), `fsck`(1M), `halt`(1M), `init`(1M), `kernel`(1M), `shutdown`(1M), `sync`(1M), `syslogd`(1M), `sync`(2), `uadmin`(2), `reboot`(3C), `attributes`(5)

NOTES

The `reboot` utility does not execute the scripts in `/etc/rc`*num*`.d` or execute shutdown actions in `inittab`(4). To ensure a complete shutdown of system services, use `shutdown`(1M) or `init`(1M) to reboot a Solaris system.

NAME

rem_drv – remove a device driver from the system

SYNOPSIS

rem_drv [-b *basedir*] *device_driver*

DESCRIPTION

The rem_drv command informs the system that the device driver *device_driver* is no longer valid. If possible, rem_drv unloads *device_driver* from memory. Entries for the device in the /devices namespace are removed. rem_drv also updates the system driver configuration files.

If rem_drv has been executed, the next time the system is rebooted it automatically performs a reconfiguration boot (see kernel(1M)).

OPTIONS

The following options are supported:

-b *basedir* Sets the path to the root directory of the diskless client. Used on the server to execute rem_drv for a client. The client machine must be rebooted to unload the driver.

EXAMPLES

EXAMPLE 1 Removing the sd Driver

The following example removes the sd driver from use:

```
example% rem_drv sd
```

EXAMPLE 2 Removing a Diskless Client

The next example removes the driver from the sun1 diskless client. The driver is not uninstalled or unloaded until the client machine is rebooted.

```
example% rem_drv -b /export/root/sun1 sd
```

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

add_drv(1M), drvconfig(1M), kernel(1M), attributes(5)

NAME

removef – remove a file from software database

SYNOPSIS

removef [[-M] -R *root_path*] [-V *fs_file*] *pkginst path...*

removef [[-M] -R *root_path*] [-V *fs_file*] -f *pkginst*

DESCRIPTION

`removef` informs the system that the user, or software, intends to remove a pathname. Output from `removef` is the list of input pathnames that may be safely removed (no other packages have a dependency on them).

OPTIONS

The following options are supported:

`-f` After all files have been processed, `removef` should be invoked with the `-f` option to indicate that the removal phase is complete.

`-M` Instruct `removef` not to use the $*root_path*`/etc/vfstab` file for determining the client's mount points. This option assumes the mount points are correct on the server and it behaves consistently with Solaris 2.5 and earlier releases.

`-R` *root_path* Define the full path name of a directory to use as the *root_path*. All files, including package system information files, are relocated to a directory tree starting in the specified *root_path*. The *root_path* may be specified when installing to a client from a server (for example, `/export/root/client1`).

`-V` *fs_file* Specify an alternative *fs_file* to map the client's file systems. For example, used in situations where the $*root_path*`/etc/vfstab` file is non-existent or unreliable.

OPERANDS

The following operands are supported:

path The pathname to be removed.

pkginst The package instance from which the pathname is being removed.

EXAMPLES

EXAMPLE 1 Using `removef`

The following example uses the `removef` command in an optional pre-install script:

```
echo "The following files are no longer part of this package
      and are being removed."
removef $PKGINST /dev/xt[0-9][0-9][0-9] |
while read pathname
do
     echo "$pathname"
     rm -f $pathname
done
removef -f $PKGINST || exit 2
```

EXIT STATUS

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`pkginfo`(1), `pkgmk`(1), `pkgparam`(1), `pkgproto`(1), `pkgtrans`(1), `installf`(1M), `pkgadd`(1M), `pkgask`(1M), `pkgchk`(1M), `attributes`(5)

Application Packaging Developer's Guide

NAME

repquota – summarize quotas for a ufs file system

SYNOPSIS

repquota [-v] *filesystem*...

repquota -a [-v]

DESCRIPTION

repquota prints a summary of the disk usage and quotas for the specified ufs file systems. The current number of files and amount of space (in kilobytes) is printed for each user along with any quotas created with edquota(1M).

The *filesystem* must have the file quotas in its root directory.

Only the super-user may view quotas which are not their own.

OPTIONS

The following options are supported:

-a Report on all mounted ufs file systems that have rq in the mntopts field of the /etc/vfstab file.

-v Report quotas for all users, even those who do not consume resources.

USAGE

See largefile(5) for the description of the behavior of repquota when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

edquota(1M), quota(1M), quotacheck(1M), quotaon(1M), attributes(5), largefile(5), quotactl(7I)

NAME

re-preinstall – installs the JumpStart software on a system

SYNOPSIS

cdrom-mnt-pt`/Solaris_`*XX*`/Tools/Boot/usr/sbin/install.d/re-preinstall`
`[-m` *Solaris_boot_dir*`] [-k` *platform_name*`]` *target-slice*

DESCRIPTION

`re-preinstall` installs the JumpStart software (preinstall boot image) on a system, so you can power-on the system and have it automatically install the Solaris software (perform a JumpStart installation on the system). When you turn on a re-preinstalled system, the system looks for the JumpStart software on the system's default boot disk. *All* new SPARC systems have the JumpStart software already preinstalled. The *XX* in Solaris_*XX* is the version number of the Solaris release being used.

You can use the `re-preinstall` command in several ways. The most common way is to run `re-preinstall` on a system to install the JumpStart software on its own default boot disk. This is useful if you want to restore a system to its original factory conditions. (See the first procedure described in `EXAMPLES`.)

You can also run `re-preinstall` on a system to install JumpStart software on any attached disk (non-boot disk). After you install the JumpStart software on a disk, you can move the disk to a different system and perform a JumpStart installation on the different system. (See the second procedure described in `EXAMPLES`.)

`re-preinstall` creates a standard file system on the specified *target-slice* (usually slice 0), and `re-preinstall` makes sure there is enough space on the *target-slice* for the JumpStart software. If sufficient space is not available, `re-preinstall` fails with the following message:

`re-preinstall:` *target-slice* `too small` *xx* `Megabytes required`

You can use the `format`(1M) command to create sufficient space on the *target-slice* for the JumpStart software.

OPTIONS

The following options are supported:

`-k` *platform_name* — Platform name of the system that will use the disk with the JumpStart software. The default is the platform name of the system running `re-preinstall`. (Use the `uname`(1) command (`-i` option) to determine a system's platform name.)

`-m` *Solaris_boot_dir* — Absolute path to the `Solaris_`*XX*`/Tools/Boot` subdirectory of a mounted Solaris CD or a Solaris CD copied to disk that `re-preinstall` uses to install the JumpStart software. The default is `/cdrom/Solaris_`*XX*`/Tools/Boot`, which is where the Solaris CD is mounted in single-user mode.

OPERANDS

The following operands are supported:

target-slice — Device name of the disk slice where the JumpStart software will be installed (usually slice 0), for example, `c0t3d0s0`.

EXAMPLES

EXAMPLE 1 Installing the JumpStart Software

The following procedure installs the JumpStart software on a system's own default boot disk:

1. From the `ok` prompt, boot the system from the Solaris CD (local or remote) in single-user mode:

   ```
   ok boot cdrom -s
   ```

2. With the `re-preinstall` command, install the JumpStart software on the system's default boot disk, which is a slice on the disk (usually slice 0) where the system automatically boots. (The system's default boot disk is probably where the current root (`/`) file system is located, which can be determined with the `format`(1M) command.)

   For example, the following command installs the JumpStart software on the system's default boot disk, `c0t3d0s0` on a Solaris 2.6 system:

   ```
   example# /cdrom/Solaris_2.6/Tools/Boot/usr/sbin/install.d\
                /re-preinstall c0t3d0s0
   ```

The following procedure installs the JumpStart software on a system's attached disk (non-boot disk):

1. Mount the Solaris CD if `vold`(1M) is not running or CD is not mounted.
2. Use the `format`(1M) command to determine the target-slice where JumpStart will be installed.
3. Use the `uname`(1) command (`-i` option) to determine the platform name of the system that will use the re-preinstalled disk
4. Run `re-preinstall` with the `-m` *Solaris_boot_dir* option if the Solaris CD is not mounted on `/cdrom`.

   For example, the following command installs the JumpStart software on the system's attached disk for a system with a Sun4c kernel architecture, and it uses the Solaris CD mounted with `vold`(1M) on a Solaris 2.6 system:

   ```
   example# /cdrom/cdrom0/s0/Solaris_2.6/Tools/Boot/usr/bin\
                /install.d/re-preinstall\
                -m /cdrom/cdrom0/s0/Solaris_2.6/Tools/Boot\
                -k sun4c c0t2d0s0
   ```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` An error has occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcdrom (Solaris CD, SPARC Platform Edition)

SEE ALSO `uname`(1), `eeprom`(1M), `format`(1M), `mount`(1M), `vold`(1M), `attributes`(5)

Solaris 9 Installation Guide

NAME

rmmount – removable media mounter for CD-ROM, floppy, Jaz drive, and others

SYNOPSIS

/usr/sbin/rmmount [-D]

DESCRIPTION

The `rmmount` utility is a removable media mounter that is executed by Volume Management whenever a removable medium, such as a CD-ROM or a floppy, is inserted. The Volume Management daemon, `vold`(1M), manages removable media. `rmmount` can also be called by using `volrmmount`(1).

Upon insertion of a medium and following invocation of the `volcheck`(1) command, `rmmount` determines what type of file system (if any) is on that medium. If a file system is present, `rmmount` mounts the file system in one of the locations listed below.

For a diskette (floppy):

`/floppy/floppy0`
  symbolic link to mounted floppy in local floppy drive

`/floppy/floppy_name`
  mounted named floppy

`/floppy/unnamed_floppy`
  mounted unnamed floppy

For a CD-ROM or a DVD-ROM:

`/cdrom/cdrom0`
  symbolic link to mounted CD-ROM in local CD-ROM drive

`/cdrom/CD-ROM_name`
  mounted named CD-ROM

`/cdrom/CD-ROM_name/partition`
  mounted named CD-ROM with partitioned file system

`/cdrom/unnamed_cdrom`
  mounted unnamed CD-ROM

For a Zip drive:

`/rmdisk/zip0`
  symbolic link to mounted Zip medium in local Zip drive

`/rmdisk/`*Zip_name*
  mounted named Zip medium

`/rmdisk/`*Zip_name*`/partition`
  mounted named Zip medium with partitioned file system

`/rmdisk/unnamed_zip`
  mounted unnamed Zip medium

For a Jaz drive:

`/rmdisk/jaz0`
symbolic link to mounted Jaz medium in local Jaz drive

`/rmdisk/`*Jaz_name*
mounted named Jaz medium

`/rmdisk/`*Jaz_name*`/partition`
mounted named Jaz medium with partitioned file system

`/rmdisk/unnamed_Jaz`
mounted unnamed Jaz medium

For a generic "rmdisk" drive:

`/rmdisk/rmdisk0`
symbolic link to mounted removable medium in local removable medium drive

`/rmdisk/`*rmdisk_name*
mounted named removable medium

`/rmdisk/`*rmdisk_name*`/partition`
mounted named removable medium with partitioned file system

`/rmdisk/unnamed_rmdisk`
mounted unnamed removable medium

If the media is read-only (for example, a CD-ROM or a floppy with write-protect tab set), the file system is mounted read-only.

If a file system is not identified, `rmmount` does not mount a file system. See the *System Administration Guide: Basic Administration* for more information on the location of CD-ROM, floppy, and other media without file systems. Also see `volfs`(7FS).

If a file system type has been determined, it is then checked to see that it is "clean." If the file system is "dirty," `fsck -p` (see `fsck`(1M)) is run in an attempt to clean it. If `fsck` fails, the file system is mounted read-only.

After the mount is complete, "actions" associated with the media type are executed. These actions allow for the notification to other programs that new media are available. These actions are shared objects and are described in the configuration file, `/etc/rmmount.conf`. See `rmmount.conf`(4).

Actions are executed in the order in which they appear in the configuration file. The action function can return either `1` or `0`. If it returns `0`, no further actions will be executed. This allows the function to control which applications are executed.

In order to execute an action, `rmmount` performs a `dlopen`(3DL) on the shared object and calls the action function defined within it. The definition of the interface to actions can be found in `/usr/include/rmmount.h`.

File systems mounted by `rmmount` are always mounted with the `nosuid` flag set, thereby disabling setuid programs and access to block or character devices in that file system. Upon ejection, `rmmount` unmounts mounted file systems and executes actions associated with the media type. If a file system is "busy" (that is, it contains the current working directory of a live process), the ejection will fail.

OPTIONS

`-D` Turn on the debugging output from the `rmmount` `dprintf` calls.

FILES

`/etc/rmmount.conf`	removable media mounter configuration file
`/usr/lib/rmmount/*.so.1`	shared objects used by `rmmount`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWvolu

SEE ALSO

`volcancel`(1), `volcheck`(1), `volmissing`(1), `volrmmount`(1), `fsck`(1M), `vold`(1M), `dlopen`(3DL), `rmmount.conf`(4), `vold.conf`(4), `attributes`(5), `volfs`(7FS)

System Administration Guide: Basic Administration

NAME

rmt – remote magtape protocol module

SYNOPSIS

/usr/sbin/rmt

DESCRIPTION

`rmt` is a program used by the remote dump and restore programs in manipulating a magnetic tape drive through an interprocess communication connection. `rmt` is normally started up with an `rexec`(3SOCKET) or `rcmd`(3SOCKET) call.

The `rmt` program accepts requests that are specific to the manipulation of magnetic tapes, performs the commands, then responds with a status indication. All responses are in ASCII and in one of two forms. Successful commands have responses of:

A*number*\n
where *number* is an ASCII representation of a decimal number.

Unsuccessful commands are responded to with:

E*error-number*\n*error-message*\n

where *error-number* is one of the possible error numbers described in `intro`(3), and *error-message* is the corresponding error string as printed from a call to `perror`(3C).

The protocol consists of the following commands:

S\n	Return the status of the open device, as obtained with a `MTIOCGET ioctl` call. If the operation was successful, an "ack" is sent with the size of the status buffer, then the status buffer is sent (in binary).
C*device*\n	Close the currently open device. The *device* specified is ignored.
I*operation*\n*count*\n	Perform a `MTIOCOP ioctl`(2) command using the specified parameters. The parameters are interpreted as the ASCII representations of the decimal values to place in the *mt_op* and *mt_count* fields of the structure used in the `ioctl` call. When the operation is successful the return value is the *count* parameter.
L*offset*\nwhence\n	Perform an `lseek`(2) operation using the specified parameters. The response value is returned from the `lseek` call.
O*device*\n*mode*\n	Open the specified *device* using the indicated *mode*. *device* is a full pathname, and *mode* is an ASCII representation of a decimal number suitable for passing to `open`(9E). If a device is already open, it is closed before a new open is performed.
R*count*\n	Read *count* bytes of data from the open device. `rmt` performs the requested `read`(9E) and responds with A*count-read*\n if the read was successful; otherwise an

error in standard format is returned. If the read was successful, the data read is sent.

`W`*count*`\n` — Write data onto the open device. `rmt` reads *count* bytes from the connection, aborting if a premature EOF is encountered. The response value is returned from the `write`(9E) call.

Any other command causes `rmt` to exit.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWrcmdc

SEE ALSO

`ufsdump`(1M), `ufsrestore`(1M), `intro`(3), `ioctl`(2), `lseek`(2), `perror` (3C), `rcmd`(3SOCKET), `rexec`(3SOCKET), `attributes`(5), `mtio`(7I), `open`(9E), `read`(9E), `write`(9E)

DIAGNOSTICS

All responses are of the form described above.

BUGS

Do not use this for a remote file access protocol.

NAME

roleadd – administer a new role account on the system

SYNOPSIS

roleadd [-c *comment*] [-d *dir*] [-e *expire*] [-f *inactive*] [-g *group*] [-G *group* [, *group*...]] [-m [-k *skel_dir*]] [-u *uid* [-o]] [-s *shell*] [-A *authorization* [,*authorization*...]] *role*

roleadd -D [-b *base_dir*] [-e *expire*] [-f *inactive*] [-g *group*] [-A *authorization* [,*authorization*...]] [-P *profile* [,*profile*...]]

DESCRIPTION

`roleadd` adds a role entry to the `/etc/passwd` and `/etc/shadow` and `/etc/user_attr` files. The `-A` and `-P` options respectively assign authorizations and profiles to the role. Roles cannot be assigned to other roles.

`roleadd` also creates supplementary group memberships for the role (`-G` option) and creates the home directory (`-m` option) for the role if requested. The new role account remains locked until the `passwd`(1) command is executed.

Specifying `roleadd -D` with the `-g`, `-b`, `-f`, or `-e` option (or any combination of these option) sets the default values for the respective fields. See the `-D` option. Subsequent `roleadd` commands without the `-D` option use these arguments.

The system file entries created with this command have a limit of 512 characters per line. Specifying long arguments to several options can exceed this limit.

The role (`role`) field accepts a string of no more than eight bytes consisting of characters from the set of alphabetic characters, numeric characters, period (`.`), underscore (_), and hyphen (`-`). The first character should be alphabetic and the field should contain at least one lower case alphabetic character. A warning message will be written if these restrictions are not met. A future Solaris release may refuse to accept role fields that do not meet these requirements.

The `role` field must contain at least one character and must not contain a colon (`:`) or a newline (`\n`).

OPTIONS

The following options are supported:

`-b` *base_dir*
: The default base directory for the system if `-d` *dir* is not specified. *base_dir* is concatenated with the account name to define the home directory. If the `-m` option is not used, *base_dir* must exist.

`-c` *comment*
: Any text string. It is generally a short description of the role. This information is stored in the role's `/etc/passwd` entry.

`-d` *dir*
: The home directory of the new role. It defaults to *base_dir*/*account_name*, where *base_dir* is the base directory for new login home directories and *account_name* is the new role name.

`-D`
: Display the default values for *group*, *base_dir*, *skel_dir*, *shell*, *inactive*, and *expire*. When used with the `-g`, `-b`, or `-f`, options, the `-D` option sets the default values for the specified fields. The default values are:

	group	`other` (GID of 1)
	base_dir	`/home`
	skel_dir	`/etc/skel`
	shell	`/bin/sh`
	inactive	`0`
	expire	Null
	auths	Null
	profiles	Null

`-e` *expire*
: Specify the expiration date for a role. After this date, no user will be able to access this role. The expire option argument is a date entered using one of the date formats included in the template file `/etc/datemsk`. See `getdate`(3C).

  If the date format that you choose includes spaces, it must be quoted. For example, you can enter `10/6/90` or `"October 6, 1990"`. A null value (`" "`) defeats the status of the expired date. This option is useful for creating temporary roles.

`-f` *inactive*
: The maximum number of days allowed between uses of a role ID before that ID is declared invalid. Normal values are positive integers. A value of `0` defeats the status.

`-g` *group*
: An existing group's integer ID or character-string name. Without the `-D` option, it defines the new role's primary group membership and defaults to the default group. You can reset this default value by invoking `roleadd -D -g` *group.*

`-G` *group*
: An existing group's integer ID or character-string name. It defines the new role's supplementary group membership. Duplicates between *group* with the `-g` and `-G` options are ignored. No more than `NGROUPS_MAX` groups can be specified.

`-k` *skel_dir*
: A directory that contains skeleton information (such as `.profile`) that can be copied into a new role's home directory. This directory must already exist. The system provides the `/etc/skel` directory that can be used for this purpose.

`-m`
: Create the new role's home directory if it does not already exist. If the directory already exists, it must have read, write, and execute permissions by *group*, where *group* is the role's primary group.

`-o`
: This option allows a UID to be duplicated (non-unique).

`-s` *shell*
: Full pathname of the program used as the user's shell on login. It defaults to an empty field causing the system to use `/bin/sh` as the default. The value of *shell* must be a valid executable file.

`-u` *uid*
: The UID of the new role. This UID must be a non-negative decimal integer below `MAXUID` as defined in `<sys/param.h>`. The UID defaults to the next available (unique) number above the highest number currently assigned. For example, if UIDs 100, 105, and 200 are assigned, the next default UID number will be 201. (UIDs from `0-99` are reserved for possible use in future applications.)

FILES

`/etc/datemsk`

`/etc/passwd`

`/etc/shadow`

`/etc/group`

`/etc/skel`

`/usr/include/limits.h`

`/etc/user_attr`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`passwd`(1)`profiles`(1)`roles`(1), `users`(1B), `groupadd`(1M), `groupdel`(1M), `groupmod`(1M), `grpck`(1M), `logins`(1M), `pwck`(1M), `userdel`(1M), `usermod`(1M), `getdate`(3C), `auth_attr`(4), `passwd`(4), `prof_attr`(4), `user_attr`(4), `attributes`(5)

DIAGNOSTICS

In case of an error, `roleadd` prints an error message and exits with a non-zero status.

The following indicates that `login` specified is already in use:

```
UX: roleadd: ERROR: login is already in use. Choose another.
```

The following indicates that the *uid* specified with the `-u` option is not unique:

```
UX: roleadd: ERROR: uid uid is already in use. Choose another.
```

The following indicates that the *group* specified with the `-g` option is already in use:

```
UX: roleadd: ERROR: group group does not exist. Choose another.
```

The following indicates that the *uid* specified with the `-u` option is in the range of reserved UIDs (from `0-99`):

```
UX: roleadd: WARNING: uid uid is reserved.
```

The following indicates that the *uid* specified with the `-u` option exceeds `MAXUID` as defined in `<sys/param.h>`:

```
UX: roleadd: ERROR: uid uid is too big. Choose another.
```

The following indicates that the `/etc/passwd` or `/etc/shadow` files do not exist:

```
UX: roleadd: ERROR: Cannot update system files - login cannot be created.
```

NOTES If a network nameservice such as NIS or NIS+ is being used to supplement the local `/etc/passwd` file with additional entries, `roleadd` cannot change information supplied by the network nameservice.

NAME

roledel – delete a role's login from the system

SYNOPSIS

`roledel` `[-r]` *role*

DESCRIPTION

The `roledel` utility deletes a role account from the system and makes the appropriate account-related changes to the system file and file system. `roledel` also removes the role from each user's list of assumable roles.

OPTIONS

The following options are supported:

`-r` Remove the role's home directory from the system. This directory must exist. The files and directories under the home directory will no longer be accessible following successful execution of the command.

OPERANDS

The following operands are supported:

role An existing role name to be deleted.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`2` Invalid command syntax. A usage message for the `roledel` command is displayed.

`6` The account to be removed does not exist.

`8` The account to be removed is in use.

`10` Cannot update the `/etc/group` or `/etc/user_attr` file but the login is removed from the `/etc/passwd` file.

`12` Cannot remove or otherwise modify the home directory.

FILES

`/etc/passwd` system password file

`/etc/shadow` system file containing roles' encrypted passwords and related information

`/etc/group` system file containing group definitions

`/etc/user_attr` system file containing additional role attributes

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO auths(1), passwd(1), profiles(1), roles(1), users(1B), groupadd(1M), groupdel(1M), groupmod(1M), logins(1M), roleadd(1M), rolemod(1M), useradd(1M), userdel(1M), usermod(1M), passwd(4), prof_attr(4), user_attr(4), attributes(5)

NOTES The roledel utility only deletes an account definition that is in the local /etc/group, /etc/passwd, /etc/shadow, and /etc/user_attr file. file. If a network name service such as NIS or NIS+ is being used to supplement the local /etc/passwd file with additional entries, roledel cannot change information supplied by the network name service.

NAME

rolemod – modify a role's login information on the system

SYNOPSIS

rolemod [-u *uid* [-o]] [-g *group*] [-G *group* [, *group*...]] [-d *dir* [-m]] [-s *shell*] [-c *comment*] [-l *new_name*] [-f *inactive*] [-e *expire*] [-A *authorization* [, *authorization*]] [-P *profile* [, *profile*]] *role*

DESCRIPTION

The `rolemod` utility modifies a role's login information on the system. It changes the definition of the specified login and makes the appropriate login-related system file and file system changes.

The system file entries created with this command have a limit of 512 characters per line. Specifying long arguments to several options may exceed this limit.

OPTIONS

The following options are supported:

`-A` *authorization* — One or more comma separated authorizations as deined in `auth_attr`(4). Only role with `grant` rights to the `authorization` can assign it to an account. This replaces any existing authorization setting.

`-c` *comment* — Specify a comment string. *comment* can be any text string. It is generally a short description of the login, and is currently used as the field for the user's full name. This information is stored in the user's `/etc/passwd` entry.

`-d` *dir* — Specify the new home directory of the role. It defaults to *base_dir/login*, where *base_dir* is the base directory for new login home directories, and `login` is the new login.

`-e` *expire* — Specify the expiration date for a role. After this date, no role will be able to access this login. The expire option argument is a date entered using one of the date formats included in the template file `/etc/datemsk`. See `getdate`(3C).

For example, you may enter `10/6/90` or `October 6, 1990`. A value of `'' ''` defeats the status of the expired date.

`-f` *inactive* — Specify the maximum number of days allowed between uses of a login ID before that login ID is declared invalid. Normal values are positive integers. A value of `0` defeats the status.

`-g` *group* — Specify an existing group's integer ID or character-string name. It redefines the role's primary group membership.

`-G` *group* — Specify an existing group's integer "ID" "," or character string name. It redefines the role's supplementary

group membership. Duplicates between *group* with the `-g` and `-G` options are ignored. No more than `NGROUPS_UMAX` groups may be specified as defined in `<param.h>`.

`-l` *new_logname*
: Specify the new login name for the role. The *new_logname* argument is a string no more than eight bytes consisting of characters from the set of alphabetic characters, numeric characters, period (`.`), underline (_), and hypen (–). The first character should be alphabetic and the field should contain at least one lower case alphabetic character. A warning message will be written if these restrictions are not met. A future Solaris release may refuse to accept login fields that do not meet these requirements. The *new_logname* argument must contain at least one character and must not contain a colon (`:`) or NEWLINE (`\n`).

`-m`
: Move the role's home directory to the new directory specified with the `-d` option. If the directory already exists, it must have permissions read/write/execute by *group*, where *group* is the role's primary group.

`-o`
: This option allows the specified UID to be duplicated (non-unique).

`-P` *profile*
: One or more comma-separated execution profiles defined in `auth_attr`(4). This replaces any existing profile setting.

`-s` *shell*
: Specify the full pathname of the program that is used as the role's shell on login. The value of *shell* must be a valid executable file.

`-u` *uid*
: Specify a new UID for the role. It must be a non-negative decimal integer less than `MAXUID` as defined in `<param.h>`. The UID associated with the role's home directory is not modified with this option; a role will not have access to their home directory until the UID is manually reassigned using `chown`(1M).

OPERANDS

The following operands are supported:

`login`
: An existing login name to be modified.

EXIT STATUS

In case of an error, `rolemod` prints an error message and exits with one of the following values:

`2`
: The command syntax was invalid. A usage message for the `rolemod` command is displayed.

`3`
: An invalid argument was provided to an option.

4	The *uid* given with the `-u` option is already in use.
5	The password files contain an error. `pwconv`(1M) can be used to correct possible errors. See `passwd`(4).
6	The login to be modified does not exist, the *group* does not exist, or the login shell does not exist.
8	The login to be modified is in use.
9	The *new_logname* is already in use.
10	Cannot update the `/etc/group` or `/etc/user_attr` file. Other update requests will be implemented.
11	Insufficient space to move the home directory (`-m` option). Other update requests will be implemented.
12	Unable to complete the move of the home directory to the new home directory.

FILES

`/etc/group`	system file containing group definitions
`/etc/datemsk`	system file of date formats
`/etc/passwd`	system password file
`/etc/shadow`	system file containing users' and roles' encrypted passwords and related information
`/etc/usr_attr`	system file containing additional user and role attributes

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`passwd`(1), `users`(1B), `chown`(1M), `groupadd`(1M), `groupdel`(1M), `groupmod`(1M), `logins`(1M), `pwconv`(1M), `roleadd`(1M), `roledel`(1M), `useradd`(1M), `userdel`(1M), `usermod`(1M), `getdate`(3C), `auth_attr`(4), `passwd`(4), `attributes`(5)

NAME

route – manually manipulate the routing tables

SYNOPSIS

route [-fnvq] *command* [[*modifiers*] *args*]

route [-fnvq]add | delete [*modifiers*] *destination gateway* [*args*]

route [-fnvq]change | get [*modifiers*] *destination* [*gateway* [*args*]]

route [-n] monitor [*modifiers*]

route [-n] flush [*modifiers*]

DESCRIPTION

route manually manipulates the network routing tables. These tables are normally maintained by the system routing daemon, such as in.routed(1M) and in.ripngd(1M).

This utility supports a limited number of general options, but a rich command language. It enables the user to specify any arbitrary request that could be delivered by means of the programmatic interface discussed in route(7P).

route uses a routing socket and the new message types RTM_ADD, RTM_DELETE, RTM_GET, and RTM_CHANGE. As such, only the superuser may modify the routing tables.

OPTIONS

-f Flush the routing tables of all gateway entries. If this is used in conjunction with one of the commands described above, route flushes the gateways before performing the command.

-n Prevent attempts to print host and network names symbolically when reporting actions. This is useful, for example, when all name servers are down on your local net, and you need a route before you can contact the name server.

-v (Verbose) Print additional details.

-q Suppress all output.

Commands

route executes one of four *commands* on a route to a *destination*. Two additional *commands* operate globally on all routing information. The six commands are:

add Add a route.

change Change aspects of a route (such as its gateway).

delete Delete a specific route.

flush Remove all gateway entries from the routing table.

get Lookup and display the route for a destination.

monitor Continuously report any changes to the routing information base, routing lookup misses, or suspected network partitionings.

The add, delete, and change commands have the following syntax:

```
route [ -fnvq ] add | delete [ -net | -host ] destination gateway [args]
```

or

`route [ -fnvq ] change | get [ -net | -host ] destination gateway [args]` where *destination* is the destination host or network, and *gateway* is the next-hop intermediary through which packets should be routed.

OPERANDS

`route` executes its commands on routes to destinations.

Destinations

By default, a destination is looked up under the `AF_INET` address family or as an IPv4 address. All symbolic names specified for a destination or gateway are looked up first as a host name, using `getipnodebyname`(3SOCKET). If this lookup fails in the `AF_INET` case, `getnetbyname`(3SOCKET) is used to interpret the name as that of a network.

An optional modifier may be included on the command line before a *destination*, to force how `route` interprets a destination:

`-host`	Forces the destination to be interpreted as a host.
`-net`	Forces the destination to be interpreted as a network.
`-inet`	Forces the destination to be interpreted under the `AF_INET` address family or as an IPv4 address.
`-inet6`	Forces the destination to be interpreted under the `AF_INET6` address family or as an IPv6 address.

In the case of the `AF_INET` address family or an IPv4 address, routes to a particular host may be distinguished from those to a network by interpreting the Internet address specified as the *destination*. If the *destination* has a "local address part" of `INADDR_ANY`, or if the *destination* is the symbolic name of a network, then the route is assumed to be to a network; otherwise, it is presumed to be a route to a host.

For example:

The following route:	**Is interpreted as:**
128.32	`-host` 128.0.0.32
128.32.130	`-host` 128.32.0.130
`-net` 128.32	128.32.0.0
`-net` 128.32.130	128.32.130.0

If the destination is directly reachable by way of an interface requiring no intermediary system to act as a gateway, this can be indicated by including one of two optional modifiers after the destination: The `-interface` modifier can be included or a *metric* of `0` can be specified. These modifiers are illustrated in the following alternative examples:

```
example% route add default hostname -interface
example% route add default hostname 0
```

hostname is the name or IP address associated with the network interface all packets should be sent over. On a host with a single network interface, hostname is normally the same as the *nodename* returned by uname -n (see uname(1)).

In the above examples, the route does not refer to a gateway, but rather to one of the machine's interfaces. Destinations matching such a route are sent out on the interface identified by the *gateway* address. For interfaces using the ARP protocol, this type of route is used to specify *all destinations are local*. That is, a host should ARP for all addresses by adding a default route using one of the two commands listed above.

With the AF_INET address family or an IPv4 address, the optional -netmask qualifier is intended to manually add subnet routes with netmasks different from that of the implied network interface. The implicit network mask generated in the AF_INET case can be overridden by making sure this option, and an ensuing address parameter (to be interpreted as a network mask), follows the destination parameter.

Alternatively, the length of the netmask may be supplied by appending a slash character and the length immediately after the destination. For example:

example% **route add 192.0.2.32/27 somegateway** will create an IPv4 route to the destination 192.0.2.32 with a netmask of 255.255.255.224, and

example% **route add -inet6 3ffe::/16 somegateway** will create an IPv6 route to the destination 33fe:: with a netmask of 16 one-bits followed by 112 zero-bits.

Routing Flags

Routes have associated flags which influence operation of the protocols when sending to destinations matched by the routes. These flags may be set (or sometimes cleared) by including the following corresponding modifiers on the command line:

Modifier	Flag	Description
-cloning	RTF_CLONING	generates a new route on use
-xresolve	RTF_XRESOLVE	emit mesg on use (for external lookup)
-iface	~RTF_GATEWAY	destination is directly reachable
-static	RTF_STATIC	manually added route
-nostatic	~RTF_STATIC	pretend route added by kernel or daemon
-reject	RTF_REJECT	emit an ICMP unreachable when matched
-blackhole	RTF_BLACKHOLE	silently discard pkts (during updates)
-proto1	RTF_PROTO1	set protocol specific routing flag #1

Modifier	Flag	Description
-proto2	RTF_PROTO2	set protocol specific routing flag #2
-private	RTF_PRIVATE	do not adveritse this route

The optional modifiers -rtt, -rttvar, -sendpipe, -recvpipe, -mtu, -hopcount, -expire, and -ssthresh provide initial values to quantities maintained in the routing entry by transport level protocols, such as TCP. These may be individually locked either by preceding each modifier to be locked by the -lock meta-modifier, or by specifying that all ensuing metrics may be locked by the -lockrest meta-modifier.

The optional modifiers are defined as follows:

-expire Lifetime for the entry. This optional modifier is not currently supported.

-hopcount Maximum hop count. This optional modifier is not currently supported.

-mtu Maximum MTU in bytes.

-recvpipe Receive pipe size in bytes.

-rtt Round trip time in microseconds.

-rttvar Round trip time variance in microseconds.

-sendpipe Send pipe size in bytes.

-ssthresh Send pipe size threshold in bytes.

Some transport layer protocols may support only some of these metrics.

In a change or add command where the destination and gateway are not sufficient to specify the route (for example, , when several interfaces have the same address), the -ifp or -ifa modifiers may be used to determine the interface or interface address.

FILES

/etc/defaultrouter list of default routers

/etc/hosts list of host names and net addresses

/etc/networks list of network names and addresses

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`get`(1), `uname`(1), `in.rdisc`(1M), `netstat`(1M), `routed`(1M), `ioctl`(2), `getipnodebyname`(3SOCKET), `getnetbyname`(3SOCKET), `defaultrouter`(4), `hosts`(4), `networks`(4), `attributes`(5), `ARP`(7P), `ip`(7P), `route`(7P), `routing`(7P)

DIAGNOSTICS

`add` [`host` | `network`] *destination*:*gateway flags*
The specified route is being added to the tables. The values printed are from the routing table entry supplied in the `ioctl`(2) call. If the gateway address used was not the primary address of the gateway (the first one returned by `getipnodebyname`(3SOCKET)) the gateway address is printed numerically as well as symbolically.

`delete` [`host` | `network`] *destination*:*gateway flags*
As above, but when deleting an entry.

destination `done`
When the `-f` flag is specified, or in the `flush` command, each routing table entry deleted is indicated with a message of this form.

`Network is unreachable`
An attempt to add a route failed because the gateway listed was not on a directly-connected network. Give the next-hop gateway instead.

`not in table`
A delete operation was attempted for an entry that is not in the table.

`routing table overflow`
An add operation was attempted, but the system was unable to allocate memory to create the new entry.

NOTES

All destinations are local assumes that the routers implement the protocol, `proxy arp`. Normally, using `router discovery` (see `in.rdisc`(1M)) is more reliable than using `proxy arp`.

Combining the *all destinations are local* route with subnet or network routes can lead to unpredictable results: the search order as it relates to the *all destinations are local* route are undefined and may vary from release to release.

NAME

rpcbind – universal addresses to RPC program number mapper

SYNOPSIS

```
rpcbind [-d] [-w]
```

DESCRIPTION

`rpcbind` is a server that converts RPC program numbers into universal addresses. It must be running on the host to be able to make RPC calls on a server on that machine.

When an RPC service is started, it tells `rpcbind` the address at which it is listening, and the RPC program numbers it is prepared to serve. When a client wishes to make an RPC call to a given program number, it first contacts `rpcbind` on the server machine to determine the address where RPC requests should be sent.

`rpcbind` should be started before any other RPC service. Normally, standard RPC servers are started by port monitors, so `rpcbind` must be started before port monitors are invoked.

When `rpcbind` is started, it checks that certain name-to-address translation-calls function correctly. If they fail, the network configuration databases may be corrupt. Since RPC services cannot function correctly in this situation, `rpcbind` reports the condition and terminates.

`rpcbind` can only be started by the super-user.

OPTIONS

The following options are supported:

`-d` Run in debug mode. In this mode, `rpcbind` will not fork when it starts, will print additional information during operation, and will abort on certain errors. With this option, the name-to-address translation consistency checks are shown in detail.

`-w` Do a warm start. If `rpcbind` aborts or terminates on `SIGINT` or `SIGTERM`, it will write the current list of registered services to `/tmp/portmap.file` and `/tmp/rpcbind.file`. Starting `rpcbind` with the `-w` option instructs it to look for these files and start operation with the registrations found in them. This allows `rpcbind` to resume operation without requiring all RPC services to be restarted.

FILES

`/tmp/portmap.file`

`/tmp/rpcbind.file`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`rpcinfo`(1M), `rpcbind`(3NSL), `attributes`(5)

NOTES Terminating `rpcbind` with `SIGKILL` will prevent the warm-start files from being written.

All RPC servers must be restarted if the following occurs: `rpcbind` crashes (or is killed with `SIGKILL`) and is unable to to write the warm-start files; `rpcbind` is started without the `-w` option after a graceful termination; or, the warm-start files are not found by `rpcbind`.

NAME

rpc.bootparamd, bootparamd – boot parameter server

SYNOPSIS

/usr/sbin/rpc.bootparamd [-d]

DESCRIPTION

`rpc.bootparamd` is a server process that provides information from a `bootparams` database to diskless clients at boot time. See `bootparams`(4)

The source for the `bootparams` database is determined by the `nsswitch.conf`(4) file (on the machine running the `rpc.bootparamd` process).

The `rpc.bootparamd` program can be invoked either by `inetd`(1M) or directly from the command line.

OPTIONS

`-d` Display debugging information.

FILES

`/etc/bootparams` boot parameter data base

`/etc/nsswitch.conf` configuration file for the name-service switch

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWbsu

SEE ALSO

`inetd`(1M), `bootparams`(4), `nsswitch.conf`(4), `attributes`(5)

NOTES

A diskless client requires service from at least one `rpc.bootparamd` process running on a server that is on the same IP subnetwork as the diskless client.

Some routines that compare hostnames use case-sensitive string comparisons; some do not. If an incoming request fails, verify that the case of the hostname in the file to be parsed matches the case of the hostname called for, and attempt the request again.

NAME

rpcinfo – report RPC information

SYNOPSIS

```
rpcinfo [-m | -s] [host]
rpcinfo -p [host]
rpcinfo -T transport host prognum [versnum]
rpcinfo -l [-T transport] host prognum versnum
rpcinfo [-n portnum] -u host prognum [versnum]
rpcinfo [-n portnum] -t host prognum [versnum]
rpcinfo -a serv_address -T transport prognum [versnum]
rpcinfo -b [-T transport] prognum versnum
rpcinfo -d [-T transport] prognum versnum
```

DESCRIPTION

`rpcinfo` makes an RPC call to an RPC server and reports what it finds.

In the first synopsis, `rpcinfo` lists all the registered RPC services with `rpcbind` on *host*. If *host* is not specified, the local host is the default. If `-s` is used, the information is displayed in a concise format.

In the second synopsis, `rpcinfo` lists all the RPC services registered with `rpcbind`, version 2. Note that the format of the information is different in the first and the second synopsis. This is because the second synopsis is an older protocol used to collect the information displayed (version 2 of the `rpcbind` protocol).

The third synopsis makes an RPC call to procedure 0 of *prognum* and *versnum* on the specified *host* and reports whether a response was received. *transport* is the transport which has to be used for contacting the given service. The remote address of the service is obtained by making a call to the remote `rpcbind`.

The *prognum* argument is a number that represents an RPC program number (see `rpc`(4)).

If a *versnum* is specified, `rpcinfo` attempts to call that version of the specified *prognum*. Otherwise, `rpcinfo` attempts to find all the registered version numbers for the specified *prognum* by calling version 0, which is presumed not to exist; if it does exist, `rpcinfo` attempts to obtain this information by calling an extremely high version number instead, and attempts to call each registered version. Note that the version number is required for `-b` and `-d` options.

The `EXAMPLES` section describe other ways of using `rpcinfo`.

OPTIONS

`-T` *transport*
: Specify the transport on which the service is required. If this option is not specified, `rpcinfo` uses the transport specified in the `NETPATH` environment variable, or if that is unset or `NULL`, the transport in the `netconfig`(4) database is used. This is a generic

option, and can be used in conjunction with other options as shown in the SYNOPSIS.

`-a` *serv_address*
: Use *serv_address* as the (universal) address for the service on *transport* to `ping` procedure 0 of the specified *prognum* and report whether a response was received. The `-T` option is required with the `-a` option. If *versnum* is not specified, `rpcinfo` tries to ping all available version numbers for that program number. This option avoids calls to remote `rpcbind` to find the address of the service. The *serv_address* is specified in universal address format of the given transport.

`-b`
: Make an RPC broadcast to procedure 0 of the specified *prognum* and *versnum* and report all hosts that respond. If *transport* is specified, it broadcasts its request only on the specified transport. If broadcasting is not supported by any transport, an error message is printed. Use of broadcasting should be limited because of the potential for adverse effect on other systems.

`-d`
: Delete registration for the RPC service of the specified *prognum* and *versnum*. If *transport* is specified, unregister the service on only that transport, otherwise unregister the service on all the transports on which it was registered. Only the owner of a service can delete a registration, except the superuser, who can delete any service.

`-l`
: Display a list of entries with a given *prognum* and *versnum* on the specified *host*. Entries are returned for all transports in the same protocol family as that used to contact the remote `rpcbind`.

`-m`
: Display a table of statistics of `rpcbind` operations on the given *host*. The table shows statistics for each version of `rpcbind` (versions 2, 3 and 4), giving the number of times each procedure was requested and successfully serviced, the number and type of remote call requests that were made, and information about RPC address lookups that were handled. This is useful for monitoring RPC activities on *host*.

`-n` *portnum*
: Use *portnum* as the port number for the `-t` and `-u` options instead of the port number given by `rpcbind`. Use of this option avoids a call to the remote `rpcbind` to find out the address of the service. This option is made obsolete by the `-a` option.

`-p` Probe `rpcbind` on *host* using version 2 of the `rpcbind` protocol, and display a list of all registered RPC programs. If *host* is not specified, it defaults to the local host. This option is not useful for IPv6; use `-s` (see below) instead. Note that version 2 of the `rpcbind` protocol was previously known as the portmapper protocol.

`-s` Display a concise list of all registered RPC programs on *host*. If *host* is not specified, it defaults to the local host.

`-t` Make an RPC call to procedure 0 of *prognum* on the specified *host* using TCP, and report whether a response was received. This option is made obsolete by the `-T` option as shown in the third synopsis.

`-u` Make an RPC call to procedure 0 of *prognum* on the specified *host* using UDP, and report whether a response was received. This option is made obsolete by the `-T` option as shown in the third synopsis.

EXAMPLES

EXAMPLE 1 RPC services.

To show all of the RPC services registered on the local machine use:

```
example% rpcinfo
```

To show all of the RPC services registered with `rpcbind` on the machine named `klaxon` use:

```
example% rpcinfo klaxon
```

The information displayed by the above commands can be quite lengthy. Use the `-s` option to display a more concise list:

```
example% rpcinfo -s klaxon
```

program	vrsn	netid(s)	service	owner
100000	2,3,4	tcp,udp,ticlts,ticots,ticotsord	rpcbind	superuser
100008	1	ticotsord,ticots,ticlts,udp,tcp	walld	superuser
100002	2,1	ticotsord,ticots,ticlts,udp,tcp	rusersd	superuser
100001	2,3,4	ticotsord,ticots,tcp,ticlts,udp	rstatd	superuser
100012	1	ticotsord,ticots,ticlts,udp,tcp	sprayd	superuser
100007	3	ticotsord,ticots,ticlts,udp,tcp	ypbind	superuser
100029	1	ticotsord,ticots,ticlts	keyserv	superuser

EXAMPLE 1 RPC services. *(Continued)*

```
   100078 4      ticotsord,ticots,ticlts                          superuser
   100024 1      ticotsord,ticots,ticlts,udp,tcp     status       superuser
   100021 2,1    ticotsord,ticots,ticlts,udp,tcp     nlockmgr     superuser
   100020 1      ticotsord,ticots,ticlts,udp,tcp     llockmgr     superuser
```

To show whether the RPC service with program number *prognum* and version *versnum* is registered on the machine named `klaxon` for the transport TCP use:

```
example% rpcinfo -T tcp klaxon prognum versnum
```

To show all RPC services registered with version 2 of the `rpcbind` protocol on the local machine use:

```
example% rpcinfo -p
```

To delete the registration for version 1 of the `walld` (program number `100008`) service for all transports use:

```
example# rpcinfo -d 100008 1
```

or

```
example# rpcinfo -d walld 1
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`rpcbind`(1M), `rpc`(3NSL), `netconfig`(4), `rpc`(4), `attributes`(5)

NAME

rpc.metad – remote metaset services

SYNOPSIS

/usr/sbin/rpc.metad

DESCRIPTION

`rpc.metad` is an `rpc`(4) daemon (functioning as a server process) that is used to manage local copies of metadevice diskset information. The `rpc.metad` daemon is invoked by `inetd`(1M).

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdu

SEE ALSO

`inetd`(1M), `metaset`(1M), `rpc.metamhd`(1M), `rpc`(3NSL), `services`(4), `attributes`(5)

Solaris Volume Manager Administration Guide

NAME

rpc.metamedd – remote mediator services

SYNOPSIS

/usr/sbin/rpc.metamedd

DESCRIPTION

`rpc.metamedd` is an `rpc`(4) server which is used to manage mediator information for use in 2–string HA configurations. The `rpc.metamedd` daemon is invoked by `inetd`(1M).

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdu
Interface Stability	Evolving

SEE ALSO

`inetd`(1M), `rpc`(4), `services`(4),

Sun Cluster documentation, *Solaris Volume Manager Administration Guide*

NAME

rpc.metamhd – remote multihost disk services

SYNOPSIS

/usr/sbin/rpc.metamhd

DESCRIPTION

`rpc.metamhd` is an `rpc`(4) daemon (functioning as a server process) that is used to manage multi-hosted disks. The `rpc.metamhd` daemon is invoked by `inetd`(1M).

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmdu

SEE ALSO

`inetd`(1M), `metaset`(1M), `rpc.metad`(1M), `rpc`(3NSL), `services`(4), `attributes`(5)

Solaris Volume Manager Administration Guide

NAME

rpc.nisd, nisd – NIS+ service daemon

SYNOPSIS

/usr/sbin/rpc.nisd [-ACDFhlv] [-Y [-B [-t *netid*]]] [-d *dictionary*] [-L *load*] [-S *level*] [-m *mappingfile*] [-x *attribute=value*]... [-z *number*]

DESCRIPTION

The `rpc.nisd` daemon is an RPC service that implements the NIS+ service. This daemon must be running on all machines that serve a portion of the NIS+ namespace.

`rpc.nisd` is usually started from a system startup script.

The `-B` option causes `rpc.nisd` to start an auxiliary process, `rpc.nisd_resolv`, which provides ypserv compatible DNS forwarding for NIS host requests. `rpc.nisd_resolv` can also be started independently. See `rpc.nisd_resolv`(1M) for more information on using `rpc.nisd_resolv` independently.

The `/etc/default/rpc.nisd` file contains the following default parameter settings. See `FILES`.

`ENABLE_NIS_YP_EMULATION` Specifies whether the server is put into NIS (YP) compatibility mode. `ENABLE_NIS_YP_EMULATION=YES` is equivalent to the `-Y` command-line option. The default value for `ENABLE_NIS_YP_EMULATION` is `NO`.

OPTIONS

`-A` Authentication verbose mode. The daemon logs all the authentication related activities to `syslogd`(1M) with `LOG_INFO` priority.

`-B` Provide ypserv compatible DNS forwarding for NIS host requests. The DNS resolving process, `rpc.nisd_resolv`, is started and controlled by `rpc.nisd`. This option requires that the `/etc/resolv.conf` file be setup for communication with a DNS nameserver. The `nslookup` utility can be used to verify communication with a DNS nameserver. See `resolv.conf`(4) and `nslookup`(1M).

`-C` Open diagnostic channel on `/dev/console`.

`-D` Debug mode. Do not fork.

`-d` *dictionary* Specify an alternate dictionary for the NIS+ database. The primary use of this option is for testing. Note that the string is not interpreted, rather it is simply passed to the `db_initialize` function.>

`-F` Force the server to do a checkpoint of the database when it starts up. Forced checkpoints may be required when the server is low on disk space. This option removes updates from the transaction log that have propagated to all of the replicas.

`-h` Print list of options.

`-L` *number*
: Specify the "load" the NIS+ service is allowed to place on the server. The load is specified in terms of the *number* of child processes that the server may spawn. The value of *number* must be at least 1 for the callback functions to work correctly. The default is 128.

`-m` *mappingfile*
: Specify the name of a configuration file that maps NIS+ objects (especially tables and columns) to LDAP (entries and attributes). See `NIS+LDAPmapping`(4). The default path is `/var/nis`. The default mapping file is `NIS+LDAPmapping`. If this file exists, the `rpc.nisd` daemon will map data to and from LDAP. A template mapping file that covers the normal NIS+ directories and tables is installed as `/var/nis/NIS+LDAPmapping.template`.

  A NIS+ object must have a valid mapping entry in the mapping file in order to have data for that table read from or written to the LDAP repository.

  The `rpc.nisd`(4) file contains specifications for LDAP server addresses, LDAP authentication method, and the like. See `NIS+LDAPmapping`(4) for an overview of the setup you need to map NIS+ data to or from LDAP.

`-S` *level*
: Set the authorization security level of the service. The argument is a number between 0 and 2. By default, the daemon runs at security level 2.

  `0`
  : Security level 0 is designed to be used for testing and initial setup of the NIS+ namespace. When running at level 0, the daemon does not enforce any access controls. Any client is allowed to perform any operation, including updates and deletions.

  `1`
  : At security level 1, the daemon accepts both `AUTH_SYS` and `AUTH_DES` credentials for authenticating clients and authorizing them to perform NIS+ operations. This is not a secure mode of operation since `AUTH_SYS` credentials are easily forged. It should not be used on networks in which any untrusted users may potentially have access.

  `2`
  : At security level 2, the daemon only accepts authentication using the security mechanisms configured by `nisauthconf`(1M). The default security mechanism is `AUTH_DES`. Security level 2 is the default if the `-S` option is not used.

`-t` *netid*
: Use *netid* as the transport for communication between `rpc.nisd` and `rpc.nisd_resolv`. The default transport is `ticots`(7D) (`tcp` on SunOS 4.x systems).

`-v` Verbose. With this option, the daemon sends a running narration of what it is doing to the syslog daemon (see `syslogd`(1M)) at `LOG_INFO` priority. This option is most useful for debugging problems with the service. See also `-A` option.

`-x` *attribute=value* Specify the value of the named *attribute*. Attributes that control the NIS+ to LDAP mapping operation are derived as follows:

1. Retrieve from LDAP.
2. Override with values from the *mappingfile*, if any. See the `-m` option.
3. Override with values from the command line `-x` options.

See `NIS+LDAPmapping`(4) and `rpc.nisd`(4) for the recognized attributes and their syntax.

As a special case, you can use the `nisplusLdapConfig*` attributes to derive additional information from LDAP. You can only specify the `nisplusLdapConfig*` attributes in `rpc.nisd`(4) or by means of the command line.

`-Y` Put the server into NIS (YP) compatibility mode. When operating in this mode, the NIS+ server will respond to NIS Version 2 requests using the version 2 protocol. Because the YP protocol is not authenticated, only those items that have read access to nobody (the unauthenticated request) will be visible through the V2 protocol. It supports only the standard Version 2 maps in this mode (see `-B` option and `NOTES` in `ypfiles`(4)). See `FILES`.

`-z` *number* Specify the maximum RPC record size that can be used over connection oriented transports. The default is 9000 bytes. If you specify a size less than the default value, the default value will be used instead.

EXAMPLES

EXAMPLE 1 Setting up the NIS+ Service

The following example sets up the NIS+ service.

```
example% rpc.nisd
```

EXAMPLE 2 Setting Up NIS+ Service Emulating YP With DNS Forwarding

The following example sets up the NIS+ service, emulating YP with DNS forwarding.

```
example% rpc.nisd -YB
```

EXAMPLE 3 Specifying NIS+ and LDAP Mapping Information

The following example shows how to specify that all additional NIS+ and LDAP mapping information should be retrieved from DN "dc=x,dc=y,dc=z", from the LDAP server at IP address 1.2.3.4, port 389. The examples uses the simple authentication method and the `cn=nisplusAdmin,ou=People,` proxy user. The `-m` option is omitted for clarity in this example..

```
-x nisplusLDAPconfigDN=dc=x,dc=y,dc=z \
-x nisplusLDAPconfigPreferredServerList=127.0.0.1:389 \
-x nisplusLDAPconfigAuthenticationMethod=simple \
-x nisplusLDAPconfigProxyUser=cn=nisplusAdmin,ou=People, \
-x nisplusLDAPconfigProxyPassword=xyzzy
```

ENVIRONMENT VARIABLES

`NETPATH` The transports that the NIS+ service will use can be limited by setting this environment variable. See `netconfig`(4).

FILES

`/var/nis/data/parent.object`	This file describes the namespace that is logically above the NIS+ namespace. The most common type of parent object is a DNS object. This object contains contact information for a server of that domain.
`/var/nis/data/root.object`	This file describes the root object of the NIS+ namespace. It is a standard XDR-encoded NIS+ directory object that can be modified by authorized clients using the `nis_modify`(3NSL) interface.
`/etc/init.d/rpc`	Initialization script for NIS+.
`/etc/default/rpc.nisd`	LDAP connection and general `rpc.nisd` configuration. You can override some of the settings by command-line options.
`/var/nis/NIS+LDAPmapping`	Default path for LDAP mapping file. See the discussion of the `-m` option.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`nis_cachemgr`(1M), `nisauthconf`(1M), `nisinit`(1M), `nissetup`(1M), `nisldapmaptest`(1M), `nslookup`(1M), `rpc.nisd_resolv`(1M), `rpc.nispasswdd`(1M), `syslogd`(1M), `nis_modify`(3NSL), `NIS+LDAPmapping`(4), `netconfig`(4), `nisfiles`(4), `resolv.conf`(4), `rpc.nisd`(4), `ypfiles`(4), `attributes`(5), `ticots`(7D)

NOTES NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

rpc.nisd_resolv, nisd_resolv – NIS+ service daemon

SYNOPSIS

rpc.nisd_resolv [-v | -V] [-F [-C *fd*]] [-t *xx*] [-p *yy*]

DESCRIPTION

rpc.nisd_resolv is an auxiliary process which provides DNS forwarding service for NIS hosts requests to both ypserv and rpc.nisd that are running in the NIS compatibility mode. It is generally started by invoking rpc.nisd(1M) with the -B option or ypserv(1M) with the -d option. Although it is not recommended, rpc.nisd_resolv can also be started independently with the following options.

This command requires that the /etc/resolv.conf file be setup for communication with a DNS nameserver. The nslookup utility can be used to verify communication with a DNS nameserver. See resolv.conf(4) and nslookup(1M).

OPTIONS

-F Run in foreground.

-C *fd* Use *fd* for service xprt (from nisd).

-v Verbose. Send output to the syslog daemon.

-V Verbose. Send output to stdout.

-t *xx* Use transport *xx*.

-p *yy* Use transient program# *yy*.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

nslookup(1M), rpc.nisd(1M), resolv.conf(4), attributes(5)

NOTES

NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

rpc.nispasswdd, nispasswdd – NIS+ password update daemon

SYNOPSIS

/usr/sbin/rpc.nispasswdd [-a *attempts*] [-c *minutes*] [-D] [-g] [-v]

DESCRIPTION

rpc.nispasswdd daemon is an ONC+ RPC service that services password update requests from nispasswd(1) and yppasswd(1). It updates password entries in the NIS+ passwd table.

rpc.nispasswdd is normally started from a system startup script after the NIS+ server (rpc.nisd(1M)) has been started. rpc.nispasswdd will determine whether it is running on a machine that is a master server for one or more NIS+ directories. If it discovers that the host is not a master server, then it will promptly exit. It will also determine if rpc.nisd(1M) is running in NIS (YP) compatibility mode (the -Yoption) and will register as yppasswdd for NIS (YP) clients as well.

rpc.nispasswdd will syslog all failed password update attempts, which will allow an administrator to determine whether someone was trying to "crack" the passwords.

rpc.nispasswdd has to be run by a superuser.

OPTIONS

-a *attempts*	Set the maximum number of attempts allowed to authenticate the caller within a password update request session. Failed attempts are syslogd(1M) and the request is cached by the daemon. After the maximum number of allowed attempts the daemon severs the connection to the client. The default value is set to 3.
-c *minutes*	Set the number of minutes a failed password update request should be cached by the daemon. This is the time during which if the daemon receives further password update requests for the same user and authentication of the caller fails, then the daemon will simply not respond. The default value is set to 30minutes.
-D	Debug. Run in debugging mode.
-g	Generate DES credential. By default the DES credential is not generated for the user if they do not have one. By specifying this option, if the user does not have a credential, then one will be generated for them and stored in the NIS+ cred table.
-v	Verbose. With this option, the daemon sends a running narration of what it is doing to the syslog daemon. This option is useful for debugging problems.

EXIT STATUS

0	success
1	an error has occurred.

FILES

/etc/init.d/rpc	initialization script for NIS+

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`nispasswd`(1), `passwd`(1), `yppasswd`(1), `rpc.nisd`(1M), `syslogd`(1M), `nsswitch.conf`(4), `attributes`(5)

NOTES

NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

rpc.rexd, rexd – RPC-based remote execution server

SYNOPSIS

/usr/sbin/rpc.rexd [-s]

DESCRIPTION

`rpc.rexd` is the Sun RPC server for remote program execution. This daemon is started by `inetd`(1M) whenever a remote execution request is made.

For non-interactive programs, the standard file descriptors are connected directly to TCP connections. Interactive programs involve pseudo-terminals, in a fashion that is similar to the login sessions provided by `rlogin`(1). This daemon may use NFS to mount file systems specified in the remote execution request.

OPTIONS

The following options are supported:

`-s` Secure. When specified, requests must have valid DES credentials. If the request does not have a DES credential it is rejected. The default publickey credential is rejected. Only newer `on`(1) commands send DES credentials.

If access is denied with an authentication error, you may have to set your publickey with the `chkey`(1) command.

Specifying the `-s` option without presenting secure credentials will result in an error message: `Unix too weak auth (DesONly)!`

SECURITY

`rpc.rexd` uses `pam`(3PAM) for account and session management. The PAM configuration policy, listed through `/etc/pam.conf`, specifies the modules to be used for `rpc.rexd`. Here is a partial `pam.conf` file with `rpc.rexd` entries for account and session management using the UNIX module.

```
rpc.rexd   account requisite       pam_roles.so.1
rpc.rexd   account required        pam_projects.so.1
rpc.rexd   account required        pam_unix_account.so.1

rpc.rexd      session required      pam_unix_session.so.1
```

If there are no entries for the `rpc.rexd` service, then the entries for the "other" service will be used. `rpc.rexd` uses the `getpwuid()` call to determine whether the given user is a legal user.

FILES

`/dev/pts`*n*	Pseudo-terminals used for interactive mode
`/etc/passwd`	Authorized users
`/tmp_rex/rexd??????`	Temporary mount points for remote file systems

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

chkey(1), on(1), rlogin(1), inetd(1M), pam(3PAM), inetd.conf(4), pam.conf(4), publickey(4), attributes(5), pam_authtok_check(5), pam_authtok_get(5), pam_authtok_store(5), pam_dhkeys(5), pam_passwd_auth(5), pam_unix(5), pam_unix_account(5), pam_unix_auth(5), pam_unix_session(5)

DIAGNOSTICS

Diagnostic messages are normally printed on the console, and returned to the requestor.

NOTES

Root cannot execute commands using rexd client programs such as on(1).

The pam_unix(5) module might not be supported in a future release. Similar functionality is provided by pam_authtok_check(5), pam_authtok_get(5), pam_authtok_store(5), pam_dhkeys(5), pam_passwd_auth(5), pam_unix_account(5), pam_unix_auth(5), and pam_unix_session(5).

NAME

rpc.rstatd, rstatd – kernel statistics server

SYNOPSIS

/usr/lib/netsvc/rstat/rpc.rstatd

DESCRIPTION

`rpc.rstatd` is a server which returns performance statistics obtained from the kernel. `rup`(1) uses `rpc.rstatd` to collect the `uptime` information that it displays.

`rpc.rstatd` is an RPC service.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWrcmds

SEE ALSO

`rup`(1), `inetd`(1M), `services`(4), `attributes`(5)

NAME

rpc.rusersd, rusersd – network username server

SYNOPSIS

/usr/lib/netsvc/rusers/rpc.rusersd

DESCRIPTION

`rpc.rusersd` is a server that returns a list of users on the host. The `rpc.rusersd` daemon may be started by `inetd`(1M) or `listen`(1M).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWrcmds

SEE ALSO

`inetd`(1M), `listen`(1M), `pmadm`(1M), `sacadm`(1M), `attributes`(5)

NAME

rpc.rwalld, rwalld – network rwall server

SYNOPSIS

/usr/lib/netsvc/rwall/rpc.rwalld

DESCRIPTION

`rpc.rwalld` is a server that handles `rwall`(1M) requests. It is implemented by calling `wall`(1M) on all the appropriate network machines. The `rpc.rwalld` daemon may be started by `inetd`(1M) or `listen`(1M).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWrcmds

SEE ALSO

`inetd`(1M), `listen`(1M), `rwall`(1M), `wall`(1M), `attributes`(5)

NAME

rpc.smserverd – removable media device server

SYNOPSIS

/usr/lib/smedia/rpc.smserverd

DESCRIPTION

`rpc.smserverd` is a server that handles requests from client applications, including the Volume Management daemon (`vold`(1M)), for access to removable media devices. In addition to `vold`, `rmformat`(1) and the CDE Filemanager (when performing removable media operations) are `rpc.smserverd` clients. The `rpc.smserverd` daemon is started by `inetd`(1M) when a client makes a call to a Solaris-internal library to access a SCSI, IDE, or USB device. The daemon is not started if a client attempts to access a floppy or PCMCIA device. Once started, the daemon remains active until such time as it is idle for three minutes or more.

The `rpc.smserverd` daemon is provided for the exclusive use of the client applications mentioned above. It has no external, customer-accessible interfaces, including no configuration file.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWvolu

SEE ALSO

`inetd`(1M), `vold`(1M), `vold.conf`(4), `attributes`(5)

NAME

rpc.sprayd, sprayd – spray server

SYNOPSIS

/usr/lib/netsvc/spray/rpc.sprayd

DESCRIPTION

`rpc.sprayd` is a server that records the packets sent by `spray`(1M). The `rpc.sprayd` daemon may be started by `inetd`(1M) or `listen`(1M).

The service provided by `rpc.sprayd` is not useful as a networking benchmark as it uses unreliable connectionless transports, (udp for example). It can report a large number of packets dropped when the drops were caused by the program sending packets faster than they can be buffered locally (before the packets get to the network medium).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWrcmds

SEE ALSO

`inetd`(1M) `listen`(1M), `pmadm`(1M), `sacadm`(1M), `spray`(1M), `attributes`(5)

NAME

rpc.yppasswdd, yppasswdd – server for modifying NIS password file

SYNOPSIS

```
/usr/lib/netsvc/yp/rpc.yppasswdd [-D directory] [-nogecos]
     [-noshell] [-nopw] [-m argument1 argument2...]

/usr/lib/netsvc/yp/rpc.yppasswdd [passwordfile [adjunctfile]] [-nogecos]
     [-noshell] [-nopw] [-m argument1 argument2...]
```

DESCRIPTION

`rpc.yppasswdd` is a server that handles password change requests from `yppasswd`(1). It changes a password entry in the `passwd`, `shadow`, and `security/passwd.adjunct` files. The `passwd` and `shadow` files provide the basis for the `passwd.byname` and `passwd.byuid` maps. The `passwd.adjunct` file provides the basis for the `passwd.adjunct.byname` and `passwd.adjunct.byuid` maps. Entries in the `passwd`, `shadow` or `passwd.adjunct` files are changed only if the password presented by `yppasswd`(1) matches the encrypted password of the entry. All password files are located in the `PWDIR` directory.

If the `-D` option is given, the `passwd`, `shadow`, or `passwd.adjunct` files are placed under the directory path that is the argument to `-D`.

If the `-noshell`, `-nogecos` or `-nopw` options are given, these fields cannot be changed remotely using `chfn`, `chsh`, or `passwd`(1).

If the `-m` option is given, a `make`(1) is performed in `/var/yp` after any of the `passwd`, `shadow`, or `passwd.adjunct` files are modified. All arguments following the flag are passed to `make`.

The second of the listed syntaxes is provided only for backward compatibility. If the second syntax is used, the `passwordfile` is the full pathname of the password file and `adjunctfile` is the full pathname of the optional `passwd.adjunct` file. If a shadow file is found in the same directory as `passwordfile`, the `shadowfile` is used as described above. Use of this syntax and the discovery of a `shadowfile` file generates diagnostic output. The daemon, however, starts normally.

The first and second syntaxes are mutually exclusive. You cannot specify the full pathname of the `passwd`, `passwd.adjunct` files and use the `-D` option at the same time.

The daemon is started automatically on the master server of the `passwd` map by `ypstart`(1M), which is invoked at boot time by the `/etc/init.d/rpc` script.

The server does not insist on the presence of a `shadow` file unless there is no `-D` option present or the directory named with the `-D` option is `/etc`. In addition, a `passwd.adjunct` file is not necessary. If the `-D` option is given, the server attempts to find a `passwd.adjunct` file in the `security` subdirectory of the named directory. For example, in the presence of "`-D /var/yp`" the server checks for a "`/var/yp/security/passwd.adjunct`" file.

If only a `passwd` file exists, then the encrypted password is expected in the second field. If both a `passwd` and a `passwd.adjunct` file exist, the encrypted password is expected in the second field of the adjunct file with *##username* in the second field of the `passwd` file. If all three files are in use, the encrypted password is expected in the `shadow` file. Any deviation causes a password update to fail.

If you remove or add a `shadow` or `passwd.adjunct` file after `rpc.yppasswdd` has started, you must stop and restart the daemon to enable it to recognize the change. See `ypstart`(1m) for information on restarting the daemon.

The `rpc.yppasswdd` daemon considers a shell that has a name that begins with 'r' to be a restricted shell. By default, the daemon does not check whether a shell begins with an 'r'. However, you can tell it to do so by uncommenting the "`check_restricted_shell_name=1`" line in `/etc/default/yppasswdd`. The result will be to restrict a user's ability to change from his default shell. See `yppasswdd`(4).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWypu

SEE ALSO

`make`(1), `passwd`(1), `yppasswd`(1), `inetd`(1M), `ypmake`(1M), `passwd`(4), `shadow`(4), `ypfiles`(4), `yppasswdd`(4), `attributes`(5)

NOTES

If `make` has not been installed and the `-m` option is given, the daemon outputs a warning and proceeds, effectively ignoring the `-m` flag.

When using the `-D` option, you should make sure that the `PWDIR` of the `/var/yp/Makefile` is set accordingly.

The second listed syntax is supplied only for backward compatibility and might be removed in a future release of this daemon.

The Network Information Service (NIS) was formerly known as Sun Yellow Pages (YP). The functionality of the two remains the same; only the name has changed. The name Yellow Pages is a registered trademark in the United Kingdom of British Telecommunications PLC, and cannot be used without permission.

NAME

rpc.ypupdated, ypupdated – server for changing NIS information

SYNOPSIS

/usr/lib/netsvc/yp/rpc.ypupdated [-is]

DESCRIPTION

ypupdated is a daemon that updates information in the Network Information Service (NIS). ypupdated consults the updaters(4) file in the /var/yp directory to determine which NIS maps should be updated and how to change them.

By default, the daemon requires the most secure method of authentication available to it, either DES (secure) or UNIX (insecure).

OPTIONS

-i Accept RPC calls with the insecure AUTH_UNIX credentials. This allows programmatic updating of the NIS maps in all networks.

-s Accept only calls authenticated using the secure RPC mechanism (AUTH_DES authentication). This disables programmatic updating of the NIS maps unless the network supports these calls.

FILES

/var/yp/updaters Configuration file for rpc.updated command.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWypu

SEE ALSO

keyserv(1M), updaters(4), attributes(5)

System Administration Guide, Volume I

NOTES

The Network Information Service (NIS) was formerly known as Sun Yellow Pages (YP). The functionality of the two services remains the same; only the name has changed. The name Yellow Pages is a registered trademark in the United Kingdom of British Telecommunications PLC, and must not be used without permission.

NAME

rpld – IA Network Booting RPL (Remote Program Load) Server

SYNOPSIS

/usr/sbin/rpld [-fdDMblgz] *interface*

/usr/sbin/rpld -a [-fdDMblgz]

DESCRIPTION

The RPL server provides network booting functionality to IA clients by listening to boot requests from them according to the RPL protocol specifications. Boot requests can be generated by clients using the boot floppy supplied in the IA distribution. Once the request has been received, the server validates the client and adds it to its internal service list. Subsequent requests from the client to download bootfiles will result in the sending of data frames from the server to the client specifying where to load the boot program in memory. When all the bootfiles have been downloaded, the server specifies where to start execution to initiate the boot process.

In the first synopsis, the interface parameter names the network interface upon which `rpld` is to listen for requests. For example:

```
/usr/sbin/rpld /dev/le0

/usr/sbin/rpld /dev/smc0
```

In the second synopsis, `rpld` locates all of the network interfaces present on the system and starts a daemon process for each one.

The server starts by reading the default configuration file, or an alternate configuration file if one is specified. If no configuration file can be found, internal default values will be used. Alternatively, command line options are available to override any of the values in the configuration file. After the configuration options are set, it then opens the network interface as specified in the command line and starts listening to RPL boot requests.

Network boot IA clients have to have information pre-configured on a server for the RPL server to validate and serve them. This involves putting configuration information in both the `ethers`(4) and the `bootparams`(4) databases. The `ethers` database contains a translation from the physical node address to the IP address of the clients and is normally used by the RARP server. The `bootparams` database stores all other information needed for booting off this client, such as the number of bootfiles and the file names of the various boot components. Both databases can be looked up by the RPL server through NIS. See the sub-section `Client Configuration` for information on how to set up these databases.

To assist in the administration and maintenance of the network boot activity, there are two run-time signals that the server will accept to change some run-time parameters and print out useful status information. See the sub-section `Signals` for details.

The RPL server is not limited to the ability to boot only IA clients. If properly configured, the server should be able to download any bootfiles to the clients.

Client Configuration

The following configuration information is specific to booting IA clients.

In order to allow clients to boot IA from across the network, the client's information has to be pre-configured in two databases: `ethers`(4) and `bootparams`(4). Both databases can be accessed through NIS. Refer to *Solaris 9 Installation Guide* for information on how to configure a diskless IA client. The discussion contained in the rest of this section is provided for your information only and should not be performed manually.

The `ethers` database contains a translation table to convert the physical node address to the IP address of the client. Therefore, an IP address must be assigned to the client (if this has not been done already), the node address of the client must be obtained, and then this information needs to be entered in the `ethers` database.

The bulk of the configuration is done in the `bootparams` database. This is a free-format database that essentially contains a number of keyword-value string pairs. A number of keywords have been defined for specific purposes, like the `bootparams` RPC in `bootparamd`(1M). Three more keywords have been defined for the RPL server. They are `numbootfiles`, `bootfile`, and `bootaddr`. All three keywords must be in lowercase letters with no spaces before or after the equals symbol following the keyword.

`numbootfiles` Specifies the number of files to be downloaded to the network boot client. The format of this option is:

```
numbootfiles=n
```

Always use `numbootfiles=3` to boot IA across the network.

`bootfile` Specifies the path name of the bootfile to be downloaded and where in memory to start loading the bootfile. A complete path name should be used. For example, assuming the client's IP address is 129.181.32.15:

```
bootfile=/rplboot/129.181.32.15.hw.com:45000
bootfile=/rplboot/129.181.32.15.glue.com:35000
bootfile=/rplboot/129.181.32.15.inetboot=8000
```

The path name following the equals symbol specifies the bootfile to be downloaded, and the hex address following the colon (:) is the absolute address of the memory location to start loading that bootfile. These addresses should be in the range of 7c00 to a0000 (i.e., the base 640K range excluding the interrupt vector and BIOS data areas). Address 45000 for this `hw.com` bootfile is also a suggested value and if possible should not be changed. The address of 35000 for `glue.com` is a suggested value that, if possible, should not be changed. The address of 8000 for `inetboot` is an absolute requirement and should never be changed.

These files, when created following the procedures in the *Solaris 9 Installation Guide* are actually symbolic links to to the real file to be downloaded to the client. `hw.com` is linked to a special driver that corresponds to the network interface card of the client. `glue.com` and `inetboot` are generic to all network boot clients.

The order of these bootfile lines is not significant, but because problems have been found with certain boot PROMs, it is highly recommended that the bootfile lines be ordered in descending order of the load addresses.

`bootaddr` The absolute address in memory to start executing after all the bootfiles have been downloaded. This address should always correspond to the address where `glue.com` is being loaded. If possible, always use:

```
bootaddr=35000
```

OPTIONS The following options are supported:

`-b` *background_mode* Specify `1` to run the server in the background and relinquish the controlling terminal, or `0` to run in the foreground without relinquishing the controlling terminal. This option corresponds to the *BackGround* setting in the configuration file. If you have specified that the error or warning messages be sent to standard output in the configuration file or by using the `-D` option above, the server cannot be run in background mode. Doing so will cause the server to exit after announcing the error.

`-d` *debug_level* Specify a level of `0` if you do not want any error or warning messages to be generated, or a level from `1` to `9` for increasing amounts of messages. This option corresponds to the *DebugLevel* setting in the configuration file. The default value is `0`. Note that it is best to limit the level to `8` or below; use of level `9` may generate so many debug messages that the performance of the RPL server may be impacted.

`-D` *debug_destination* Specify `0` to send error or warning messages to standard output, `1` to syslogd, and `2` to the log file. This option corresponds to the *DebugDest* setting in the configuration file. The default value is `2`.

`-f` *config_filename* Use this to specify a configuration file name other than the system default `/etc/rpld.conf` file.

`-g` *delay_granularity* This corresponds to the *DelayGran* setting in the configuration file. If retransmission requests from clients do occur, the delay granularity factor will be used to adjust the delay count for this client upwards

	or downwards. If the retransmission request is caused by data overrun, the delay count will be incremented by delay granularity units to increase the delay between data frames. If the retransmission request is caused by sending data too slowly, this will be used to adjust the delay count downwards to shorten the delay. Eventually the server will settle at the delay count value that works best with the speed of the client and no retransmission request will be needed. The default value is `2`.
`-l` *log_filename*	Specify an alternate log file name to hold the error or warning messages in connection with the `-D 2` option or the configuration file *DebugDest = 2* setting. This option corresponds to the *LogFile* setting in the configuration file. The default is `/var/spool/rpld.log`.
`-M` *maximum_clients*	Specify the maximum number of simultaneous network boot clients to be served. This option corresponds to the *MaxClients* setting in the configuration file. A value of `–1` means unlimited, and the actual number will depend on available system resources. The default value is `–1`.
`-s` *start_delay_count*	This option corresponds to the *StartDelay* setting in the configuration file. Specify the number of delay units between outgoing data frames sent to clients to avoid retransmission requests from them. Using the LLC type `1` protocol, data transfer is a one-way, best-effort delivery mechanism. The server, without any type of delay mechanism, can overrun the client by sending data frames too quickly. Therefore, a variable delay is built into the server to limit the speed of sending data to the clients, thus avoiding the clients sending back retransmission requests. This value should be machine environment specific. If you have a fast server machine but slow client machines, you may want to set a large start delay count. If you have comparable server and client machines, the delay count may be set to `1`. The delay is only approximate and should not be taken as an accurate measure of time. There is no specific correlation between the delay unit and the actual time of delay. The default value is `20`.
`-z` *frame_size*	This option corresponds to the *FrameSize* setting in the configuration file. This specifies the size of the data frames used to send data to the clients. This is limited by the underlying physical medium. For

`ethernet/802.3`, the maximum physical frame size is 1500 octets. The default value is `1500`. Note that the protocol overhead of LLC1 and RPL is 32 octets, resulting in a maximum data length of 1468 octets.

Signals

The RPL server accepts two signals to change run-time parameters and display status information, respectively:

`HANGUP` This will cause the RPL server to reread the default configuration file `/etc/rpld.conf` or an alternate configuration file if one is specified when the server is started. New values of certain parameters can be used immediately, such as *DebugLevel*, *DebugDest*, *LogFile*, *DelayGran*, and *FrameSize*. For *MaxClients*, if the server is already serving more than the new value, the server will not accept additional boot requests until the number has fallen below the *MaxClients* parameter. For *StartDelay*, this will only affect new boot requests. All the existing delay counts for the various clients in service will not be affected. Finally, the *BackGround* parameter will have no effect once the server has been running. You cannot change the mode of service without first killing the server and then restarting it.

`USR1` This signal will cause the server to dump all the parameter values and the status of each individual boot client to the destination specified by *DebugDest*.

FILES

`/usr/sbin/rpld`

`/etc/rpld.conf`

`/var/spool/rpld.log`

`/etc/ethers`

`/etc/bootparams`

`/rplboot`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Architecture	IA
Availability	SUNWbsu

SEE ALSO

`bootparamd`(1M), `in.rarpd`(1M), `bootparams`(4), `ethers`(4), `nsswitch.conf`(4), `rpld.conf`(4), `attributes`(5)

Solaris 9 Installation Guide

NAME

rquotad – remote quota server

SYNOPSIS

/usr/lib/nfs/rquotad

DESCRIPTION

`rquotad` is an `rpc`(4) server which returns quotas for a user of a local file system which is mounted by a remote machine over the NFS. The results are used by `quota`(1M) to display user quotas for remote file systems. The `rquotad` daemon is normally invoked by `inetd`(1M).

FILES

`quotas` quota file at the file system root

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnfssu

SEE ALSO

`inetd`(1M), `quota`(1M), `rpc`(4), `services`(4), `attributes`(5)

Solaris 9 Installation Guide

NAME

rsh, restricted_shell – restricted shell command interpreter

SYNOPSIS

/usr/lib/rsh [-acefhiknprstuvx] [*argument*...]

DESCRIPTION

rsh is a limiting version of the standard command interpreter sh, used to restrict logins to execution environments whose capabilities are more controlled than those of sh (see sh(1) for complete description and usage).

When the shell is invoked, it scans the environment for the value of the environmental variable, SHELL. If it is found and rsh is the file name part of its value, the shell becomes a restricted shell.

The actions of rsh are identical to those of sh, except that the following are disallowed:

- changing directory (see cd(1)),
- setting the value of $PATH,
- pecifying path or command names containing /,
- redirecting output (> and >>).

The restrictions above are enforced after *.profile* is interpreted.

A restricted shell can be invoked in one of the following ways:

1. rsh is the file name part of the last entry in the /etc/passwd file (see passwd(4));
2. the environment variable SHELL exists and rsh is the file name part of its value; the environment variable SHELL needs to be set in the .login file;
3. the shell is invoked and rsh is the file name part of argument 0;
4. the shell is invoke with the -r option.

When a command to be executed is found to be a shell procedure, rsh invokes sh to execute it. Thus, it is possible to provide to the end-user shell procedures that have access to the full power of the standard shell, while imposing a limited menu of commands; this scheme assumes that the end-user does not have write and execute permissions in the same directory.

The net effect of these rules is that the writer of the *.profile* (see profile(4)) has complete control over user actions by performing guaranteed setup actions and leaving the user in an appropriate directory (probably *not* the login directory).

The system administrator often sets up a directory of commands (that is, /usr/rbin) that can be safely invoked by a restricted shell. Some systems also provide a restricted editor, red.

EXIT STATUS

Errors detected by the shell, such as syntax errors, cause the shell to return a non-zero exit status. If the shell is being used non-interactively execution of the shell file is abandoned. Otherwise, the shell returns the exit status of the last command executed.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO `intro`(1), `cd`(1), `login`(1), `rsh`(1), `sh`(1), `exec`(2), `passwd`(4), `profile`(4), `attributes`(5)

NOTES The restricted shell, `/usr/lib/rsh`, should not be confused with the remote shell, `/usr/bin/rsh`, which is documented in `rsh`(1).

NAME

rtc – provide all real-time clock and GMT-lag management

SYNOPSIS

/usr/sbin/rtc [-c] [-z *zone-name*]

DESCRIPTION

The rtc command reconciles the difference in the way that time is established between UNIX and MS-DOS systems. UNIX systems utilize Greenwich Mean Time (GMT), while MS-DOS systems utilize local time.

Without arguments, rtc displays the currently configured time zone string. The currently configured time zone string is based on what was last recorded by rtc-z *zone-name*.

The rtc command is not normally run from a shell prompt; it is generally invoked by the system. Commands such as date(1) and rdate(1M), which are used to set the time on a system, invoke /usr/sbin/rtc -c to ensure that daylight savings time (DST) is corrected for properly.

OPTIONS

-c — This option checks for DST and makes corrections if necessary. It is normally run once a day by a cron job.

If there is no RTC time zone or /etc/rtc_config file, this option will do nothing.

-z *zone-name* — This option, which is normally run by the system at software installation time, is used to specify the time zone in which the RTC is to be maintained. It updates the configuration file /etc/rtc_config with the name of the specified zone and the current GMT lag for that zone. If there is an existing rtc_config file, this command will update it. If not, this command will create it.

FILES

/etc/rtc_config — The data file used to record the time zone and GMT lag. This file is completely managed by /usr/sbin/rtc, and it is read by the kernel.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Architecture	IA
Availability	SUNWcsu

SEE ALSO

date(1), rdate(1M), attributes(5)

NAME

runacct – run daily accounting

SYNOPSIS

/usr/lib/acct/runacct [*mmdd* [*state*]]

DESCRIPTION

`runacct` is the main daily accounting shell procedure. It is normally initiated using `cron`. `runacct` processes connect, fee, disk, and process accounting files. It also prepares summary files for `prdaily` or billing purposes. `runacct` is distributed only to source code licensees.

`runacct` takes care not to damage active accounting files or summary files in the event of errors. It records its progress by writing descriptive diagnostic messages into `active`. When an error is detected, a message is written to `/dev/console`, mail (see `mail`(1)) is sent to `root` and `adm`, and `runacct` terminates. `runacct` uses a series of lock files to protect against re-invocation. The files `lock` and `lock1` are used to prevent simultaneous invocation, and `lastdate` is used to prevent more than one invocation per day.

`runacct` breaks its processing into separate, restartable *states* using `statefile` to remember the last *state* completed. It accomplishes this by writing the *state* name into `statefile`. `runacct` then looks in `statefile` to see what it has done and to determine what to process next. *states* are executed in the following order:

`SETUP`	Move active accounting files into working files.
`WTMPFIX`	Verify integrity of `wtmpx` file, correcting date changes if necessary.
`CONNECT`	Produce connect session records in `tacct.h` format.
`PROCESS`	Convert process accounting records into `tacct.h` format.
`MERGE`	Merge the connect and process accounting records.
`FEES`	Convert output of `chargefee` into `tacct.h` format, merge with connect, and process accounting records.
`DISK`	Merge disk accounting records with connect, process, and fee accounting records.
`MERGETACCT`	Merge the daily total accounting records in `daytacct` with the summary total accounting records in `/var/adm/acct/sum/tacct`.
`CMS`	Produce command summaries.
`USEREXIT`	Any installation dependent accounting programs can be included here.
`CLEANUP`	Clean up temporary files and exit. To restart `runacct` after a failure, first check the `active` file for diagnostics, then fix any corrupted data files, such as `pacct` or `wtmpx`. The `lock`, `lock1`, and `lastdate` files must be removed before `runacct` can be restarted. The argument *mmdd* is necessary if `runacct` is being restarted. *mmdd* specifies the month and day for which `runacct` will rerun the accounting. The entry point for processing is based

on the contents of `statefile`; to override this, include the desired *state* on the command line to designate where processing should begin.

EXAMPLES

EXAMPLE 1 Starting `runacct`

The following example starts `runacct`:

```
example% nohup runacct 2> /var/adm/acct/nite/fd2log &
```

EXAMPLE 2 Restarting `runacct`

The following example restarts `runacct`:

```
example% nohup runacct 0601 2>> /var/adm/acct/nite/fd2log &
```

EXAMPLE 3 Restarting `runacct` at a Specific State

The following example restarts `runacct` at a specific state:

```
example% nohup runacct 0601 MERGE 2>> /var/adm/acct/nite/fd2log &
```

FILES

`/var/adm/wtmpx`
History of user access and administration information

`/var/adm/pacct`*incr*

`/var/adm/acct/nite/active`

`/var/adm/acct/nite/daytacct`

`/var/adm/acct/nite/lock`

`/var/adm/acct/nite/lock1`

`/var/adm/acct/nite/lastdate`

`/var/adm/acct/nite/statefile`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWaccu

SEE ALSO

`acctcom`(1), `mail`(1), `acct`(1M), `acctcms`(1M), `acctcon`(1M), `acctmerg`(1M), `acctprc`(1M), `acctsh`(1M), `cron`(1M), `fwtmp`(1M), `acct`(2), `acct`(3HEAD), `utmpx`(4), `attributes`(5)

NOTES

It is not recommended to restart `runacct` in the SETUP *state*. Run SETUP manually and restart using:

`runacct` *mmdd* WTMPFIX

If `runacct` failed in the PROCESS *state,* remove the last `ptacct` file because it will not be complete.

The `runacct` command can process a maximum of

- 6000 distinct sessions
- 1000 distinct terminal lines
- 2000 distinct login names

during a single invocation of the command. If at some point the actual number of any one of these items exceeds the maximum, the command will not succeed.

Do not invoke `runacct` at the same time as `ckpacct`, as there may be a conflict if both scripts attempt to execute `turnacct switch` simultaneously.

NAME

rwall – write to all users over a network

SYNOPSIS

/usr/sbin/rwall *hostname...*

/usr/sbin/rwall -n *netgroup...*

/usr/sbin/rwall -h *hostname* -n *netgroup*

DESCRIPTION

`rwall` reads a message from standard input until EOF. It then sends this message, preceded by the line:

```
Broadcast Message . . .
```

to all users logged in on the specified host machines. With the `-n` option, it sends to the specified network groups.

OPTIONS

`-n` *netgroup* Send the broadcast message to the specified network groups.

`-h` *hostname* Specify the `hostname`, the name of the host machine.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWrcmdc

SEE ALSO

`inetd`(1M), `listen`(1M), `pmadm`(1M), `sacadm`(1M), `wall`(1M), `attributes`(5)

NOTES

The timeout is fairly short to allow transmission to a large group of machines (some of which may be down) in a reasonable amount of time. Thus the message may not get through to a heavily loaded machine.

NAME

sac – service access controller

SYNOPSIS

sac -t *sanity_interval*

/usr/lib/saf/sac

DESCRIPTION

The Service Access Controller (SAC) is the overseer of the server machine. It is started when the server machine enters multiuser mode. The SAC performs several important functions as explained below.

Customizing the SAC Environment

When sac is invoked, it first looks for the per-system configuration script /etc/saf/_sysconfig. sac interprets _sysconfig to customize its own environment. The modifications made to the SAC environment by _sysconfig are inherited by all the children of the SAC. This inherited environment may be modified by the children.

Starting Port Monitors

After it has interpreted the _sysconfig file, the sac reads its administrative file /etc/saf/_sactab. _sactab specifies which port monitors are to be started. For each port monitor to be started, sac forks a child (see fork(2)) and creates a utmpx entry with the type field set to LOGIN_PROCESS. Each child then interprets its per-port monitor configuration script /etc/saf/*pmtag*/_config , if the file exists. These modifications to the environment affect the port monitor and will be inherited by all its children. Finally, the child process execs the port monitor, using the command found in the _sactab entry. (See sacadm; this is the command given with the -c option when the port monitor is added to the system.)

Polling Port Monitors to Detect Failure

The -t option sets the frequency with which sac polls the port monitors on the system. This time may also be thought of as half of the maximum latency required to detect that a port monitor has failed and that recovery action is necessary.

Administrative functions

The Service Access Controller represents the administrative point of control for port monitors. Its administrative tasks are explained below.

When queried (sacadm with either -l or -L), the Service Access Controller returns the status of the port monitors specified, which sacadm prints on the standard output. A port monitor may be in one of six states:

ENABLED	The port monitor is currently running and is accepting connections. See sacadm(1M) with the -e option.
DISABLED	The port monitor is currently running and is not accepting connections. See sacadm with the -d option, and see NOTRUNNING, below.
STARTING	The port monitor is in the process of starting up. STARTING is an intermediate state on the way to ENABLED or DISABLED.
FAILED	The port monitor was unable to start and remain running.
STOPPING	The port monitor has been manually terminated but has not completed its shutdown procedure. STOPPING is an intermediate state on the way to NOTRUNNING.

`NOTRUNNING` The port monitor is not currently running. (See `sacadm` with `-k`.) This is the normal "not running" state. When a port monitor is killed, all ports it was monitoring are inaccessible. It is not possible for an external user to tell whether a port is not being monitored or the system is down. If the port monitor is not killed but is in the `DISABLED` state, it may be possible (depending on the port monitor being used) to write a message on the inaccessible port telling the user who is trying to access the port that it is disabled. This is the advantage of having a `DISABLED` state as well as the `NOTRUNNING` state.

When a port monitor terminates, the SAC removes the `utmpx` entry for that port monitor.

The SAC receives all requests to enable, disable, start, or stop port monitors and takes the appropriate action.

The SAC is responsible for restarting port monitors that terminate. Whether or not the SAC will restart a given port monitor depends on two things:

- The restart count specified for the port monitor when the port monitor was added by `sacadm`; this information is included in `/etc/saf/`*pmtag*`/_sactab`.
- The number of times the port monitor has already been restarted.

SECURITY

`sac` uses `pam`(3PAM) for session management. The PAM configuration policy, listed through `/etc/pam.conf`, specifies the session management module to be used for `sac`. Here is a partial `pam.conf` file with entries for `sac` using the UNIX session management module.

```
sac  session   required   pam_unix_session.so.1
```

If there are no entries for the `sac` service, then the entries for the "other" service will be used.

OPTIONS

`-t` *sanity_interval* Sets the frequency (*sanity_interval*) with which `sac` polls the port monitors on the system.

FILES

`/etc/saf/_sactab`

`/etc/saf/_sysconfig`

`/var/adm/utmpx`

`/var/saf/_log`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO | pmadm(1M), sacadm(1M), fork(2) pam(3PAM), pam.conf(4), attributes(5), pam_authtok_check(5), pam_authtok_get(5), pam_authtok_store(5), pam_dhkeys(5), pam_passwd_auth(5), pam_unix(5), pam_unix_account(5), pam_unix_auth(5), pam_unix_session(5)

NOTES | The pam_unix(5) module might not be supported in a future release. Similar functionality is provided by pam_authtok_check(5), pam_authtok_get(5), pam_authtok_store(5), pam_dhkeys(5), pam_passwd_auth(5), pam_unix_account(5), pam_unix_auth(5), and pam_unix_session(5).

NAME

sacadm – service access controller administration

SYNOPSIS

sacadm -a -p *pmtag* -t *type* -c *cmd* -v *ver* [-f dx] [-n *count*] [-y *comment*] [-z *script*]

sacadm -r -p *pmtag*

sacadm -s -p *pmtag*

sacadm -k -p *pmtag*

sacadm -e -p *pmtag*

sacadm -d -p *pmtag*

sacadm -l [-p *pmtag* | -t *type*]

sacadm -L [-p *pmtag* | -t *type*]

sacadm -g -p *pmtag* [-z *script*]

sacadm -G [-z *script*]

sacadm -x [-p *pmtag*]

DESCRIPTION

`sacadm` is the administrative command for the upper level of the Service Access Facility hierarchy (port monitor administration). `sacadm` performs the following functions:

- adds or removes a port monitor
- starts or stops a port monitor
- enables or disables a port monitor
- installs or replaces a per-system configuration script
- installs or replaces a per-port monitor configuration script
- prints requested port monitor information

Requests about the status of port monitors (`-l` and `-L`) and requests to print per-port monitor and per-system configuration scripts (`-g` and `-G` without the `-z` option) may be executed by any user on the system. Other `sacadm` commands may be executed only by the super-user.

OPTIONS

`-a` Add a port monitor. When adding a port monitor, `sacadm` creates the supporting directory structure in `/etc/saf` and `/var/saf` and adds an entry for the new port monitor to `/etc/saf/_sactab`. The file `_sactab` already exists on the delivered system. Initially, it is empty except for a single line, which contains the version number of the Service Access Controller. Unless the command line that adds the new port monitor includes the `-f` option with the `-x` argument, the new port monitor will be started. Because of the complexity of the options and arguments that follow the `- a` option, it may be convenient to use a command script or the menu system to add port monitors.

`-c` *cmd* — Execute the command string *cmd* to start a port monitor. The `-c` option may be used only with a `-a`. A `-a` option requires a `-c`.

`-d` — Disable the port monitor *pmtag*.

`-e` — Enable the port monitor *pmtag*.

`-f dx` — The `-f` option specifies one or both of the following two flags which are then included in the flags field of the `_sactab` entry for the new port monitor. If the `-f` option is not included on the command line, no flags are set and the default conditions prevail. By default, a port monitor is started. A `-f` option with no following argument is illegal.

- d — Do not enable the new port monitor.
- x — Do not start the new port monitor.

`-g` — The `-g` option is used to request output or to install or replace the per-port monitor configuration script `/etc/saf/`*pmtag*`/_config`. `-g` requires a `-p` option. The `-g` option with only a `-p` option prints the per-port monitor configuration script for port monitor *pmtag*. The `-g` option with a `-p` option and a `-z` option installs the file `script` as the per-port monitor configuration script for port monitor *pmtag*. Other combinations of options with `-g` are invalid.

`-G` — The `-G` option is used to request output or to install or replace the per-system configuration script `/etc/saf/_sysconfig`. The `-G` option by itself prints the per-system configuration script. The `-G` option in combination with a `-z` option installs the file `script` as the per-system configuration script. Other combinations of options with a `-G` option are invalid.

`-k` — Stop port monitor *pmtag*.

`-l` — The `-l` option is used to request port monitor information. The `-l` by itself lists all port monitors on the system. The `-l` option in combination with the `-p` option lists only the port monitor specified by *pmtag*. A `-l` in combination with the `-t` option lists all port monitors of type `type`. Any other combination of options with the `-l` option is invalid.

`-L` — The `-L` option is identical to the `-l` option except that the output appears in a condensed format.

`-n` *count* — Set the restart count to *count*. If a restart count is not specified, count is set to `0`. A count of `0` indicates that the port monitor is not to be restarted if it fails.

`-p` *pmtag* — Specifies the tag associated with a port monitor.

`-r` — Remove port monitor *pmtag*. `sacadm` removes the port monitor entry from `/etc/saf/_sactab`. If the removed port monitor is

not running, then no further action is taken. If the removed port monitor is running, the Service Access Controller (SAC) sends it `SIGTERM` to indicate that it should shut down. Note that the port monitor's directory structure remains intact.

`-s` Start a port monitor. The SAC starts the port monitor *pmtag*.

`-t type` Specifies the port monitor type.

`-v ver` Specifies the version number of the port monitor. This version number may be given as

`-v '`*pmspec* `-V'`

where *pmspec* is the special administrative command for port monitor *pmtag*. This special command is `ttyadm` for `ttymon` and `nlsadmin` for `listen`. The version stamp of the port monitor is known by the command and is returned when *pmspec* is invoked with a `-V` option.

`-x` The `-x` option by itself tells the SAC to read its database file (`_sactab`). The `-x` option with the `-p` option tells port monitor *pmtag* to read its administrative file.

`-y` *comment* Include *comment* in the `_sactab` entry for port monitor *pmtag*.

`-z script` Used with the `-g` and `-G` options to specify the name of a file that contains a configuration script. With the `-g` option, `script` is a per-port monitor configuration script; with `-G` it is a per-system configuration script. Modifying a configuration script is a three-step procedure. First a copy of the existing script is made (`-g` or `-G`). Then the copy is edited. Finally, the copy is put in place over the existing script (`-g` or `-G` with `-z`).

OUTPUT

If successful, `sacadm` will exit with a status of `0`. If `sacadm` fails for any reason, it will exit with a nonzero status. Options that request information will write the information on the standard output. In the condensed format (`-L`), port monitor information is printed as a sequence of colon-separated fields; empty fields are indicated by two successive colons. The standard format (`-l`) prints a header identifying the columns, and port monitor information is aligned under the appropriate headings. In this format, an empty field is indicated by a hyphen. The comment character is `#`.

EXAMPLES

EXAMPLE 1 A sample output of the `sacadm` command.

The following command line adds a port monitor. The port monitor tag is `npack`; its type is `listen`; if necessary, it will restart three times before failing; its administrative command is `nlsadmin`; and the configuration script to be read is in the file `script`:

```
sacadm -a -p npack -t listen -c /usr/lib/saf/listen npack
-v 'nlsadmin -V' -n 3 -z script
```

Remove a port monitor whose tag is `pmtag`:

```
sacadm -r -p pmtag
```

EXAMPLE 1 A sample output of the `sacadm` command. *(Continued)*

Start the port monitor whose tag is `pmtag`:

```
sacadm -s -p pmtag
```

Stop the port monitor whose tag is `pmtag`:

```
sacadm -k -p pmtag
```

Enable the port monitor whose tag is `pmtag`:

```
sacadm -e -p pmtag
```

Disable the port monitor whose tag is `pmtag`:

```
sacadm -d -p pmtag
```

List status information for all port monitors:

```
sacadm -l
```

List status information for the port monitor whose tag is `pmtag`:

```
sacadm -l -p pmtag
```

List the same information in condensed format:

```
sacadm -L -p pmtag
```

List status information for all port monitors whose type is `listen`:

```
sacadm -l -t listen
```

Replace the per-port monitor configuration script associated with the port monitor whose tag is `pmtag` with the contents of the file `file.config`:

```
sacadm -g -p pmtag -z file.config
```

FILES

`/etc/saf/_sactab`

`/etc/saf/_sysconfig`

`/etc/saf/`*`pmtag`*`/_config`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`pmadm`(1M), `sac`(1M), `doconfig`(3NSL), `attributes`(5)

NAME

sadmind – distributed system administration daemon

SYNOPSIS

```
sadmind [-c keywords] [-i secs] [-l [logfile]] [-O OW_path_name]
     [-S security_level] [-v]
```

DESCRIPTION

`sadmind` is the daemon used by `Solstice AdminSuite` applications to perform distributed system administration operations.

The `sadmind` daemon is started automatically by the `inetd` daemon whenever a request to invoke an operation is received. The `sadmind` daemon process continues to run for 15 minutes after the last request is completed, unless a different idle-time is specified with the `-i` command line option. The `sadmind` daemon may be started independently from the command line, for example, at system boot time. In this case, the `-i` option has no effect; `sadmind` continues to run, even if there are no active requests.

The `sadmind` daemon process can be configured to write tracing information into a log file by specifying the `-c` and `-l` command line options. The `-c` option specifies a comma-separated list of keywords indicating the types of information to be logged. The following keywords may be useful to administrators:

`Errors`	Includes messages about errors that occurred during the daemon execution.
`Requests`	Includes messages about which operations `sadmind` invoked and when.
`System-Info`	Includes messages about when the `sadmind` daemon was started and stopped.
`*`	Includes all possible log messages.

The `-l` option enables logging and optionally specifies the path and file name of the log file. If no log file is specified, the default log file `/var/adm/admin.log` is used.

OPTIONS

The following options are supported:

`-c` *keywords*	Specify the types of information to be logged as a comma-separated list of keywords. The default is to log all types of messages.
`-i` *secs*	Specify the number of seconds for `sadmind` to stay up after the last request is completed. The default is 15 minutes (900 seconds). If *secs* is 0 or over 10,000,000, `sadmind` stays up forever. `-i` only applies when `sadmind` is started by the `inetd` daemon. You may want `sadmind` to run permanently (or for extended durations) on systems that are frequently administered by applications using `sadmind` (for example, a server managed through `Host Manager`) to improve application performance.

`-l` [*logfile*]
: Enable logging and optionally define the path name to the distributed system administration log file. The default log file is: `/var/adm/admin.log`

`-O` *OW_path_name*
: Define the path name to the OpenWindows home directory. If this option is not specified, the `sadmind` daemon will use the OpenWindows home directory defined in the `OPENWINHOME` environment variable, if defined; the home directory specified in the `/etc/OPENWINHOME` file, if it exists; or the default directory `/usr/openwin`. When the `sadmind` daemon is started by the `inetd` daemon, the environment variable `OPENWINHOME` is typically not defined. If the OpenWindows home directory is not one of the path names specified (`/usr/openwin` or in the file `/etc/OPENWINHOME)`, the `-O` option must be added to the `sadmind` entry in the `inetd.conf`(4) configuration file.

`-S` *security_level*
: Define the level of security to be used by the `sadmind` daemon when checking a client's right to perform an operation on the server system. Security level specifies the authentication mechanism used to provide and check the client's identity. The client's identity must be authenticated by the specified mechanism for `sadmind` to accept his or her request. The system-wide authentication requirements set by the security level may take precedence over any operation-specific requirements. Consequently, the security level can be used system-wide to ensure that all operations meet minimum authentication requirements, regardless of the requirements assigned specifically to an operation. In addition, the security level determines whether `sadmind` will perform authorization access control checking.

  Security level may be one of the following:

  `0`
  : Set authentication type to `NONE`. All clients' user and group identities are set to the `nobody` identity by `sadmind` (see *Solstice AdminSuite 2.1 User's Guide*). If access is granted to `nobody`, `sadmind` executes the operation. Use this level only for testing.

  `1`
  : Set authentication type to `WEAK`. Clients' user and group identities are set by `sadmind` from their authentication credentials. Client identities are accepted by

sadmind when they have satisfied either AUTH_SYS or AUTH_DES authentication mechanisms. The authenticated client identity is checked by sadmind for authorization to execute the operation. If an operation calls for a stronger security level, sadmind demotes the user identity to nobody, and then checks whether nobody is authorized to execute the operation. Since AUTH_SYS client credentials are easily forged, this level should be used only in relatively secure environments. No check is done that the user ID of the client represents the same user on the server system as on the client system. It is assumed that user and group identities are set up consistently on the network. This security level is the default.

2 — Set authentication type to STRONG. Clients' user and group identities are set by sadmind from their authentication credential mappings (effectively, user and group IDs from netid.byname for NIS, or cred table for NIS+). Client identities are accepted by sadmind only when they have satisfied the AUTH_DES authentication mechanism. The sadmind daemon checks whether the client identity is authorized to execute the operation. This level provides the most secure environment for executing distributed administration operations. It overrides any weaker level specific to an operation. A DES credential must exist for the host running the sadmind daemon and all administration client user identities.

-v — Enable the writing of log messages to the system logger, syslogd. Messages logged include fatal errors encountered while attempting to start the sadmind daemon process and those specified by the -c trace message keywords.

EXAMPLES

EXAMPLE 1 Using the sadmind command

By default, the line in /etc/inetd.conf that starts sadmind appears as follows:

```
100232/10        tli     rpc/udp    wait root
/usr/sbin/sadmind sadmind
```

EXAMPLE 1 Using the `sadmind` command *(Continued)*

To make a network as secure as possible, change the line to:

```
100232/10        tli     rpc/udp    wait root
/usr/sbin/sadmind sadmind -S 2
```

To minimize delays due to starting up `sadmind`, change the line to include the `-i` option:

```
100232/10        tli     rpc/udp    wait root
/usr/sbin/sadmind sadmind -i 86400
```

In this example, the duration that `sadmind` remains up after the last operation request was completed is extended to 24 hours (86,400 seconds). Extending the timeout period may enhance performance on servers and workstations that frequently run or are administered by applications that use the `sadmind` daemon (for example, `Solstice AdminSuite` applications such as `Host Manager`).

FILES

`/var/adm/admin.log`	Distributed system administration default log file
`/etc/inetd.conf`	Internet servers database file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWadmfw

SEE ALSO

`inetd`(1M), `rpcbind`(1M), `inetd.conf`(4), `attributes`(5)

Solstice AdminSuite 2.1 User's Guide

NOTES

Whenever `inetd` fails to start `sadmind`, re-register the RPC number for `sadmind`, 100232, with `rpcbind` by sending the `inetd` process a `SIGHUP` signal:

```
example%  kill -HUP pid
```

or

```
example%  kill –1
```

Sometimes `inetd` does not start `sadmind` in response to system administration requests, even though the `inetd.conf` file has the correct entry for the `sadmind` daemon. This can happen when `sadmind` is started manually from the command line and takes over the previous registration of the `sadmind` RPC number, 100232, by `inetd`. When the manually-started `sadmind` daemon is terminated, the `sadmind` RPC number, 100232, is de-registered with `rpcbind`. Consequently, system administration requests are ignored by `inetd`.

NAME

saf – Service Access Facility

DESCRIPTION

The SAF generalizes the procedures for service access so that login access on the local system and network access to local services are managed in similar ways. Under the SAF, systems may access services using a variety of port monitors, including ttymon, the listener, and port monitors written expressly for a user's application. The manner in which a port monitor observes and manages access ports is specific to the port monitor and not to any component of the SAF. Users may therefore extend their systems by developing and installing their own port monitors. One of the important features of the SAF is that it can be extended in this way by users.

Relative to the SAF, a service is a process that is started. There are no restrictions on the functions a service may provide. The SAF consists of a controlling process, the service access controller (SAC), and two administrative levels corresponding to two levels in the supporting directory structure. The top administrative level is concerned with port monitor administration, the lower level with service administration. The SAC is documented in the `sac`(1M) man page. The administrative levels and associated utilities are documented in the *System Administration Guide - Volume II*. The requirements for writing port monitors and the functions a port monitor must perform to run under the SAF and the SAC are documented here.

Port Monitors

A port monitor is a process that is responsible for monitoring a set of homogeneous, incoming ports on a machine. A port monitor's major purpose is to detect incoming service requests and to dispatch them appropriately.

A port is an externally seen access point on a system. A port may be an address on a network (TSAP or PSAP), a hardwired terminal line, an incoming phone line, etc. The definition of what constitutes a port is strictly a function of the port monitor itself.

A port monitor performs certain basic functions. Some of these are required to conform to the SAF; others may be specified by the requirements and design of the port monitor itself. Port monitors have two main functions: managing ports and monitoring ports for indications of activity.

Port Management
: The first function of a port monitor is to manage a port. The actual details of how a port is managed are defined by the person who defines the port monitor. A port monitor is not restricted to handling a single port; it may handle multiple ports simultaneously.

  Some examples of port management are setting the line speed on incoming phone connections, binding an appropriate network address, reinitializing the port when the service terminates, outputting a prompt, etc.

Activity Monitoring
: The second function of a port monitor is to monitor the port or ports for which it is responsible for indications of activity. Two types of activity may be detected.

The first is an indication to the port monitor to take some port monitor-specific action. Pressing the break key to indicate that the line speed should be cycled is an example of a port monitor activity. Not all port monitors need to recognize and respond to the same indications. The indication used to attract the attention of the port monitor is defined by the person who defines the port monitor.

The second is an incoming service request. When a service request is received, a port monitor must be able to determine which service is being requested from the port on which the request is received. The same service may be available on more than one port.

Other Port Monitor Functions

This section briefly describes other port monitor functions.

Restricting Access to the System
: A port monitor must be able to restrict access to the system without disturbing services that are still running. In order to do this, a port monitor must maintain two internal states: enabled and disabled. The port monitor starts in the state indicated by the ISTATE environment variable provided by the sac. See sac(1M) for details. Enabling or disabling a port monitor affects all ports for which the port monitor is responsible. If a port monitor is responsible for a single port, only that port will be affected. If a port monitor is responsible for multiple ports, the entire collection of ports will be affected. Enabling or disabling a port monitor is a dynamic operation: it causes the port monitor to change its internal state. The effect does not persist across new invocations of the port monitor. Enabling or disabling an individual port, however, is a static operation: it causes a change to an administrative file. The effect of this change will persist across new invocations of the port monitor.

Creating `utmpx` Entries
: Port monitors are responsible for creating `utmpx` entries with the type field set to USER_PROCESS for services they start. If this action has been specified, by using the `-fu` option in the `pmadm` command line that added the service, these `utmpx` entries may in turn be modified by the service. When the service terminates, the `utmpx` entry must be set to DEAD_PROCESS.

Port Monitor Process IDs and Lock Files
: When a port monitor starts, it writes its process id into a file named `_pid` in the current directory and places an advisory lock on the file.

Changing the Service Environment: Running
: `doconfig`(3NSL) Before invoking the service designated in the port monitor administrative file, `_pmtab`, a port monitor must arrange for the per-service configuration script to be run, if one exists, by calling the library function `doconfig`(3NSL). Because the per-service configuration script may specify the execution of restricted commands, as well as for other security reasons, port monitors are invoked with root permissions. The details of how services are invoked are specified by the person who defines the port monitor.

Terminating a Port Monitor
: A port monitor must terminate itself gracefully on receipt of the signal SIGTERM. The termination sequence is the following:

1. The port monitor enters the stopping state; no further service requests are accepted.
2. Any attempt to re-enable the port monitor will be ignored.
3. The port monitor yields control of all ports for which it is responsible. It must be possible for a new instantiation of the port monitor to start correctly while a previous instantiation is stopping.
4. The advisory lock on the process id file is released. Once this lock is released, the contents of the process id file are undefined and a new invocation of the port monitor may be started.

SAF Files

This section briefly covers the files used by the SAF.

The Port Monitor Administrative File

A port monitor's current directory contains an administrative file named `_pmtab`; `_pmtab` is maintained by the `pmadm` command in conjunction with a port monitor-specific administrative command.

The port monitor administrative command for a listen port monitor is `nlsadmin`(1M); the port monitor administrative command for ttymon is `ttyadm`(1M). Any port monitor written by a user must be provided with an administrative command specific to that port monitor to perform similar functions.

Per-Service Configuration Files

A port monitor's current directory also contains the per-service configuration scripts, if they exist. The names of the per-service configuration scripts correspond to the service tags in the `_pmtab` file.

Private Port Monitor Files

A port monitor may create private files in the directory `/var/saf/`*tag*, where *tag* is the name of the port monitor. Examples of private files are log files or temporary files.

The SAC/Port Monitor Interface

The SAC creates two environment variables for each port monitor it starts:PMTAG and ISTATE.

This variable is set to a unique port monitor tag by the SAC. The port monitor uses this tag to identify itself in response to `sac` messages. ISTATE is used to indicate to the port monitor what its initial internal state should be. ISTATE is set to "enabled" or "disabled" to indicate that the port monitor is to start in the enabled or disabled state respectively.

The SAC performs a periodic sanity poll of the port monitors. The SAC communicates with port monitors through FIFOs. A port monitor should open _pmpipe, in the current directory, to receive messages from the SAC and `../_sacpipe` to send return messages to the SAC.

Message Formats

This section describes the messages that may be sent from the SAC to a port monitor (`sac` messages), and from a port monitor to the SAC (port monitor messages). These messages are sent through FIFOs and are in the form of C structures.

`sac` Messages

The format of messages from the SAC is defined by the structure `sacmsg`:

```
struct sacmsg
{
    int sc_size; /* size of optional data portion */
    char sc_type; /* type of message */
};
```

The SAC may send four types of messages to port monitors. The type of message is indicated by setting the `sc_type` field of the `sacmsg` structure to one of the following:

SC_STATUS	status request
SC_ENABLE	enable message
SC_DISABLE	disable message
SC_READDB	message indicating that the port monitor's _pmtab file should be read

The `sc_size` field indicates the size of the optional data part of the message. See "Message Classes." For Solaris, `sc_size` should always be set to 0. A port monitor must respond to every message sent by the sac.

Port Monitor Messages

The format of messages from a port monitor to the SAC is defined by the structure `pmmsg`:

```
struct pmmsg {
    char pm_type;                /* type of message */
    unchar_t pm_state;           /* current state of port monitor */
    char pm_maxclass;            /* maximum message class this port
                                    monitor understands */
    char pm_tag[PMTAGSIZE + 1];  /* port monitor's tag */
    int pm_size;                 /* size of optional data portion */
};
```

Port monitors may send two types of messages to the SAC. The type of message is indicated by setting the `pm_type` field of the `pmmsg` structure to one of the following:

PM_STATUS
  state information

PM_UNKNOWN
  negative acknowledgment

For both types of messages, the `pm_tag` field is set to the port monitor's tag and the `pm_state` field is set to the port monitor's current state. Valid states are:

PM_STARTING	starting
PM_ENABLED	enabled
PM_DISABLED	disabled

PM_STOPPING stopping

The current state reflects any changes caused by the last message from the SAC. The status message is the normal return message. The negative acknowledgment should be sent only when the message received is not understood. `pm_size` indicates the size of the optional data part of the message. `pm_maxclass` is used to specify a message class. Both are discussed under "Message Classes." In Solaris, always set `pm_maxclass` to 1 and `sc_size` to 0. Port monitors may never initiate messages; they may only respond to messages that they receive.

Message Classes

The concept of message class has been included to accommodate possible SAF extensions. The messages described above are all class 1 messages. None of these messages contains a variable data portion; all pertinent information is contained in the message header. If new messages are added to the protocol, they will be defined as new message classes (for example, class 2). The first message the SAC sends to a port monitor will always be a class 1 message. Since all port monitors, by definition, understand class 1 messages, the first message the SAC sends is guaranteed to be understood. In its response to the SAC, the port monitor sets the `pm_maxclass` field to the maximum message class number for that port monitor. The SAC will not send messages to a port monitor from a class with a larger number than the value of `pm_maxclass`. Requests that require messages of a higher class than the port monitor can understand will fail. For Solaris, always set `pm_maxclass` to 1.

For any given port monitor, messages of class `pm_maxclass` and messages of all classes with values lower than `pm_maxclass` are valid. Thus, if the `pm_maxclass` field is set to 3, the port monitor understands messages of classes 1, 2, and 3. Port monitors may not generate messages; they may only respond to messages. A port monitor's response must be of the same class as the originating message. Since only the SAC can generate messages, this protocol will function even if the port monitor is capable of dealing with messages of a higher class than the SAC can generate. `pm_size` (an element of the pmmsg structure) and `sc_size` (an element of the `sacmsg` structure) indicate the size of the optional data part of the message. The format of this part of the message is undefined. Its definition is inherent in the type of message. For Solaris, always set both `sc_size` and `pm_size` to `0`.

Administrative Interface

This section discusses the port monitor administrative files available under the SAC.

The SAC Administrative File _sactab

The service access controller's administrative file contains information about all the port monitors for which the SAC is responsible. This file exists on the delivered system. Initially, it is empty except for a single comment line that contains the version number of the SAC. Port monitors are added to the system by making entries in the SAC's administrative file. These entries should be made using the administrative command `sacadm`(1M) with a `-a` option. `sacadm`(1M) is also used to remove entries from the SAC's administrative file. Each entry in the SAC's administrative file contains the following information.

PMTAG
A unique tag that identifies a particular port monitor. The system administrator is responsible for naming a port monitor. This tag is then used by the SAC to identify the port monitor for all administrative purposes. PMTAG may consist of up to 14 alphanumeric characters.

PMTYPE
The type of the port monitor. In addition to its unique tag, each port monitor has a type designator. The type designator identifies a group of port monitors that are different invocations of the same entity. ttymon and listen are examples of valid port monitor types. The type designator is used to facilitate the administration of groups of related port monitors. Without a type designator, the system administrator has no way of knowing which port monitor tags correspond to port monitors of the same type. PMTYPE may consist of up to 14 alphanumeric characters.

FLGS
The flags that are currently defined are:

d When started, do not enable the port monitor.

x Do not start the port monitor.

If no flag is specified, the default action is taken. By default a port monitor is started and enabled.

RCNT
The number of times a port monitor may fail before being placed in a failed state. Once a port monitor enters the failed state, the SAC will not try to restart it. If a count is not specified when the entry is created, this field is set to 0. A restart count of 0 indicates that the port monitor is not to be restarted when it fails.

COMMAND
A string representing the command that will start the port monitor. The first component of the string, the command itself, must be a full path name.

The Port Monitor Administrative File _pmtab

Each port monitor will have two directories for its exclusive use. The current directory will contain files defined by the SAF (`_pmtab`, `_pid`) and the per-service configuration scripts, if they exist. The directory `/var/saf/`*pmtag*, where *pmtag* is the tag of the port monitor, is available for the port monitor's private files. Each port monitor has its own administrative file. The `pmadm`(1M) command should be used to add, remove, or modify service entries in this file. Each time a change is made using `pmadm`(1M), the corresponding port monitor rereads its administrative file. Each entry in a port monitor's administrative file defines how the port monitor treats a specific port and what service is to be invoked on that port. Some fields must be present for all types of port monitors. Each entry must include a service tag to identify the service uniquely and an identity to be assigned to the service when it is started (for example, root).

The combination of a service tag and a port monitor tag uniquely define an instance of a service. The same service tag may be used to identify a service under a different port monitor. The record must also contain port monitor specific data (for example, for a

ttymon port monitor, this will include the prompt string which is meaningful to ttymon). Each type of port monitor must provide a command that takes the necessary port monitor-specific data as arguments and outputs these data in a form suitable for storage in the file. The `ttyadm`(1M) command does this for ttymon and `nlsadmin`(1M) does it for listen. For a user-defined port monitor, a similar administrative command must also be supplied. Each service entry in the port monitor administrative file must have the following format and contain the information listed below:

```
svctag:flgs:id:reserved:reserved:reserved:pmspecific# comment
```

SVCTAG is a unique tag that identifies a service. This tag is unique only for the port monitor through which the service is available. Other port monitors may offer the same or other services with the same tag. A service requires both a port monitor tag and a service tag to identify it uniquely. SVCTAG may consist of up to 14 alphanumeric characters. The service entries are defined as:

`FLGS`
Flags with the following meanings may currently be included in this field:

x Do not enable this port. By default the port is enabled.

u Create a utmpx entry for this service. By default no utmpx entry is created for the service.

`ID`
The identity under which the service is to be started. The identity has the form of a login name as it appears in `/etc/passwd`.

`PMSPECIFIC`
Examples of port monitor information are addresses, the name of a process to execute, or the name of a STREAMS pipe to pass a connection through. This information will vary to meet the needs of each different type of port monitor.

`COMMENT`
A comment associated with the service entry. Port monitors may ignore the *u* flag if creating a utmpx entry for the service is not appropriate to the manner in which the service is to be invoked. Some services may not start properly unless utmpx entries have been created for them (for example, login). Each port monitor administrative file must contain one special comment of the form:

VERSION=value

where *value* is an integer that represents the port monitor's version number. The version number defines the format of the port monitor administrative file. This comment line is created automatically when a port monitor is added to the system. It appears on a line by itself, before the service entries.

Monitor-Specific Administrative Command

Previously, two pieces of information included in the `_pmtab` file were described: the port monitor's version number and the port monitor part of the service entries in the port monitor's `_pmtab` file. When a new port monitor is added, the version number

must be known so that the _pmtab file can be correctly initialized. When a new service is added, the port monitor part of the _pmtab entry must be formatted correctly. Each port monitor must have an administrative command to perform these two tasks. The person who defines the port monitor must also define such an administrative command and its input options. When the command is invoked with these options, the information required for the port monitor part of the service entry must be correctly formatted for inclusion in the port monitor's _pmtab file and must be written to the standard output. To request the version number the command must be invoked with a -V option; when it is invoked in this way, the port monitor's current version number must be written to the standard output. If the command fails for any reason during the execution of either of these tasks, no data should be written to standard output.

The Port Monitor/Service Interface

The interface between a port monitor and a service is determined solely by the service. Two mechanisms for invoking a service are presented here as examples.

New Service Invocations

The first interface is for services that are started anew with each request. This interface requires the port monitor to first fork(2) a child process. The child will eventually become the designated service by performing an exec(1). Before the exec(1) happens, the port monitor may take some port monitor-specific action; however, one action that must occur is the interpretation of the per-service configuration script, if one is present. This is done by calling the library routine doconfig(3NSL).

Standing Service Invocations

The second interface is for invocations of services that are actively running. To use this interface, a service must have one end of a stream pipe open and be prepared to receive connections through it.

Port Monitor Requirements

To implement a port monitor, several generic requirements must be met. This section summarizes these requirements. In addition to the port monitor itself, an administrative command must be supplied.

Initial Environment

When a port monitor is started, it expects an initial execution environment in which:

- It has no file descriptors open
- It cannot be a process group leader
- It has an entry in /etc/utmpx of type LOGIN_PROCESS
- An environment variable, ISTATE, is set to "enabled" or "disabled" to indicate the port monitor's correct initial state
- An environment variable, PMTAG, is set to the port monitor's assigned tag
- The directory that contains the port monitor's administrative files is its current directory
- pThe port monitor is able to create private files in the directory /var/saf/*tag*, where *tag* is the port monitor's tag
- The port monitor is running with user id 0 (root)

Important Files

Relative to its current directory, the following key files exist for a port monitor.

`_config`
The port monitor's configuration script. The port monitor configuration script is run by the SAC. The SAC is started by `init`(1M) as a result of an entry in `/etc/inittab` that calls `sac`(1M).

`_pid`
The file into which the port monitor writes its process id.

`_pmtab`
The port monitor's administrative file. This file contains information about the ports and services for which the port monitor is responsible.

`_pmpipe`
The FIFO through which the port monitor will receive messages from the SAC.

`svctag`
The per-service configuration script for the service with the tag *svctag*.

`../_sacpipe`
The FIFO through which the port monitor will send messages to `sac`(1M).

Port Monitor Responsibilities

A port monitor is responsible for performing the following tasks in addition to its port monitor function:

- Write its process id into the file `_pid` and place an advisory lock on the file
- Terminate gracefully on receipt of the signal SIGTERM
- Follow the protocol for message exchange with the SAC

A port monitor must perform the following tasks during service invocation:

- Create a `utmpx` entry if the requested service has the `u` flag set in `_pmtab`
- Port monitors may ignore this flag if creating a `utmpx` entry for the service does not make sense because of the manner in which the service is to be invoked. On the other hand, some services may not start properly unless utmpx entries have been created for them.
- Interpret the per-service configuration script for the requested service, if it exists, by calling the `doconfig`(3NSL) library routine

Configuration Files and Scripts

The library routine `doconfig`(3NSL), defined in `libnsl.so`, interprets the configuration scripts contained in the files `/etc/saf/_sysconfig` (the per-system configuration file), and `/etc/saf/`*pmtag*`/_config` (per-port monitor configuration files); and in `/etc/saf/`*pmtag*`/svctag` (per-service configuration files). Its syntax is:

```
#include <sac.h>
int doconfig (int fd, char *script, long rflag);
```

`script` is the name of the configuration script; *fd* is a file descriptor that designates the stream to which stream manipulation operations are to be applied; *rflag* is a bitmask that indicates the mode in which script is to be interpreted. *rflag* may take two values, NORUN and NOASSIGN, which may be or'd. If *rflag* is zero, all commands in

the configuration script are eligible to be interpreted. If *rflag* has the NOASSIGN bit set, the assign command is considered illegal and will generate an error return. If *rflag* has the NORUN bit set, the run and runwait commands are considered illegal and will generate error returns. If a command in the script fails, the interpretation of the script ceases at that point and a positive integer is returned; this number indicates which line in the script failed. If a system error occurs, a value of –1 is returned. If a script fails, the process whose environment was being established should not be started. In the example, `doconfig`(3NSL) is used to interpret a per-service configuration script.

```
. . .
      if ((i = doconfig (fd, svctag, 0)) != 0){
      error ("doconfig failed on line %d of script %s",i,svctag);
  }
```

The Per-System Configuration File

The per-system configuration file, `/etc/saf/_sysconfig`, is delivered empty. It may be used to customize the environment for all services on the system by writing a command script in the interpreted language described in this chapter and on the `doconfig`(3NSL) manpage. When the SAC is started, it calls the `doconfig`(3NSL) function to interpret the per-system configuration script. The SAC is started when the system enters multiuser mode.

Per-Port Monitor Configuration Files

Per-port monitor configuration scripts (`/etc/saf/`*pmtag*`/_config`) are optional. They allow the user to customize the environment for any given port monitor and for the services that are available through the ports for which that port monitor is responsible. Per-port monitor configuration scripts are written in the same language used for per-system configuration scripts. The per-port monitor configuration script is interpreted when the port monitor is started. The port monitor is started by the SAC after the SAC has itself been started and after it has run its own configuration script, `/etc/saf/_sysconfig`. The per-port monitor configuration script may override defaults provided by the per-system configuration script.

Per-Service Configuration Files

Per-service configuration files allow the user to customize the environment for a specific service. For example, a service may require special privileges that are not available to the general user. Using the language described in the `doconfig`(3NSL) manpage, you can write a script that will grant or limit such special privileges to a particular service offered through a particular port monitor. The per-service configuration may override defaults provided by higher-level configuration scripts. For example, the per-service configuration script may specify a set of STREAMS modules other than the default set.

The Configuration Language

The language in which configuration scripts are written consists of a sequence of commands, each of which is interpreted separately. The following reserved keywords are defined: `assign`, `push`, `pop`, `runwait`, and `run`. The comment character is #. Blank lines are not significant. No line in a command script may exceed 1024 characters.

`assign` *variable=value*
Used to define environment variables; *variable* is the name of the environment variable and *value* is the value to be assigned to it. The value assigned must be a string constant; no form of parameter substitution is available. *value* may be quoted. The quoting rules are those used by the shell for defining environment variables. `assign` will fail if space cannot be allocated for the new variable or if any part of the specification is invalid.

`push` *module1*[, *module2*, *module3*, . . .]
Used to push STREAMS modules onto the stream designated by `fd`; *module1* is the name of the first module to be pushed, *module2* is the name of the second module to be pushed, and so on. The command will fail if any of the named modules cannot be pushed. If a module cannot be pushed, the subsequent modules on the same command line will be ignored and modules that have already been pushed will be popped.

`pop` [*module*]
Used to pop STREAMS modules off the designated stream. If `pop` is invoked with no arguments, the top module on the stream is popped. If an argument is given, modules will be popped one at a time until the named module is at the top of the stream. If the named module is not on the designated stream, the stream is left as it was and the command fails. If *module* is the special keyword `ALL`, then all modules on the stream will be popped. Only modules above the topmost driver are affected.

`runwait command`
The `runwait` command runs a command and waits for it to complete; `command` is the path name of the command to be run. The command is run with `/bin/sh -c` prepended to it; shell scripts may thus be executed from configuration scripts. The `runwait` command will fail if command cannot be found or cannot be executed, or if `command` exits with a nonzero status.

`run command`
The `run` command is identical to `runwait` except that it does not wait for command to complete; `command` is the path name of the command to be run. `run` will not fail unless it is unable to create a child process to execute the command. Although they are syntactically indistinguishable, some of the commands available to `run` and `runwait` are interpreter built-in commands. Interpreter built-ins are used when it is necessary to alter the state of a process within the context of that process. The `doconfig` interpreter built-in commands are similar to the shell special commands and, like these, they do not spawn another process for execution. See the `sh`(1) man page. The initial set of built-in commands is: `cd`, `ulimit`, `umask`.

Sample Port Monitor Code

This example shows an example of a "null" port monitor that simply responds to messages from the SAC.

```
# include <stdlib.h>
# include <stdio.h>
# include <unistd.h>
# include <fcntl.h>
# include <signal.h>
```

```
# include <sac.h>

char Scratch[BUFSIZ]; /* scratch buffer */
char Tag[PMTAGSIZE + 1]; /* port monitor's tag */
FILE *Fp; /* file pointer for log file */
FILE *Tfp; /* file pointer for pid file */
char State; /* portmonitor's current state*/

main(argc, argv)
    int argc;
    char *argv[];
{
    char *istate;
    strcpy(Tag, getenv("PMTAG"));
/*
 * open up a log file in port monitor's private directory
 */
    sprintf(Scratch, "/var/saf/%s/log", Tag);
    Fp = fopen(Scratch, "a+");
    if (Fp == (FILE *)NULL)
        exit(1);
    log(Fp, "starting");
/*
 * retrieve initial state (either "enabled" or "disabled") and set
 * State accordingly
 */
    istate = getenv("ISTATE");
    sprintf(Scratch, "ISTATE is %s", istate);
    log(Fp, Scratch);
    if (!strcmp(istate, "enabled"))
        State = PM_ENABLED;
    else if (!strcmp(istate, "disabled"))
        State = PM_DISABLED;
    else {
        log(Fp, "invalid initial state");
        exit(1);
    }
    sprintf(Scratch, "PMTAG is %s", Tag);
    log(Fp, Scratch);
/*
 * set up pid file and lock it to indicate that we are active
 */
    Tfp = fopen("_pid", "w");
    if (Tfp == (FILE *)NULL) {
        log(Fp, "couldn't open pid file");
        exit(1);
    }
    if (lockf(fileno(Tfp), F_TEST, 0) < 0) {
        log(Fp, "pid file already locked");
        exit(1);
    }

    log(Fp, "locking file");
    if (lockf(fileno(Tfp), F_LOCK, 0) < 0) {
        log(Fp, "lock failed");
        exit(1);
    }
    fprintf(Tfp, "%d", getpid());
```

```
    fflush(Tfp);

/*
 * handle poll messages from the sac ... this function never returns
 */
    handlepoll();
    pause();
    fclose(Tfp);
    fclose(Fp);
}

handlepoll()
{
    int pfd; /* file descriptor for incoming pipe */
    int sfd; /* file descriptor for outgoing pipe */
    struct sacmsg sacmsg; /* incoming message */
    struct pmmsg pmmsg; /* outgoing message */
/*
 * open pipe for incoming messages from the sac
 */
    pfd = open("_pmpipe", O_RDONLY|O_NONBLOCK);
    if (pfd < 0) {
        log(Fp, "_pmpipe open failed");
        exit(1);
    }
/*
 * open pipe for outgoing messages to the sac
 */
    sfd = open("../_sacpipe", O_WRONLY);
    if (sfd < 0) {
        log(Fp, "_sacpipe open failed");
        exit(1);
    }
/*
 * start to build a return message; we only support class 1 messages
 */
    strcpy(pmmsg.pm_tag, Tag);
    pmmsg.pm_size = 0;
    pmmsg.pm_maxclass = 1;
/*
 * keep responding to messages from the sac
 */
     for (;;) {
         if (read(pfd, &sacmsg, sizeof(sacmsg)) != sizeof(sacmsg)) {
             log(Fp, "_pmpipe read failed");
             exit(1);
         }
/*
 * determine the message type and respond appropriately
 */
         switch (sacmsg.sc_type) {
             case SC_STATUS:
                 log(Fp, "Got SC_STATUS message");
                 pmmsg.pm_type = PM_STATUS;
                 pmmsg.pm_state = State;
                 break;
             case SC_ENABLE:
                 /*note internal state change below*/
```

```
                log(Fp, "Got SC_ENABLE message");
                pmmsg.pm_type = PM_STATUS;
                State = PM_ENABLED;
                pmmsg.pm_state = State;
                break;
            case SC_DISABLE:
                /*note internal state change below*/
                log(Fp, "Got SC_DISABLE message");
                pmmsg.pm_type = PM_STATUS;
                State = PM_DISABLED;
                pmmsg.pm_state = State;
                break;
            case SC_READDB:
                /*
                * if this were a fully functional port
                * monitor it would read _pmtab here
                * and take appropriate action
                */
                log(Fp, "Got SC_READDB message");
                pmmsg.pm_type = PM_STATUS;
                pmmsg.pm_state = State;
                break;
            default:
                sprintf(Scratch, "Got unknown message <%d>",
                sacmsg.sc_type);
                log(Fp, Scratch);
                pmmsg.pm_type = PM_UNKNOWN;
                pmmsg.pm_state = State;
                break;
        }
/*
 * send back a response to the poll
 * indicating current state
 */
        if (write(sfd, &pmmsg, sizeof(pmmsg)) != sizeof(pmmsg))
            log(Fp, "sanity response failed");
    }
}
/*
 * general logging function
 */
log(fp, msg)
    FILE *fp;
    char *msg;
{
    fprintf(fp, "%d; %s\n", getpid(), msg);
    fflush(fp);
}
```

The sac.h Header File

The following example shows the sac.h header file.

```
/* length in bytes of a utmpx id */
# define IDLEN 4
/* wild character for utmpx ids */
# define SC_WILDC 0xff
/* max len in bytes for port monitor tag */
# define PMTAGSIZE 14
/*
```

```
 * values for rflag in doconfig()
 */
/* don't allow assign operations */
# define NOASSIGN 0x1
/* don't allow run or runwait operations */
# define NORUN 0x2
/*
 * message to SAC (header only). This header is forever fixed. The
 * size field (pm_size) defines the size of the data portion of the
 * message, which follows the header. The form of this optional data
 * portion is defined strictly by the message type (pm_type).
 */
struct pmmsg {
    char pm_type;               /* type of message */
    unchar_t pm_state;            /* current state of pm */
    char pm_maxclass;           /* max message class this port monitor
                                   understands */
    char pm_tag[PMTAGSIZE + 1]; /* pm's tag */
    int pm_size;                /* size of opt data portion */
};
/*
 * pm_type values
 */
# define PM_STATUS 1 /* status response */
# define PM_UNKNOWN 2 /* unknown message was received */
/*
 * pm_state values
 */
/*
 * Class 1 responses
 */
# define PM_STARTING 1   /* monitor in starting state */
# define PM_ENABLED 2    /* monitor in enabled state */
# define PM_DISABLED 3   /* monitor in disabled state */
# define PM_STOPPING 4   /* monitor in stopping state */
/*
 * message to port monitor
 */
struct sacmsg {
    int sc_size;         /* size of optional data portion */
    char sc_type;        /* type of message */
};
/*
 * sc_type values
 * These represent commands that the SAC sends to a port monitor.
 * These commands are divided into "classes" for extensibility. Each
 * subsequent "class" is a superset of the previous "classes" plus
 * the new commands defined within that "class". The header for all
 * commands is identical; however, a command may be defined such that
 * an optional data portion may be sent in addition to the header.
 * The format of this optional data piece is self-defining based on
 * the command. The first message sent by the SAC
 * will always be a class 1 message. The port monitor response
 * indicates the maximum class that it is able to understand. Another
 * note is that port monitors should only respond to a message with
 * an equivalent class response (i.e. a class 1 command causes a
 * class 1 response).
 */
```

```
/*
 * Class 1 commands (currently, there are only class 1 commands)
 */
# define SC_STATUS 1    /* status request *
# define SC_ENABLE 2    /* enable request */
# define SC_DISABLE 3   /* disable request */
# define SC_READDB 4    /* read pmtab request */
/*
 * 'errno' values for Saferrno, note that Saferrno is used by both
 * pmadm and sacadm and these values are shared between them
 */
# define E_BADARGS 1   /* bad args/ill-formed cmd line */
# define E_NOPRIV 2    /* user not priv for operation */
# define E_SAFERR 3    /* generic SAF error */
# define E_SYSERR 4    /* system error */
# define E_NOEXIST 5   /* invalid specification */
# define E_DUP 6       /* entry already exists */
# define E_PMRUN 7     /* port monitor is running */
# define E_PMNOTRUN 8  /* port monitor is not running */
# define E_RECOVER 9
   /* in recovery */
```

Directory Structure

This section gives a description of the SAF files and directories.

`/etc/saf/_sysconfig`	The per-system configuration script.
`/etc/saf/_sactab`	The SAC's administrative file. Contains information about the port monitors for which the SAC is responsible.
`/etc/saf/`*pmtag*	The home directory for port monitor *pmtag*.
`/etc/saf/`*pmtag*`/_config`	The per-port monitor configuration script for port monitor pmtag. `/etc/saf/`*pmtag*`/_pmtab` Port monitor pmtag's administrative file. Contains information about the services for which *pmtag* is responsible. `/etc/saf/`*pmtag*`/`*svctag* The file in which the per-service configuration script for service *svctag* (available through port monitor `pmtag`) is placed. `/etc/saf/`*pmtag*`/_pid` The file in which a port monitor writes its process id in the current directory and places an advisory lock on the file. `/etc/saf/ pmtag /_pmpipe` The file in which the port monitor receives messages from the SAC and `../_sacpipe` and sends return messages to the SAC. `/var/saf/_log` The SAC's log file. `/var/saf/`*pmtag* The directory for files created by port monitor *pmtag,* for example its log file.

LIST OF COMMANDS

The following administrative commands relate to SAF.

`sacadm`(1M)	port monitor administrative command
`pmadm`(1M)	service administration command

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsr

SEE ALSO

`exec`(1), `sh`(1), `init`(1M), `nlsadmin`(1M), `pmadm`(1M), `sac`(1M), `sacadm`(1M), `ttyadm`(1M), `fork`(2), `doconfig`(3NSL), `attributes`(5)

NAME

sar, sa1, sa2, sadc – system activity report package

SYNOPSIS

/usr/lib/sa/sadc [*t n*] [*ofile*]

/usr/lib/sa/sa1 [*t n*]

/usr/lib/sa/sa2 [-aAbcdgkmpqruvwy] [-e *time*] [-f *filename*] [-i *sec*] [-s *time*]

DESCRIPTION

System activity data can be accessed at the special request of a user (see sar(1)) and automatically, on a routine basis, as described here. The operating system contains several counters that are incremented as various system actions occur. These include counters for CPU utilization, buffer usage, disk and tape I/O activity, TTY device activity, switching and system-call activity, file-access, queue activity, inter-process communications, and paging. For more general system statistics, use iostat (1M), sar(1), or vmstat(1M).

sadc and two shell procedures, sa1 and sa2, are used to sample, save, and process this data.

sadc, the data collector, samples system data *n* times, with an interval of *t* seconds between samples, and writes in binary format to *ofile* or to standard output. The sampling interval *t* should be greater than 5 seconds; otherwise, the activity of sadc itself may affect the sample. If *t* and *n* are omitted, a special record is written. This facility can be used at system boot time, when booting to a multi-user state, to mark the time at which the counters restart from zero. For example, when accounting is enabled, the /etc/init.d/perf file writes the restart mark to the daily data file using the command entry:

```
su sys -c "/usr/lib/sa/sadc /var/adm/sa/sa`date +%d`"
```

The shell script sa1, a variant of sadc, is used to collect and store data in the binary file /var/adm/sa/sa*dd*, where dd is the current day. The arguments *t* and *n* cause records to be written *n* times at an interval of *t* seconds, or once if omitted. The following entries in /var/spool/cron/crontabs/sys will produce records every 20 minutes during working hours and hourly otherwise:

```
0 * * * 0-6 /usr/lib/sa/sa1
20,40 8–17 * * 1–5 /usr/lib/sa/sa1
```

See crontab(1) for details.

The shell script sa2, a variant of sar, writes a daily report in the file /var/adm/sa/sar*dd*. See the OPTIONS section in sar(1) for an explanation of the various options. The following entry in /var/spool/cron/crontabs/sys will report important activities hourly during the working day:

```
5 18 * * 1–5 /usr/lib/sa/sa2 -s 8:00 -e 18:01 -i 1200 -A
```

FILES

`/etc/init.d/perf`	
`/tmp/sa.adrfl`	address file
`/var/adm/sa/sadd`	daily data file
`/var/adm/sa/sardd`	daily report file
`/var/spool/cron/crontabs/sys`	

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWaccu

SEE ALSO

`crontab`(1), `sag`(1), `sar`(1), `timex`(1), `iostat`(1M), `vmstat`(1M), `attributes`(5)

System Administration Guide: Basic Administration

NAME

savecore – save a crash dump of the operating system

SYNOPSIS

/usr/bin/savecore [-Lvd] [-f *dumpfile*] *directory*

DESCRIPTION

The savecore utility saves a crash dump of the kernel (assuming that one was made) and writes a reboot message in the shutdown log. It is invoked by the /etc/init.d/savecore file after the system boots, if savecore is enabled by way of dumpadm(1M). savecore is enable on reboot by default.

The savecore utility checks the crash dump to be certain it corresponds with the version of the operating system currently running. If it does, savecore saves the crash dump data in the file *directory*/vmcore.*n* and the kernel's namelist in *directory*/unix.*n*. The trailing .*n* in the pathnames is replaced by a number which grows every time savecore is run in that directory.

Before writing out a crash dump, savecore reads a number from the file *directory*/minfree. This is the minimum number of kilobytes that must remain free on the file system containing *directory*. If after saving the crash dump the file system containing *directory* would have less free space the number of kilobytes specified in minfree, the crash dump is not saved. if the minfree file does not exist, savecore assumes a minfree value of 1 megabyte.

The savecore utility also logs a reboot message using facility LOG_AUTH (see syslog(3C)). If the system crashed as a result of a panic, savecore logs the panic string too.

OPTIONS

The following options are supported:

-d — Disregard dump header valid flag. Force savecore to attempt to save a crash dump even if the header information stored on the dump device indicates the dump has already been saved.

-f *dumpfile* — Attempt to save a crash dump from the specified file instead of from the system's current dump device. This option may be useful if the information stored on the dump device has been copied to an on-disk file by means of the dd(1M) command.

-L — Save a crash dump of the live running Solaris system, without actually rebooting or altering the system in any way. This option forces savecore to save a live snapshot of the system to the dump device, and then immediately to retrieve the data and to write it out to a new set of crash dump files in the specified directory. Live system crash dumps may only be performed if you have configured your system to have a dedicated dump device using dumpadm(1M).

-v — Verbose. Enables verbose error messages from savecore.

OPERANDS

The following operands are supported:

directory — Save the crash dump files to the specified directory. If no directory argument is present on the command line, savecore saves the crash

dump files to the default `savecore` *directory*, configured by the `dumpadm`(1M) command.

FILES

directory`/vmcore.`*n*

directory`/unix.`*n*

directory`/bounds`

directory`/minfree`

`/dev/ksyms`	the kernel namelist
`/etc/init.d/savecore`	
`/var/crash/'uname -n'`	default crash dump directory

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu (32-bit) SUNWcsxu (64-bit)

SEE ALSO

`adb`(1), `mdb`(1), `dd`(1M), `dumpadm`(1M), `syslog`(3C), `attributes`(5)

BUGS

If the dump device is also being used as a swap device, you must run `savecore` very soon after booting, before the swap space containing the crash dump is overwritten by programs currently running.

NAME

sckmd – Sun Fire 15K key management daemon

SYNOPSIS

/usr/platform/SUNW,Sun-Fire-15000/lib/sckmd

DESCRIPTION

`sckmd` is a server process that resides on a Sun Fire 15K domain. `sckmd` maintains the Internet Protocol Security (IPsec) Security Associations (SAs) needed to secure the communication between the Sun Fire 15K System Controller (SC) and the `cvcd`(1M) and `dcs`(1M) daemons running on a Sun Fire 15K domain. See `ipsec`(7P) for a description of Security Associations.

`sckmd` receives SAs from the SC and provides these SAs to the Security Association Databases (SADBs) using `pf_key`(7P).

`sckmd` normally starts up at system boot time. Each domain supports only one running `sckmd` process at a time.

FILES

`/etc/inet/ipsecinit.conf`	Configuration file for default system-wide IPsec policies

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Architecture	Sun Fire 15K systems
Availability	SUNWsckmx.u, SUNWsckmu.u, SUNWsckmr

SEE ALSO

`cvcd`(1M), `dcs`(1M), `ipsecconf`(1M), `attributes`(5), `authmd5h`(7M), `encr3des`(7M), `ipsec`(7P), `pf_key`(7P)

Sun Enterprise 10000 SSP Reference Manual

Sun System Management Services (SMS) Reference Manual

NOTES

IPsec is used by Sun Fire 15K systems to secure the communication between the SC, and the `cvcd`(1M) and `dcs`(1M) daemons running on a domain. System-wide IPsec policies for these daemons are configured on a domain with `ipsecconf`(1M). Default policies are defined when the `SUNWsckmr` package is installed on a Sun Fire 15K domain at OS install time.

Package `SUNWsckmr` configures default system-wide policies for `cvcd`(1M) and `dcs`(1M) by adding the following entries in `/etc/inet/ipsecinit.conf`:

```
{ dport sun-dr ulp tcp } permit
        { auth_alg md5 }
{ sport sun-dr ulp tcp } apply
        { auth_alg md5 sa unique }
{ dport cvc_hostd ulp tcp } permit
        { auth_alg md5 }
{ sport cvc_hostd ulp tcp } apply
        { auth_alg md5 sa unique }
```

The cvc_hostd service represents cvcd(1M) and the sun-dr service represents dcs(1M) in the preceding entries.

These policies conform to the format defined by ipsec(7P) and require HMAC-MD5 authentication. See authmd5h(7M).

System-wide policies for cvcd(1M) and dcs(1M) configured on a domain using ipsecconf(1M) must match the IPsec policies defined for these services on the SC. On an SC, IPsec policies for these services are defined by the SMS key management daemon. Refer to thekmd(1M) man page in the *Sun System Management Services (SMS) Reference Manual*.

IPsec encryption or authentication with encryption can be enabled on the domain using the encr_algs and encr_auth_algs properties, as described in the ipsecconf(1M) manual page. For example, the following ipsecconf(1M) entries require Triple-DES and HMAC-MD5 authentication for the network console cvcd(1M) service:

```
{ dport cvc_hostd ulp tcp } permit
        { encr_algs 3des encr_auth_algs md5 }
{ sport cvc_hostd ulp tcp } apply
        { encr_algs 3des encr_auth_algs md5 sa unique }
```

See encr3des(7M) for Triple-DES authentication and authmd5h(7M) for HMAC-M5 authentication.

NAME

sendmail – send mail over the internet

SYNOPSIS

```
/usr/lib/sendmail [-ba] [-bD] [-bd] [-bi] [-bm] [-bp] [-bP] [-bs]
     [-bt] [-bv] [-B type] [-C file] [-d X] [-F fullname] [-f name] [-G]
     [-h N] [-L tag] [-M xvalue] [-Nnotifications] [-n] [-Ooption =value]
     [-o xvalue] [-p protocol] [-q [time]] [-q Xstring] [-R ret] [-r name]
     [-t] [-V envid] [-v] [-X logfile] [ address...]
```

DESCRIPTION

`sendmail` sends a message to one or more people, routing the message over whatever networks are necessary. `sendmail` does internetwork forwarding as necessary to deliver the message to the correct place.

`sendmail` is not intended as a user interface routine; other programs provide user-friendly front ends. `sendmail` is used only to deliver pre-formatted messages.

With no flags, `sendmail` reads its standard input up to an EOF, or a line with a single dot, and sends a copy of the letter found there to all of the addresses listed. It determines the network to use based on the syntax and contents of the addresses.

Local addresses are looked up in the local `aliases`(4) file, or in a name service as defined by the `nsswitch.conf`(4) file, and aliased appropriately. In addition, if there is a `.forward` file in a recipient's home directory, `sendmail` forwards a copy of each message to the list of recipients that file contains. Refer to the `NOTES` section for more information about `.forward` files. Aliasing can be prevented by preceding the address with a backslash.

There are several conditions under which the expected behavior is for the alias database to be either built or rebuilt. This cannot occur under any circumstances unless root owns *and* has exclusive write permission to the `/etc/mail/aliases*` files.

If a message is found to be undeliverable, it is returned to the sender with diagnostics that indicate the location and nature of the failure; or, the message is placed in a `dead.letter` file in the sender's home directory.

OPTIONS

The following options are supported:

`-Ac` Use `submit.cf` even if the operation mode does not indicate an initial mail submission.

`-Am` Use `sendmail.cf` even if the operation mode indicates an initial mail submission.

`-ba` Go into `ARPANET` mode. All input lines must end with a RETURN-LINEFEED, and all messages are generated with a RETURN-LINEFEED at the end. Also, the `From:` and `Sender:` fields are examined for the name of the sender.

`-bd` Run as a daemon in the background, waiting for incoming SMTP connections.

`-bD`	Run as a daemon in the foreground, waiting for incoming SMTP connections.
`-bi`	Initialize the `aliases`(4) database. Root must own *and* have exclusive write permission to the `/etc/mail/aliases*` files for successful use of this option.
`-bm`	Deliver mail in the usual way (default).
`-bp`	Print a summary of the mail queues.
`-bP`	Print the number of entries in the queues. This option is only available with shared memory support.
`-bs`	Use the SMTP protocol as described in RFC 821. This flag implies all the operations of the `-ba` flag that are compatible with SMTP.
`-bt`	Run in address test mode. This mode reads addresses and shows the steps in parsing; it is used for debugging configuration tables.
`-bv`	Verify names only; do not try to collect or deliver a message. Verify mode is normally used for validating users or mailing lists.
`-B` *type*	Indicate body *type* (`7BIT` or `8BITMIME`).
`-C` *file*	Use alternate configuration file.
`-d` *X*	Set debugging value to *X*.
`-F` *fullname*	Set the full name of the sender.
`-G`	When accepting messages by way of the command line, indicate that they are for relay (gateway) submission. When this flag is set, `sendmail` might complain about syntactically invalid messages, for example, unqualified host names, rather than fixing them. `sendmail` does not do any canonicalization in this mode.
`-f` *name*	Sets the name of the "from" person (that is, the sender of the mail).
`-h` *N*	Set the hop count to *N*. The hop count is incremented every time the mail is processed. When it reaches a limit, the mail is returned with an error message, the victim of an aliasing loop.
`-L` *tag*	Set the identifier used in `syslog` messages to the supplied *tag*.
`-M`*xvalue*	Set macro *x* to the specified *value*.
`-n`	Do not do aliasing.
`-N` *notifications*	Tag all addresses being sent as wanting the indicated *notifications*, which consists of the word "NEVER" or a comma-separated list of "SUCCESS", "FAILURE", and "DELAY" for successful delivery, failure and a message that is stuck in a queue somwhere. The default is "FAILURE,DELAY".

`-o`*xvalue*	Set option *x* to the specified *value*. `Processing Options` are described below.
`-O`*option=value*	Set *option* to the specified *value* (for long from names). `Processing Options` are described below.
`-p` *protocol*	Set the sending protocol. The *protocol* field can be in form *protocol*`:`*host* to set both the sending protocol and the sending host. For example: `-pUUCP:uunet` sets the sending *protocol* to `UUCP` and the sending host to `uunet`. (Some existing programs use `-oM` to set the *r* and *s* macros; this is equivalent to using `-p`).
`-q`[*time*]	Process saved messages in the queue at given intervals. If *time* is omitted, process the queue once. *time* is given as a tagged number, with *s* being seconds, *m* being minutes, *h* being hours, *d* being days, and *w* being weeks. For example, `-q1h30m` or `-q90m` would both set the timeout to one hour thirty minutes. By default, sendmail runs in the background. This option can be used safely with -bd.
`-qp`[*time*]	Similar to `-q`*time*, except that instead of periodically forking a child to process the queue, `sendmail` forks a single persistent child for each queue that alternates between processing the queue and sleeping. The sleep time (*time*) is specified as the argument; it defaults to `1` second. The process always sleeps at least `5` seconds if the queue was empty in the previous queue run.
`-qf`	Process saved messages in the queue once and do not `fork`(2), but run in the foreground.
`-qG` *name*	Process jobs in queue group called *name* only.
`-q[!]I` *substr*	Limit processed jobs to those containing *substr* as a substring of the queue ID or not when `!` is specified.
`-q[!]R` *substr*	Limit processed jobs to those containing *substr* as a substring of one of the recipients or not when `!` is specified.
`--q[!]S` *substr*	Limit processed jobs to those containing *substr* as a substring of the sender or not when `!` is specified.
`-r` *name*	An alternate and obsolete form of the `-f` flag.
`-R` *ret*	Identify the information you want returned if the message bounces; *ret* can be “HDRS” for headers only or “FULL” for headers plus body.
`-t`	Read message for recipients. `To:`, `Cc:`, and `Bcc:` lines are scanned for people to send to. The `Bcc:` line is deleted before transmission. Any addresses in the argument list is be suppressed.

	The `NoRecipientAction` Processing Option can be used to change the behaviour when no legal recipients are included in the message.
`-v`	Go into verbose mode. Alias expansions are announced, and so forth.
`-V` *envid*	The indicated *envid* is passed with the envelope of the message and returned if the message bounces.
`-X` *logfile*	Log all traffic in and out of `sendmail` in the indicated *logfile* for debugging mailer problems. This produces a lot of data very quickly and should be used sparingly.

Processing Options

There are a number of "random" options that can be set from a configuration file. Options are represented by a single character or by multiple character names. The syntax for the single character names of is:

`O`*xvalue*

This sets option *x* to be *value*. Depending on the option, *value* may be a string, an integer, a boolean (with legal values `t`, `T`, `f`, or `F`; the default is `TRUE`), or a time interval.

The multiple character or long names use this syntax:

`O` *Longname=argument*

This sets the option *Longname* to be *argument*. The long names are beneficial because they are easier to interpret than the single character names.

Not all processing options have single character names associated with them. In the list below the multiple character name is presented first followed by the single character syntax enclosed in parentheses.

`AliasFile (A`*file*`)`
Specify possible alias files.

`AliasWait (a` *N*`)`
If set, wait up to *N* minutes for an "@:@" entry to exist in the `aliases`(4) database before starting up. If it does not appear in *N* minutes, issue a warning. Defaults to 10 minutes.

`AllowBogusHELO`
Allow a `HELO SMTP` command that does not include a host name. By default this option is disabled.

`BadRcptThrottle=`*N*
If set and more than the specified number of recipients in a single SMTP envelope are rejected, sleep for one second after each rejected RCPT command.

`BlankSub (B`*c*`)`
Set the blank substitution character to *c*. Unquoted spaces in addresses are replaced by this character. Defaults to SPACE (that is, no change is made).

`CheckAliases (n)`
Validate the RHS of aliases when rebuilding the `aliases`(4) database.

`CheckpointInterval (C`*N*`)`
Checkpoints the queue every *N* (default `10`) addresses sent. If your system crashes during delivery to a large list, this prevents retransmission to any but the last *N* recipients.

`ClassFactor (z`*fact*`)`
The indicated factor *fact* is multiplied by the message class (determined by the `Precedence:` field in the user header and the `P` lines in the configuration file) and subtracted from the priority. Thus, messages with a higher `Priority:` are favored. Defaults to `1800`.

`ClientPortOptions`
Set client SMTP options. The options are key=value pairs. Known keys are:

`Addr` *Address Mask*	*Address Mask* defaults to `INADDR_ANY`. The address mask can be a numeric address in dot notation or a network name.
`Family`	Address family (defaults to INET).
`Listen`	Size of listen queue (defaults to 10).
`Port`	Name/number of listening port (defaults to `smtp`).
`RcvBufSize`	The size of the TCP/IP receive buffer.
`SndBufSize`	The size of the TCP/IP send buffer.
`Modifier`	Options (flags) for the daemon. Can be: `h` Use name of interface for `HELO` command. If `h` is set, the name corresponding to the outgoing interface address (whether chosen by means of the `Connection` parameter or the default) is used for the `HELO/EHLO` command.

`ColonOkInAddr`
If set, colons are treated as a regular character in addresses. If not set, they are treated as the introducer to the RFC 822 “group” syntax. This option is on for version 5 and lower configuration files.

`ConnectionCacheSize (k`*N*`)`
The maximum number of open connections that are be cached at a time. The default is `1`. This delays closing the current connection until either this invocation

of `sendmail` needs to connect to another host or it terminates. Setting it to `0` defaults to the old behavior, that is, connections are closed immediately.

`ConnectionCacheTimeout (K`*timeout*`)`
The maximum amount of time a cached connection is permitted to idle without activity. If this time is exceeded, the connection is immediately closed. This value should be small (on the order of ten minutes). Before `sendmail` uses a cached connection, it always sends a `NOOP` (no operation) command to check the connection; if this fails, it reopens the connection. This keeps your end from failing if the other end times out. The point of this option is to be a good network neighbor and avoid using up excessive resources on the other end. The default is five minutes.

`ConnectionRateThrottle`
The maximum number of connections permitted per second. After this many connections are accepted, further connections are delayed. If not set or <= 0, there is no limit.

`ControlSocketName`
Name of the control socket for daemon management. A running `sendmail` daemon can be controlled through this Unix domain socket. Available commands are: `help`, `restart`, `shutdown`, and `status`. The `status` command returns the current number of daemon children, the free disk space (in blocks) of the queue directory, and the load average of the machine expressed as an integer. If not set, no control socket is available. For the sake of security, this Unix domain socket must be in a directory which is accessible only by root; `/var/spool/mqueue/.smcontrol` is recommended for the socket name.

`DaemonPortOptions (O`*options*`)`
Set server SMTP options. The options are *key=value* pairs. Known keys are:

`Name`	User-definable name for the daemon (defaults to "`Daemon#`"); used for error messages and logging.
`Addr`	Address mask (defaults `INADDR_ANY`) The address mask may be a numeric address in dot notation or a network name.
`Family`	Address family (defaults to `INET`)
`Listen`	Size of listen queue (defaults to `10`)
`Modifier`	Options (flags) for the daemon; can be a sequence (without any delimiters) of: `a` Require authentication. `b` Bind to interface through which mail has been received. `c` Perform hostname canonification (`.cf`). `f` Require fully qualified hostname (`.cf`).

	`h`	Use name of interface for `HELO` command.
	`u`	Allow unqualified addresses (`.cf`).
	`C`	Do not perform hostname canonification.
	`E`	Disallow `ETRN` (see RFC 2476).
`Name`		User-definable name for the daemon (defaults to `Daemon#`); used for error messages and logging.
`Port`		Name/number of listening port (defaults to `smtp`)
`ReceiveSize`		The size of the TCP/IP receive buffer.
`SendSize`		The size of the TCP/IP send buffer.

`sendmail` listens on a new socket for each occurrence of the `DaemonPortOptions` option in a configuration file.

`DataFileBufferSize`
Set the threshold, in bytes, before a memory-bases queue data file becomes disk-based. The default is 4096 bytes

`DeadLetterDrop`
Defines the location of the system-wide dead.letter file, formerly hard-coded to `/var/tmp/dead.letter`. If this option is not set (the default), `sendmail` does not attempt to save to a system-wide `dead.letter` file in the event it cannot bounce the mail to the user or postmaster. Instead, it renames the `qf` file as it has in the past when the `dead.letter` file could not be opened.

`DefaultCharSet`
Set the default character set to use when converting unlabeled 8 bit input to MIME.

`DefaultUser (g`*gid*`) or (u`*uid*`)`
Set the default group ID for mailers to run in to *gid* or set the default userid for mailers to *uid*. Defaults to `1`. The value can also be given as a symbolic group or user name.

`DelayLA=`*LA*
When the system load average exceeds *LA*, `sendmail` sleeps for one second on most SMTP commands and before accepting connections.

`DeliverByMin=`*time*
Set minimum time for `Deliver By SMTP Service Extension` (RFC 2852). If `0`, no time is listed, if less than `0`, the extension is not offered, if greater than `0`, it is listed as minimum time for the `EHLO` keyword `DELIVERBY`.

`DeliveryMode (d`*x*`)`
Deliver in mode *x*. Legal modes are:

`i`	Deliver interactively (synchronously).
`b`	Deliver in background (asynchronously).

`d`	Deferred mode — database lookups are deferred until the actual queue run.
`q`	Just queue the message (deliver during queue run).

Defaults to `b` if no option is specified, `i` if it is specified but given no argument (that is, `Od` is equivalent to `Odi`).

`DialDelay`
If a connection fails, wait this many seconds and try again. Zero means "do not retry".

`DontBlameSendmail`
If set, override the file safety checks. This compromises system security and should not be used. See http://www.sendmail.org/tips/DontBlameSendmail.html for more information.

`DontExpandCnames`
If set, $[... $] lookups that do DNS-based lookups do not expand CNAME records.

`DontInitGroups`
If set, the `initgroups`(3C) routine is never be invoked. If you set this, agents run on behalf of users only have their primary (`/etc/passwd`) group permissions.

`DontProbeInterfaces`
If set, `sendmail` does not insert the names and addresses of any local interfaces into the `$=w class`. If set, you must also include support for these addresses, otherwise mail to addresses in this list bounce with a configuration error.

`DontPruneRoutes (R)`
If set, do not prune route-addr syntax addresses to the minimum possible.

`DoubleBounceAddress`
If an error occurs when sending an error message, send that "double bounce" error message to this address.

`EightBitMode (8)`
Use 8–bit data handling. This option requires one of the following keys. The key can selected by using just the first character, but using the full word is better for clarity.

`mimify`	Do any necessary conversion of `8BITMIME` to 7–bit.
`pass`	Pass unlabeled 8–bit input through as is.
`strict`	Reject unlabeled 8–bit input.

`ErrorHeader (E`*file/message*`)`
Append error messages with the indicated message. If it begins with a slash, it is assumed to be the pathname of a file containing a message (this is the recommended setting). Otherwise, it is a literal message. The error file might contain the name, email address, and/or phone number of a local postmaster who could provide assistance to end users. If the option is missing or `NULL`, or if it names a file which does not exist or which is not readable, no message is printed.

`ErrorMode (e`*x*`)`
Dispose of errors using mode *x*. The values for *x* are:

`e`	Mail back errors and give `0` exit status always.
`m`	Mail back errors.
`p`	Print error messages (default).
`q`	No messages, just give exit status.
`w`	Write back errors (mail if user not logged in).

`FallbackMXhost (V`*fallbackhost*`)`
If specified, the *fallbackhost* acts like a very low priority MX on every host. This is intended to be used by sites with poor network connectivity.

`FastSplit`
If set to a value greater than zero (the default is one), it suppresses the MX lookups on addresses when they are initially sorted, that is, for the first delivery attempt. This usually results in faster envelope splitting unless the MX records are readily available in a local DNS cache. To enforce initial sorting based on MX records set `FastSplit` to zero. If the mail is submitted directly from the command line, then the value also limits the number of processes to deliver the envelopes; if more envelopes are created they are only queued up and must be taken care of by a queue run. Since the default submission method is by way of SMTP (either from a MUA or by way of the MSP), the value of `FastSplit` is seldom used to limit the number of processes to deliver the envelopes.

`ForkEachJob (Y)`
If set, deliver each job that is run from the queue in a separate process. Use this option if you are short of memory, since the default tends to consume considerable amounts of memory while the queue is being processed.

`ForwardPath (J`*path*`)`
Set the path for searching for users' `.forward` files. The default is `$z/.forward`. Some sites that use the automounter may prefer to change this to `/var/forward/$u` to search a file with the same name as the user in a system directory. It can also be set to a sequence of paths separated by colons; `sendmail` stops at the first file it can successfully and safely open. For example, `/var/forward/$u:$z/.forward` searches first in `/var/forward/` *username* and then in *~username*`/.forward` (but only if the first file does not exist). Refer to the `NOTES` section for more information.

`HelpFile (H`*file*`)`
Specify the help file for SMTP.

`HoldExpensive (c)`
If an outgoing mailer is marked as being expensive, don't connect immediately.

`HostsFile`
Set the file to use when doing "file" type access of host names.

`HostStatusDirectory`
If set, host status is kept on disk between `sendmail` runs in the named directory tree. If a full path is not used, then the path is interpreted relative to the queue directory.

`IgnoreDots (i)`
Ignore dots in incoming messages. This is always disabled (that is, dots are always accepted) when reading SMTP mail.

`LogLevel (L`*n*`)`
Set the default log level to *n*. Defaults to `9`.

`(M`*x value*`)`
Set the macro *x* to *value*. This is intended only for use from the command line.

`MailboxDatabase`
Type of lookup to find information about local mail boxes, defaults to `pw` which uses `getpwnam`(3C). Other types can be introduced by adding them to the source code, see `libsm/mbdb.c` for details.

`MatchGECOS (G)`
Try to match recipient names using the GECOS field. This allows for mail to be delivered using names defined in the GECOS field in `/etc/passwd` as well as the login name.

`MaxDaemonChildren`
The maximum number of children the daemon permits. After this number, connections are rejected. If not set or <=0, there is no limit.

`MaxHopCount (h`*N*`)`
The maximum hop count. Messages that have been processed more than *N* times are assumed to be in a loop and are rejected. Defaults to `25`.

`MaxMessageSize`
The maximum size of messages that are accepted (in bytes).

`MaxMimeHeaderLength=`*M*`[/`*N*`]`
Sets the maximum length of certain MIME header field values to *M* characters. For some of these headers which take parameters, the maximum length of each parameter is set to *N* if specified. If */N* is not specified, one half of *M* is used. By default, these values are `0`, meaning no checks are done.

`MaxQueueChildren=`*N*
When set, this limits the number of concurrent queue runner processes to *N*. This helps to control the amount of system resources used when processing the queue. When there are multiple queue groups defined and the total number of queue runners for these queue groups would exceed `MaxQueueChildren` then the queue groups are not all run concurrently. That is, some portion of the queue groups run concurrently such that `MaxQueueChildren` is not be exceeded, while the remaining queue groups are run later (in round robin order). See `MaxRunnersPerQueue`.

`MaxQueueRunSize`
If set, limit the maximum size of any given queue run to this number of entries. This stops reading the queue directory after this number of entries is reached; job priority is not used. If not set, there is no limit.

`MaxRunnersPerQueue=`*N*
This sets the default maximum number of queue runners for queue groups. Up to *N* queue runners work in parallel on a queue group's messages. This is useful where the processing of a message in the queue might delay the processing of subsequent messages. Such a delay can be the result of non-erroneous situations such as a low bandwidth connection. The can be overridden on a per queue group basis by setting the `Runners` option. The default is `1` when not set.

`MeToo (M)`
Send to me too, even if I am in an alias expansion.

`MaxRecipientsPerMessage`
If set, allow no more than the specified number of recipients in an SMTP envelope. Further recipients receive a 452 error code and are deferred for the next delivery attempt.

`MinFreeBlocks (b`*N*`/`*M*`)`
Insist on at least *N* blocks free on the file system that holds the queue files before accepting email by way of SMTP. If there is insufficient space, `sendmail` gives a `452` response to the `MAIL` command. This invites the sender to try again later. The optional *M* is a maximum message size advertised in the `ESMTP EHLO` response. It is currently otherwise unused.

`MinQueueAge`
The amount of time a job must sit in the queue between queue runs. This allows you to set the queue run interval low for better responsiveness without trying all jobs in each run. The default value is 0.

`MustQuoteChars`
Characters to be quoted in a full name phrase. `&,;:\()[]` are quoted automatically.

`NiceQueueRun`
The priority of queue runners. See `nice`(1).

`NoRecipientAction`
Set action if there are no legal recipient files in the message. The legal values are:

`add-apparently-to`	Add an `Apparently-to:` header with all the known recipients (which may expose blind recipients).
`add-bcc`	Add an empty `Bcc:` header.
`add-to`	Add a `To:` header with all the known recipients (which may expose blind recipients).
`add-to-undisclosed`	Add a `To: undisclosed-recipients:` header.

`none`	Do nothing, leave the message as it is.

`OldStyleHeaders (o)`
Assume that the headers may be in old format, that is, spaces delimit names. This actually turns on an adaptive algorithm: if any recipient address contains a comma, parenthesis, or angle bracket, it is assumed that commas already exist. If this flag is not on, only commas delimit names. Headers are always output with commas between the names.

`OperatorChars or $o`
Defines the list of characters that can be used to separate the components of an address into tokens.

`PidFile`
Filename of the `pid` file. The default is `/var/run/sendmail.pid`. The filename is macro-expanded before it is opened.

`PostmasterCopy (P`*postmaster*`)`
If set, copies of error messages are sent to the named *postmaster*. Only the header of the failed message is sent. Since most errors are user problems, this is probably not a good idea on large sites, and arguably contains all sorts of privacy violations, but it seems to be popular with certain operating systems vendors.

`PrivacyOptions (p`*opt,opt,...*`)`
Set privacy options. Privacy is really a misnomer; many of these are just a way of insisting on stricter adherence to the SMTP protocol.

The `goaway` pseudo-flag sets all flags except `noreceipts`, `restrictmailq`, `restrictqrun`, `restrictexpand`, `noetrn`, and `nobodyreturn`. If `mailq` is restricted, only people in the same group as the queue directory can print the queue. If queue runs are restricted, only root and the owner of the queue directory can run the queue. The `restrict-expand` pseudo-flag instructs sendmail to drop privileges when the -bv option is given by users who are neither root nor the `TrustedUser` so users cannot read private aliases, forwards, or `:include:` files. It adds the `NonRootSafeAddr` to the "DontBlame-Sendmail" option to prevent misleading unsafe address warnings. It also overrides the `-v` (verbose) command line option to prevent information leakage. Authentication Warnings add warnings about various conditions that may indicate attempts to fool the mail system, such as using an non-standard queue directory.

The options can be selected from:

`authwarnings`	Put X-Authentication-Warning: headers in messages.
`goaway`	Disallow essentially all SMTP status queries.
`needexpnhelo`	Insist on `HELO` or `EHLO` command before `EXPN`.
`needmailhelo`	Insist on `HELO` or `EHLO` command before `MAIL`.
`needvrfyhelo`	Insist on `HELO` or `EHLO` command before `VRFY`.
`noetrn`	Disallow `ETRN` entirely.

`noexpn`	Disallow `EXPN` entirely.
`noreceipts`	Prevent return receipts.
`nobodyreturn`	Do not return the body of a message with DSNs.
`novrfy`	Disallow `VRFY` entirely.
`public`	Allow open access.
`restrictexpand`	Restrict `-bv` and `-v` command line flags.
`restrictmailq`	Restrict `mailq` command.
`restrictqrun`	Restrict `-q` command line flag.

`ProcessTitlePrefix` *string*
Prefix the process title shown on "`/usr/ucb/ps auxww`" listings with *string*. The string is macro processed.

`QueueDirectory (Q`*dir*`)`
Use the named *dir* as the queue directory.

`QueueFactor (q`*factor*`)`
Use `factor` as the multiplier in the map function to decide when to just queue up jobs rather than run them. This value is divided by the difference between the current load average and the load average limit (xflag) to determine the maximum message priority to be sent. Defaults to `600000`.

`QueueFileMode=`*mode*
Default permissions for queue files (octal). If not set, sendmail uses `0600` unless its real and effective `uid` are different in which case it uses `0644`.

`QueueLA (x`*LA*`)`
When the system load average exceeds *LA*, just queue messages (that is, do not try to send them). Defaults to eight times the number of processors online when `sendmail` starts.

`QueueSortOrder=`*algorithm*
Sets the algorithm used for sorting the queue. Only the first character of the value is used. Legal values are `host` (to order by the name of the first host name of the first recipient), `filename` (to order by the name of the queue file name), `time` (to order by the submission/creation time), `random` (to order randomly), `modification` (to order by the modification time of the `qf` file (older entries first)), and `priority` (to order by message priority). Host ordering makes better use of the connection cache, but may tend to process low priority messages that go to a single host over high priority messages that go to several hosts; it probably shouldn't be used on slow network links. Filename and modification time ordering saves the overhead of reading all of the queued items before starting the queue run. Creation (submission) time ordering is almost always a bad idea, since it allows large, bulk mail to go out before smaller, personal mail, but may have applicability on some hosts with very fast connections. Random is useful if several queue runners are started by hand which try to drain the same queue since odds are they are working on different parts of the queue at the same time. Priority ordering is the default.

`QueueTimeout (T`*rtime*`/`*wtime*`)`
Set the queue timeout to *rtime*. After this interval, messages that have not been successfully sent are returned to the sender. Defaults to five days (`5d`). The optional *wtime* is the time after which a warning message is sent. If it is missing or `0`, then no warning messages are sent.

`RecipientFactor (y`*fact*`)`
The indicated factor *fact* is added to the priority (thus *lowering* the priority of the job) for each recipient, that is, this value penalizes jobs with large numbers of recipients. Defaults to `30000`.

`RefuseLA (X`*LA*`)`
When the system load average exceeds *LA*, refuse incoming SMTP connections. Defaults to 12 times the number of processors online when `sendmail` starts.

`ResolverOptions (I)`
Tune DNS lookups.

`RetryFactor (Z`*fact*`)`
The indicated factor *fact* is added to the priority every time a job is processed. Thus, each time a job is processed, its priority is decreased by the indicated value. In most environments this should be positive, since hosts that are down are all too often down for a long time. Defaults to `90000`.

`RrtImpliesDsn`
If this option is set, a "Return-Receipt-To:" header causes the request of a DSN, which is sent to the envelope sender as required by RFC 1891, not to the address given in the header.

`RunAsUser`
If set, become this user when reading and delivering mail. Intended for use of firewalls where users do not have accounts.

`SafeFileEnvironment`
If set, `sendmail` does a `chroot` into this directory before writing files.

`SaveFromLine (f)`
Save Unix-style `From` lines at the front of headers. Normally they are assumed redundant and discarded.

`SendMimeErrors (j)`
If set, send error messages in MIME format (see RFC 2045 and RFC 1344 for details). If disabled, `sendmail` does not return the DSN keyword in response to an `EHLO` and does not do Delivery Status Notification processing as described in RFC 1891.

`ServiceSwitchFile`
Defines the path to the service-switch file. Since the service-switch file is defined in the Solaris operating environment this option is ignored.

`SevenBitInput (7)`
Strip input to seven bits for compatibility with old systems. This should not be necessary.

`SharedMemoryKey`
Key to use for shared memory segment; if not set (or `0`), shared memory is not be used. If this option is set, `sendmail` can share some data between different instances. For example, the number of entries in a queue directory or the available space in a file system. This allows for more efficient program execution, since only one process needs to update the data instead of each individual process gathering the data each time it is required.

`SingleLineFromHeader`
If set, `From:` lines that have embedded newlines are unwrapped onto one line.

`SingleThreadDelivery`
If this option and the `HostStatusDirectory` option are both set, use single thread deliveries to other hosts.

`SmtpGreetingMessage or $e`
The initial SMTP greeting message.

`StatusFile (S`*file*`)`
Log statistics in the named `file`. By default, this is `/etc/mail/sendmail.st`. As root, you must `touch`(1) this file to enable `mailstats`(1).

`SuperSafe (s)`
This option can be set to `True`, `False`, or `Interactive`. If set to `True`, `sendmail` is set to super-safe when running things, that is, always instantiate the queue file, even if you are going to attempt immediate delivery. `sendmail` always instantiates the queue file before returning control to the client under any circumstances. This should really always be set to `True`. The `Interactive` value has been introduced in `8.12` and can be used together with `DeliveryMode=i`. It skips some synchronization calls which are effectively doubled in the code execution path for this mode.

`TempFileMode (F`*mode*`)`
The file mode for queue files.

`Timeout (r`*timeouts*`)`
Timeout reads after `time` interval. The *timeouts* argument is a list of *keyword=value* pairs. All but *command* apply to client SMTP. For backward compatibility, a timeout with no *keyword*= part is set all of the longer values. The recognized timeouts and their default values, and their minimum values specified in RFC 1123 section 5.3.2 are:

`aconnect`
all connections for a single delivery attempt [0, unspecified]

`command`
command read [1h, 5m]

`connect`
initial connect [0, unspecified]

`control`
complete control socket transaction [2m, none]

`datablock`
data block read [1h, 3m]

`datafinal`
reply to final `.` in data [1h, 10m]

`datainit`
reply to `DATA` command [5m, 2m]

`fileopen`
file open [60sec, none]

`helo`
reply to `HELO` or `EHLO` command [5m, none]

`hoststatus`
host retry [30m, unspecified]

`iconnect`
first attempt to connect to a host [0, unspecified]

`ident`
`IDENT` protocol timeout [5s, none]

`initial`
wait for initial greeting message [5m, 5m]

`lhlo`
wait for reply to an LMTP LHLO command [2m, unspecified].

`mail`
reply to `MAIL` command [10m, 5m]

`misc`
reply to `NOOP` and `VERB` commands [2m, none]

`queuereturn`
undeliverable message returned [5d]

`queuewarn`
deferred warning [4h]

`quit`
reply to `QUIT` command [2m, none]

`rcpt`
reply to `RCPT` command [1h, 5m]

`resolver.retrans`
Resolver's retransmission time interval (in seconds) [varies]. Sets both `Timeout.resolver.retrans.first` and `Timeout.resolver.retrans.normal`.

`resolver.retrans.first`
Resolver's retransmission time interval (in seconds) for the first attempt to deliver a message [varies].

`resolver.retrans.normal`
Resolver's retransmission time interval (in seconds) for all look-ups except the first delivery attempt [varies].

`resolver.retry`
Number of times to retransmit a resolver query [varies]. Sets both `Timeout.resolver.retry.first` and `Timeout.resolver.retry.normal`.

`resolver.retry.first`
Number of times to retransmit a resolver query for the first attempt to deliver a message [varies].

`resolver.retry.normal`
Number of times to retransmit a resolver query for all look-ups except the first delivery attempt [varies].

`rset`
reply to `RSET` command [5m, none]

`TimeZoneSpec (t`*tzinfo*`)`
Set the local time zone info to *tzinfo*, for example, "PST8PDT ". Actually, if this is not set, the `TZ` environment variable is cleared (so the system default is used); if set but null, the user's `TZ` variable is used, and if set and non-null, the `TZ` variable is set to this value.

`TrustedUser`
The user parameter can be a user name (looked up in the passwd map) or a numeric user id. Trusted user for file ownership and starting the daemon. If set, generated alias databases and the control socket (if configured) are automatically owned by this user.

`TryNullMXList (w)`
If you are the "best" (that is, lowest preference) MX for a given host, you should normally detect this situation and treat that condition specially, by forwarding the mail to a UUCP feed, treating it as local, or whatever. However, in some cases (such as Internet firewalls) you may want to try to connect directly to that host as though it had no MX records at all. Setting this option causes `sendmail` to try this. The downside is that errors in your configuration are likely to be diagnosed as "host unknown" or "message timed out" instead of something more meaningful. This option is deprecated.

`UnixFromLine or $l`
The "From " line used when sending to files or programs.

`UnsafeGroupWrites`
If set, group-writable :include: and .forward files are considered "unsafe", that is, programs and files cannot be directly referenced from such files.

`UseErrorsTo (l)`
If there is an `Errors-To`: header, send error messages to the addresses listed there. They normally go to the envelope sender. Use of this option causes `sendmail` to violate RFC 1123. This option is not recommended and deprecated.

`UseMSP`
Use as mail submission program, that is., allow group writable queue files if the group is the same as that of a set-group-id `sendmail` binary.

`UserDatabaseSpec (U)`
Defines the name and location of the file containing User Database information.

`Verbose (v)`
Run in verbose mode. If this is set, `sendmail` adjusts the `HoldExpensive` and `DeliveryMode` options so that all mail is delivered completely in a single job so that you can see the entire delivery process. The `Verbose` option should *never* be set in the configuration file; it is intended for command line use only.

`XscriptFileBufferSize`
Set the threshold, in bytes, before a memory-bases queue transcript file becomes disk-based. The default is 4096 bytes.

If the first character of the user name is a vertical bar, the rest of the user name is used as the name of a program to pipe the mail to. It may be necessary to quote the name of the user to keep `sendmail` from suppressing the blanks from between arguments.

If invoked as `newaliases`, `sendmail` rebuilds the alias database, so long as the `/etc/mail/aliases*` files are owned by root *and* root has exclusive write permission. If invoked as `mailq`, `sendmail` prints the contents of the mail queue.

OPERANDS

address address of an intended recipient of the message being sent.

USAGE

See `largefile`(5) for the description of the behavior of `sendmail` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXIT STATUS

`sendmail` returns an exit status describing what it did. The codes are defined in `/usr/include/sysexits.h`.

`EX_OK`	Successful completion on all addresses.
`EX_NOUSER`	User name not recognized.
`EX_UNAVAILABLE`	Catchall. Necessary resources were not available.
`EX_SYNTAX`	Syntax error in address.
`EX_SOFTWARE`	Internal software error, including bad arguments.
`EX_OSERR`	Temporary operating system error, such as “cannot fork”.
`EX_NOHOST`	Host name not recognized.
`EX_TEMPFAIL`	Message could not be sent immediately, but was queued.

FILES

`dead.letter`	Unmailable text
`/etc/mail/aliases`	Mail aliases file (ASCII)

`/etc/mail/aliases.db`	Database of mail aliases (binary)
`/etc/mail/aliases.dir`	Database of mail aliases (binary)
`/etc/mail/aliases.pag`	Database of mail aliases (binary)
`/etc/mail/sendmail.cf`	Defines environment for `sendmail`
`/etc/mail/sendmail.cf`	Defines environment for `sendmail`
`/var/spool/clientmqueue/*`	Temporary files and queued mail
`/var/spool/mqueue/*`	Temporary files and queued mail
`~/.forward`	List of recipients for forwarding messages

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsndmu

SEE ALSO

`biff`(1B), `mail`(1), `mailq`(1), `mailx`(1), `nice`(1), `check-hostname`(1M), `check-permissions`(1M), `newaliases`(1M), `fork`(2), `getpwnam`(3C), `getusershell`(3C), `resolver`(3RESOLV), `aliases`(4), `hosts`(4), `shells`(4), `attributes`(5), `largefile`(5)

Postel, Jon, *Simple Mail Transfer Protocol*, RFC 821, Network Information Center, SRI International, Menlo Park, Calif., August 1982.

Crocker, Dave, *Standard for the Format of ARPA-Internet Text Messages*, RFC 822, Network Information Center, SRI International, Menlo Park, Calif., August 1982.

Costales, Bryan with Eric Allman, *sendmail, Second Edition*, O'Reilly & Associates, Inc., 1997.

NOTES

The `sendmail` program requires a fully qualified host name when starting. A script has been included to help verify if the host name is defined properly (see `check-hostname`(1M)).

The permissions and the ownership of several directories have been changed in order to increase security. In particular, access to `/etc/mail` and `/var/spool/mqueue` has been restricted.

Security restrictions have been placed users using `.forward` files to pipe mail to a program or redirect mail to a file. The default shell (as listed in `/etc/passwd`) of these users must be listed in `/etc/shells`. This restriction does not affect mail that is being redirected to another alias.

Additional restrictions have been put in place on `.forward` and `:include:` files. These files and the directory structure that they are placed in cannot be group- or world-writable (see`check-permissions`(1M)).

If you have interfaces that map to domains that have MX records that point to non-local destinations, you might need to enable the `DontProbeInterfaces` option to enable delivery to those destinations. In its default startup behavior, `sendmail` probes each interface and adds an interface's IP addresses, as well as any domains that those addresses map to, to its list of domains that are considered local. For domains thus added, being on the list of local domains is equivalent to having a 0-preference MX record, with `localhost` as the MX value. If this is not the result you want, enable `DontProbeInterfaces`.

NAME

setuname – change machine information

SYNOPSIS

setuname [-t] [-n *node*] [-s *name*]

DESCRIPTION

The setuname utility changes the parameter value for the system name and node name. Each parameter can be changed using setuname and the appropriate option.

Either or both the -s and -n options must be given when invoking setuname.

The system architecture may place requirements on the size of the system and network node name. The command will issue a fatal warning message and an error message if the name entered is incompatible with the system requirements.

OPTIONS

The following options are supported:

-n *node* — Changes the node name. *node* specifies the new network node name and can consist of alphanumeric characters and the special characters dash, underbar, and dollar sign.

-s *name* — Changes the system name. *name* specifies new system name and can consist of alphanumeric characters and the special characters dash, underbar, and dollar sign.

-t — Temporary change. No attempt will be made to create a permanent change.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu (32-bit) SUNWcsxu (64-bit)

SEE ALSO

attributes(5)

NOTES

setuname attempts to change the parameter values in two places: the running kernel and, as necessary per implementation, to cross system reboots. A temporary change changes only the running kernel.

NAME

sf880drd – Sun Fire 880 Dynamic Reconfiguration daemon

SYNOPSIS

`sf880drd`

DESCRIPTION

The Sun Fire 880 Dynamic Reconfiguration daemon, `sf880drd`, is part of the PCI and system bus hotplug framework. `sf880drd` starts at boot time. It has no configuration options and does not report any system status.

`sf880drd` implements the Sun Fire `880` console-less system administration (per-slot pushbuttons and LED status indicators). It also manages various aspects of CPU/memory hotplug.

FILES

`/usr/platform/SUNW,Sun-Fire-880/lib/sf880drd`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsfdr.u

SEE ALSO

`cfgadm`(1M), `cfgadm_pci`(1M), `cfgadm_sbd`(1M), `attributes`(5)

NAME

sftp-server – SFTP server subsystem

SYNOPSIS

/usr/lib/ssh/sftp-server

DESCRIPTION

`sftp-server` implements the server side of the SSH File Transfer Protocol as defined in the IETF `draft-ietf-secsh-filexfer`.

`sftp-server` is a subsystem for `sshd`(1M) and must not be run directly. There are no options or config settings.

To enable the `sftp-server` subsystem for `sshd` add the following to `/etc/ssh/sshd_config`:

```
Subsystem   sftp     /usr/lib/ssh/sftp-server
```

See `sshd_config`(4) for a description of the format and contents of that file.

There is no relationship between the protocol used by `sftp-server` and the FTP protocol (RFC 959) provided by `in.ftpd`.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

FILES

`/usr/lib/sftp-server`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsshdu

SEE ALSO

`sftp`(1), `ssh`(1), `ssh-add`(1), `ssh-keygen`(1), `sshd`(1M), `sshd_config`(4), `attributes`(5)

To view license terms, attribution, and copyright for OpenSSH, the default path is `/var/sadm/pkg/SUNWsshdr/install/copyright`. If the Solaris operating environment has been installed anywhere other than the default, modify the given path to access the file at the installed location.

AUTHOR

Markus Friedl

NAME

share – make local resource available for mounting by remote systems

SYNOPSIS

share [-F *FSType*] [-o *specific_options*] [-d *description*] [*pathname*]

DESCRIPTION

The share command exports, or makes a resource available for mounting, through a remote file system of type *FSType*. If the option -F *FSType* is omitted, the first file system type listed in /etc/dfs/fstypes is used as default. For a description of NFS specific options, see share_nfs(1M). *pathname* is the pathname of the directory to be shared. When invoked with no arguments, share displays all shared file systems.

OPTIONS

-F *FSType*
: Specify the filesystem type.

-o *specific_options*
: The *specific_options* are used to control access of the shared resource. (See share_nfs(1M) for the NFS specific options.) They may be any of the following:

  rw
  : *pathname* is shared read/write to all clients. This is also the default behavior.

  rw=*client*[:*client*]...
  : *pathname* is shared read/write only to the listed clients. No other systems can access *pathname*.

  ro
  : *pathname* is shared read-only to all clients.

  ro=*client*[:*client*]...
  : *pathname* is shared read-only only to the listed clients. No other systems can access *pathname*.

  Separate multiple options with colons.

-d *description*
: The -d flag may be used to provide a description of the resource being shared.

EXAMPLES

EXAMPLE 1 Sharing a read-only filesystem

This line will share the /disk file system read-only at boot time.

```
share -F nfs -o ro /disk
```

EXAMPLE 2 Invoking multiple options

The following command shares the filesystem /export/manuals, with members of the netgroup having read-only access and users on the specified host having read-write access.

EXAMPLE 2 Invoking multiple options *(Continued)*

```
share -F nfs -o ro=netgroup_name:rw=hostname /export/manuals
```

FILES

`/etc/dfs/dfstab`	list of `share` commands to be executed at boot time
`/etc/dfs/fstypes`	list of file system types, NFS by default
`/etc/dfs/sharetab`	system record of shared file systems

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`mountd`(1M), `nfsd`(1M), `share_nfs`(1M), `shareall`(1M), `unshare`(1M), `attributes`(5)

NOTES

Export (old terminology): file system sharing used to be called exporting on SunOS 4.x, so the `share` command used to be invoked as `exportfs`(1B) or `/usr/sbin/exportfs`.

If `share` commands are invoked multiple times on the same filesystem, the last `share` invocation supersedes the previous—the options set by the last `share` command replace the old options. For example, if read-write permission was given to `usera` on `/somefs`, then to give read-write permission also to `userb` on `/somefs`:

```
example% share -F nfs -o rw=usera:userb /somefs
```

This behavior is not limited to sharing the root filesystem, but applies to all filesystems.

NAME

shareall, unshareall – share, unshare multiple resources

SYNOPSIS

shareall [-F *FSType* [,*FSType*...]] [-| *file*]

unshareall [-F *FSType* [,*FSType*...]]

DESCRIPTION

When used with no arguments, `shareall` shares all resources from `file`, which contains a list of `share` command lines. If the operand is a hyphen (–), then the `share` command lines are obtained from the standard input. Otherwise, if neither a `file` nor a hyphen is specified, then the file `/etc/dfs/dfstab` is used as the default.

Resources may be shared by specific file system types by specifying the file systems in a comma-separated list as an argument to `-F`.

`unshareall` unshares all currently shared resources. Without a `-F` flag, it unshares resources for all distributed file system types.

OPTIONS

`-F` *FSType* Specify file system type. Defaults to the first entry in `/etc/dfs/fstypes`.

FILES

`/etc/dfs/dfstab`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`share`(1M), `unshare`(1M), `attributes`(5)

NAME

share_nfs – make local NFS file systems available for mounting by remote systems

SYNOPSIS

share [-d *description*] [-F nfs] [-o *specific_options*] *pathname*

DESCRIPTION

The `share` utility makes local file systems available for mounting by remote systems.

If no argument is specified, then `share` displays all file systems currently shared, including NFS file systems and file systems shared through other distributed file system packages.

OPTIONS

The following options are supported:

`-d` *description*
: Provide a comment that describes the file system to be shared.

`-F nfs`
: Share NFS file system type.

`-o` *specific_options*
: Specify *specific_options* in a comma-separated list of keywords and attribute-value-assertions for interpretation by the file-system-type-specific command. If *specific_options* is not specified, then by default sharing will be read-write to all clients. *specific_options* can be any combination of the following:

  `aclok`
  : Allows the NFS server to do access control for NFS Version 2 clients (running SunOS 2.4 or earlier). When `aclok` is set on the server, maximal access is given to all clients. For example, with `aclok` set, if anyone has read permissions, then everyone does. If `aclok` is not set, minimal access is given to all clients.

  `anon=`*uid*
  : Set *uid* to be the effective user ID of unknown users. By default, unknown users are given the effective user ID `UID_NOBODY`. If *uid* is set to `−1`, access is denied.

  `index=file`
  : Load `file` rather than a listing of the directory containing this file when the directory is referenced by an NFS URL.

  `log=tag`
  : Enables NFS server logging for the specified file system. The optional tag determines the location of the related log files. The `tag` is defined in `etc/nfs/nfslog.conf`. If no `tag` is specified, the default values associated with the “global” `tag` in `etc/nfs/nfslog.conf` will be used.

`nosub`
Prevents clients from mounting subdirectories of shared directories. For example, if `/export` is shared with the `nosub` option on server *fooey* then a NFS client will not be able to do:

```
mount -F nfs fooey:/export/home/mnt
```

`nosuid`
By default, clients are allowed to create files on the shared file system with the setuid or setgid mode enabled. Specifying `nosuid` causes the server file system to silently ignore any attempt to enable the setuid or setgid mode bits.

`public`
Moves the location of the public file handle from `root` (`/`) to the exported directory for WebNFS-enabled browsers and clients. This option does not enable WebNFS service; WebNFS is always on. Only one file system per server may use this option. Any other option, including the `-ro=list` and `-rw=list` options can be included with the `public` option.

`ro`
Sharing will be read-only to all clients.

`ro=`*access_list*
Sharing will be read-only to the clients listed in *access_list*; overrides the `rw` suboption for the clients specified. See *access_list* below.

`root=`*access_list*
Only root users from the hosts specified in *access_list* will have root access. See *access_list* below. By default, no host has root access, so root users are mapped to an anonymous user ID (see the `anon=`*uid* option described above). Netgroups can be used if the file system shared is using UNIX authentication (`AUTH_SYS`).

`rw`
Sharing will be read-write to all clients.

`rw=`*access_list*
Sharing will be read-write to the clients listed in *access_list*; overrides the `ro` suboption for the clients specified. See *access_list* below.

`sec=`*mode*[`:`*mode*]. . .
Sharing will use one or more of the specified security modes. The *mode* in the `sec=`*mode* option must be a node name supported on the client. If the `sec=` option is not specified, the default security mode used is `AUTH_SYS`. Multiple `sec=` options can be specified on the command line, although each mode can appear only once. The security modes are defined in `nfssec`(5).

Each `sec=` option specifies modes that apply to any subsequent `window=, rw, ro, rw=, ro=` and `root=` options that are provided before another `sec=`option. Each additional `sec=` resets the security mode context, so that more `window=, rw, ro, rw=, ro=` and `root=` options can be supplied for additional modes.

`sec=`*none*
If the option `sec=`*none* is specified when the client uses `AUTH_NONE`, or if the client uses a security mode that is not one that the file system is shared with, then the credential of each NFS request is treated as unauthenticated. See the `anon=`*uid* option for a description of how unauthenticated requests are handled.

`secure`
This option has been deprecated in favor of the `sec=`*dh* option.

`window=`*value*
When sharing with `sec=`*dh*, set the maximum life time (in seconds) of the RPC request's credential (in the authentication header) that the NFS server will allow. If a credential arrives with a life time larger than what is allowed, the NFS server will reject the request. The default value is 30000 seconds (8.3 hours).

access_list

The *access_list* argument is a colon-separated list whose components may be any number of the following:

hostname
The name of a host. With a server configured for DNS or LDAP naming in the `nsswitch` "hosts" entry, any hostname must be represented as a fully qualified DNS or LDAP name.

netgroup
A netgroup contains a number of hostnames. With a server configured for DNS or LDAP naming in the

	`nsswitch` "hosts" entry, any hostname in a netgroup must be represented as a fully qualified DNS or LDAP name.

domain name suffix	To use domain membership the server must use DNS or LDAP to resolve hostnames to IP addresses; that is, the "hosts" entry in the `/etc/nsswitch.conf` must specify "dns" or “ldap” ahead of "nis" or "nisplus", since only DNS and LDAP return the full domain name of the host. Other name services like NIS or NIS+ cannot be used to resolve hostnames on the server because when mapping an IP address to a hostname they do not return domain information. For example,
	
	NIS or NIS+ 129.144.45.9 --> "myhost
	
	DNS or LDAP 129.144.45.9 --> "myhost.mydomain.mycompany.com"
	
	The domain name suffix is distinguished from hostnames and netgroups by a prefixed dot. For example,
	
	`rw=.mydomain.mycompany.com`
	
	A single dot can be used to match a hostname with no suffix. For example,
	
	`rw=.`
	
	will match "mydomain" but not "mydomain.mycompany.com". This feature can be used to match hosts resolved through NIS and NIS+ rather than DNS and LDAP.

network	The network or subnet component is preceded by an at-sign (`@`). It can be either a name or a dotted address. If a name, it will be converted to a dotted address by `getnetbyname`(3SOCKET). For example, `=@mynet`would be equivalent to: `=@129.144` or `=@129.144.0.0`The network prefix assumes an octet aligned netmask determined from the zero octets in the low-order part of the address. In the case where network prefixes are not byte-aligned, the

syntax will allow a mask length to be specified explicitly following a slash (/) delimiter. For example,

`=@theothernet/17` or `=@129.144.132/22`where the mask is the number of leftmost contiguous significant bits in the corresponding IP address.

A prefixed minus sign (–) denies access to that component of *access_list*. The list is searched sequentially until a match is found that either grants or denies access, or until the end of the list is reached. For example, if host "terra" is in the "engineering" netgroup, then

`rw=-terra:engineering`will deny access to terra but

`rw=engineering:-terra`will grant access to terra.

OPERANDS

The following operands are supported:

pathname The pathname of the file system to be shared.

EXAMPLES

EXAMPLE 1 Sharing A File System With Logging Enabled

The following example shows the `/export` file system shared with logging enabled:

```
example% share -o log /export
```

The default global logging parameters are used since no tag identifier is specified. The location of the log file, as well as the necessary logging work files, is specified by the global entry in `/etc/nfs/nfslog.conf`. Note that the `nfslogd`(1M) daemon will run only if at least one file system entry in `/etc/dfs/dfstab` is shared with logging enabled upon starting or rebooting the system. Simply sharing a file system with logging enabled from the command line will not start the `nfslogd`(1M).

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

FILES

`/etc/dfs/fstypes`	list of system types, NFS by default
`/etc/dfs/sharetab`	system record of shared file systems
`/etc/nfs/nfslogtab`	system record of logged file systems
`/etc/nfs/nfslog.conf`	logging configuration file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnfssu

SEE ALSO

mount(1M), mountd(1M), nfsd(1M), nfslogd(1M), share(1M), unshare(1M), getnetbyname(3SOCKET), nfslog.conf(4), netgroup(4), attributes(5), nfssec(5)

NOTES

If the sec= option is presented at least once, all uses of the window=, rw, ro, rw=, ro= and root= options must come after the first sec= option. If the sec= option is not presented, then sec=*sys* is implied.

If one or more explicit sec= options are presented, *sys* must appear in one of the options mode lists for accessing using the AUTH_SYS security mode to be allowed. For example:

```
share -F nfs /var
```

```
share -F nfs -o sec=sys /var
```

will grant read-write access to any host using AUTH_SYS, but

```
share -F nfs -o sec=dh /var
```

will grant no access to clients that use AUTH_SYS.

Unlike previous implementations of share_nfs(1M), access checking for the window=, rw, ro, rw=, and ro= options is done per NFS request, instead of per mount request.

Combining multiple security modes can be a security hole in situations where the ro= and rw= options are used to control access to weaker security modes. In this example,

```
share -F nfs -o sec=dh,rw,sec=sys,rw=hosta /var
```

an intruder can forge the IP address for hosta (albeit on each NFS request) to side-step the stronger controls of AUTH_DES. Something like:

```
share -F nfs -o sec=dh,rw,sec=sys,ro /var
```

is safer, because any client (intruder or legitimate) that avoids AUTH_DES will only get read-only access. In general, multiple security modes per share command should only be used in situations where the clients using more secure modes get stronger access than clients using less secure modes.

If rw=, and ro= options are specified in the same sec= clause, and a client is in both lists, the order of the two options determines the access the client gets. If client hosta is in two netgroups - group1 and group2 - in this example, the client would get read-only access:

```
share -F nfs -o ro=group1,rw=group2 /var
```

In this example hosta would get read-write access:

```
share -F nfs -o rw=group2,ro=group1 /var
```

If within a sec= clause, both the ro and rw= options are specified, for compatibility, the order of the options rule is not enforced. All hosts would get read-only access, with the exception to those in the read-write list. Likewise, if the ro= and rw options are specified, all hosts get read-write access with the exceptions of those in the read-only list.

The ro= and rw= options are guaranteed to work over UDP and TCP but may not work over other transport providers.

The root= option with AUTH_SYS is guaranteed to work over UDP and TCP but may not work over other transport providers.

The root= option with AUTH_DES is guaranteed to work over any transport provider.

There are no interactions between the root= option and the rw, ro, rw=, and ro= options. Putting a host in the root list does not override the semantics of the other options. The access the host gets is the same as when the root= options is absent. For example, the following share command will deny access to hostb:

```
share -F nfs -o ro=hosta,root=hostb /var
```

The following will give read-only permissions to hostb:

```
share -F nfs -o ro=hostb,root=hostb /var
```

The following will give read-write permissions to hostb:

```
share -F nfs -o ro=hosta,rw=hostb,root=hostb /var
```

If the file system being shared is a symbolic link to a valid pathname, the canonical path (the path which the symbolic link follows) will be shared. For example, if /export/foo is a symbolic link to /export/bar (/export/foo -> /export/bar), the following share command will result in /export/bar as the shared pathname (and not /export/foo).

```
example# share -F nfs /export/foo
```

Note that an NFS mount of server:/export/foo will result in server:/export/bar really being mounted.

This line in the /etc/dfs/dfstab file will share the /disk file system read-only at boot time:

```
share -F nfs -o ro /disk
```

Note that the same command entered from the command line will not share the /disk file system unless there is at least one file system entry in the /etc/dfs/dfstab file. The mountd(1M) and nfsd(1M) daemons only run if there is a file system entry in /etc/dfs/dfstab when starting or rebooting the system.

NAME

showmount – show all remote mounts

SYNOPSIS

/usr/sbin/showmount [-ade] [*hostname*]

DESCRIPTION

showmount lists all the clients that have remotely mounted a filesystem from *host*. This information is maintained by the mountd(1M) server on *host*, and is saved across crashes in the file /etc/rmtab. The default value for *host* is the value returned by hostname(1).

OPTIONS

-a Print all remote mounts in the format:

```
hostname : directory
```

where hostname is the name of the client, and directory is the root of the file system that has been mounted.

-d List directories that have been remotely mounted by clients.

-e Print the list of shared file systems.

FILES

/etc/rmtab

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnfscu

SEE ALSO

hostname(1), mountd(1M), attributes(5)

Solaris 9 Installation Guide

BUGS

If a client crashes, its entry will not be removed from the list of remote mounts on the server.

NAME

showrev – show machine, software revision, and patch revision information

SYNOPSIS

/usr/bin/showrev [-a] [-p | -p -R *root_path*] [-w] [-c *command*] [-s *hostname*]

DESCRIPTION

`showrev` displays revision information for the current hardware and software. With no arguments, `showrev` shows the system revision information including hostname, hostid, release, kernel architecture, application architecture, hardware provider, domain, and kernel version.

If a command is supplied with the `-c` option, `showrev` shows the `PATH` and `LD_LIBRARY_PATH` and finds out all the directories within the `PATH` that contain it. For each file found, its file type, revision, permissions, library information, and checksum are printed as well.

OPTIONS

The following options are supported:

`-a` Print all system revision information available. Window system and patch information are added.

`-c` *command* Print the revision information about *command.*

`-p` Print only the revision information about patches.

`-R` *root_path* Define the full path name of a directory to use as the *root_path*. By specifying the root path, `showrev` retrieves the revision information about the patch from package system information files located under a directory tree starting at *root_path*. The *root_path* can be specified when retrieving installed patch information in a client from a server, for example, `/export/root/client1`.

`-s` *hostname* Perform this operation on the specified *hostname*. The `-s` operation completes correctly only when *hostname* is running Solaris 2.5 or compatible versions.

`-w` Print only the OpenWindows revision information.

OUTPUT

Varies, based on flags passed. If no flags are passed, output similar to the following appears:

```
Hostname: system1
    Hostid: 7233808e
    Release: 5.9
    Kernel architecture: sun4u
    Application architecture: sparc
    Hardware provider: Sun_Microsystems
    Domain: a.network.COM
    Kernel version: SunOS 5.9 generic February 2001
```

EXIT STATUS

The following error values are returned:

`0` Successful completion.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWadmc

SEE ALSO

`arch`(1), `ldd`(1), `mcs`(1), `sum`(1), `patchadd`(1M), `attributes`(5)

BUGS

For the `-s` option to work when `hostname` is running a version of Solaris prior to 2.5, the Solstice AdminSuite must be installed on `hostname`.

NAME

shutdown – shut down system, change system state

SYNOPSIS

/usr/sbin/shutdown [-y] [-g *grace-period*] [-i *init-state*] [*message*]

DESCRIPTION

`shutdown` is executed by the super user to change the state of the machine. In most cases, it is used to change from the multi-user state (state 2) to another state.

By default, `shutdown` brings the system to a state where only the console has access to the operating system. This state is called single-user.

Before starting to shut down daemons and killing processes, `shutdown` sends a warning message and, by default, a final message asking for confirmation. *message* is a string that is sent out following the standard warning message "The system will be shut down in . . ." If the string contains more than one word, it should be contained within single (') or double (") quotation marks.

The warning message and the user provided *message* are output when there are 7200, 3600, 1800, 1200, 600, 300, 120, 60, and 30 seconds remaining before `shutdown` begins. See `EXAMPLES`.

System state definitions are:

state 0	Stop the operating system.
state 1	State 1 is referred to as the administrative state. In state 1 file systems required for multi-user operations are mounted, and logins requiring access to multi-user file systems can be used. When the system comes up from firmware mode into state 1, only the console is active and other multi-user (state 2) services are unavailable. Note that not all user processes are stopped when transitioning from multi-user state to state 1.
state s, S	State s (or S) is referred to as the single-user state. All user processes are stopped on transitions to this state. In the single-user state, file systems required for multi-user logins are unmounted and the system can only be accessed through the console. Logins requiring access to multi-user file systems cannot be used.
state 5	Shut the machine down so that it is safe to remove the power. Have the machine remove power, if possible. The `rc0` procedure is called to perform this task.
state 6	Stop the operating system and reboot to the state defined by the `initdefault` entry in `/etc/inittab`. The `rc6` procedure is called to perform this task.

OPTIONS

`-y`	Pre-answer the confirmation question so the command can be run without user intervention.
`-g` *grace-period*	Allow the super user to change the number of seconds from the 60-second default.

`-i` *init-state* — If there are warnings, *init-state* specifies the state `init` is to be in. By default, system state 's' is used.

EXAMPLES

EXAMPLE 1 Using `shutdown`

In the following example, `shutdown` is being executed on host `foo` and is scheduled in 120 seconds. The warning message is output 2 minutes, 1 minute, and 30 seconds before the final confirmation message.

```
example# shutdown -i S -g 120 "===== disk replacement ====="
Shutdown started.   Tue Jun   7  14:51:40 PDT  1994

Broadcast Message from root (pts/1) on foo Tue Jun  7 14:51:41. . .
The system will be shut down in 2 minutes
===== disk replacement =====
Broadcast Message from root (pts/1) on foo Tue Jun  7 14:52:41. . .
The system will be shut down in 1 minutes
===== disk replacement =====
Broadcast Message from root (pts/1) on foo Tue Jun  7 14:53:41. . .
The system will be shut down in 30 seconds
===== disk replacement =====
Do you want to continue? (y or n):
```

FILES

`/etc/inittab` controls process dispatching by `init`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`boot`(1M), `halt`(1M), `init`(1M), `killall`(1M), `reboot`(1M), `ufsdump`(1M), `init.d`(4), `inittab`(4), `nologin`(4), `attributes`(5)

NOTES

When a system transitions down to the `S` or `s` state, the `/etc/nologin` file (see `nologin`(4)) is created. Upon subsequent transition to state 2 (multi-user state), this file is removed by a script in the `/etc/rc2.d` directory.

NAME

slpd – Service Location Protocol Daemon

SYNOPSIS

/usr/lib/inet/slpd [-f *configuration-file*]

DESCRIPTION

The `slpd` daemon provides common server functionality for the Service Location Protocol ("SLP") versions 1 and 2, as defined by IETF in *RFC 2165* and *RFC 2608*. SLP provides a scalable framework for the discovery and selection of network services.

`slpd` provides the following framework services:

Directory Agent	This service automatically caches service advertisements from service agents to provide them to user agents, and makes directory agent advertisements of its services. This service is optional. `slpd` does not provide directory agent service by default. Directory agents are not databases, and they do not need to be maintained.
Service Agent Server	All service agents on the local host register and deregister with this server. This service responds to all requests for services, and forwards registrations to directory agents. By default, `slpd` is a service agent server.
Passive Directory Agent Discovery	This service listens for directory agent advertisements and maintains a table of active directory agents. When a user agent wishes to discover a directory agent, it can simply query `slpd`, obviating the need to perform discovery by means of multicast. By default, `slpd` performs this service.
Proxy Registration	This service can act as a proxy service agent for services that cannot register themselves. `slpd` reads the proxy registration file for information on services it is to proxy. By default, no services are registered by proxy.

All configuration options are available from the configuration file. `slpd` reads its configuration file upon startup.

Stop and start the `slpd` daemon by using the startup script: `/etc/init.d/slpd`. Use the command `/etc/init.d/slpd stop` to stop the `slpd` daemon. Use the command `/etc/init.d/slpd start` to start it.

The file `/etc/inet/slp.conf` must exist before the startup script can start the daemon. Only the example file `/etc/inet/slp.conf.example` is present by default. To enable SLP, copy `/etc/inet/slp.conf.example` to `/etc/inet/slp.conf`.

OPTIONS

The following options are supported:

`-f` *configuration-file* Specify an alternate configuration file

EXAMPLES

EXAMPLE 1 Stopping the `slpd` daemon

The following command stops the `slpd` daemon:

```
example# /etc/init.d/slpd stop
```

EXAMPLE 2 Restarting the `slpd` daemon

The following command restarts the `slpd` daemon:

```
example# /etc/init.d/slpd start
```

FILES

`/etc/inet/slp.conf` The default configuration file

`slpd.reg` The proxy registration file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWslpu, SUNWslpr
CSI	Enabled
Interface Stability	Evolving

SEE ALSO

`slp_api`(3SLP), `slp.conf`(4), `slpd.reg`(4), `attributes`(5), `slp`(7P)

SYSADV4-tt

Guttman, E., Perkins, C., Veizades, J., and Day, M., *RFC 2608, Service Location Protocol, Version 2*, The Internet Society, June 1999.

NAME

smartcard – configure and administer a smartcard

SYNOPSIS

smartcard `-c admin [-a` *application*`] [`*propertyname*`...]`

smartcard `-c admin [-a` *application*`] [-x { add|delete|modify}` *propertyname=value*`...]`

smartcard `-c admin -t service -j` *classname* `-x { add|delete|modify}`

smartcard `-c admin -t terminal -j` *classname* `-d` *device* `-r` *userfriendlyreadername* `-n` *readername* `-x { add|delete|modify} [-R]`

smartcard `-c admin -t debug -j` *classname* `-l` *level* `-x { add|delete|modify}`

smartcard `-c admin -t override -x { add|delete|modify}` *propertyname=value*

smartcard `-c admin -I -k` *keytype* `-i` *filename*

smartcard `-c admin -E -k` *keytype* `-o` *filename*

smartcard `-c load -A` *aid* `[-r` *userfriendlyreadername*`] -P` *pin* `[-s` *slot*`] [-i` *inputfile*`] [-p` *propfile*`] [-v] [`*propertyname=value*`...]`

smartcard `-c load -u -P` *pin* `[-A` *aid*`] [-r` *userfriendlyreadername*`] [-s` *slot*`] [-v]`

smartcard `-c bin2capx -T` *cardname* `[-i` *inputfile*`] [-o` *outputfile*`] [-p` *propfile*`] [-I` *anothercapxfile*`] [-v] [`*propertyname=value*`...]`

smartcard `-c init -A` *aid* `[-r` *readername*`] [-s` *slot*`] -L`

smartcard `-c init -A` *aid* `[-r` *readername*`] -P` *pin* `[-s` *slot*`]` `[`*propertyname=value*`...]`

smartcard `-c enable`

smartcard `-c disable`

DESCRIPTION

The `smartcard` utility is used for all configurations related to a smartcard. It comprises the following subcommands:

1. Administration of OCF properties. (`-c admin`)

   This subcommand is used to list and modify any of the OCF properties. With no arguments it will list all the current properties. It can only be executed by root. Some OCF properies are:

`defaultcard`	# default card for an application
`defaultreader`	# default reader for an application
`authmechanism`	# authentication mechanism to
`validcards`	# list of cards valid for an application

A complete listing can be obtained by using the smartcard command as described in the EXAMPLES section.

2. Loading and Unloading of applets from the smartcard (`-c load`) and performing initial configuration of a non-Java card.

   This subcommand administers the applets or properties on a smartcard. It can be used to load or unload applets and/or properites to and from a smartcard. The applet is a Java class file that has been run through a convertor to make the byte code JavaCard-compliant. This command can be used to load both an applet file in the standard format or a file converted to the `capx` format. If no `-r` option is specified, the loader tries to load to any connected reader, provided it has already been inserted using the `smartcard -c admin` command.

3. Converting card applets or properties to the `capx` format (`-c bin2capx`)

   This subcommand is used to convert a Java card applet or properties into a new format called `capx` before downloading it onto the smartcard. Converting to this format enables the applet developer to add applet-specific information that is useful during the downloading process and identifies the applet.

   In the following example,

   ```
   smartcard -c bin2capx -i cyberflex.bin \
   -T CyberFlex aidto-000102030405060708090A0B0C0D0E0F fileID=2222 \
   instanceID=2223 and more.
   ```

   if no output file is specified, a default file with the name *input_filename*.`capx` is created in the current directory. The mandatory `-T` option requires the user to specify the card name for which the `capx` file is being generated.

   The following example

   ```
   smartcard -c bin2capx -T IButton
   ```

   tells the loader eventually that the `capx` file contains the binary for `IButton`. A single `capx` file can hold binaries for multiple cards (1 per card.) Users can, for example, hold binary files for both CyberFlex and IButton in the same capx file as follows:

   ```
   smartcard -c bin2capx -T IButton -i IButton.jib -o file.capx
   ```

   In the following example,

   ```
   smartcard -c bin2capx -T CyberFlex -i cyberflex.bin \
   -l file.capx -o file.capx
   ```

   the `-l` option is used to provide an already-generated `capx` file. The output is directed to the same `capx` file, resulting in `capx` file holding binaries for both cards.

4. Personalizing the smartcard (`-c init`)

This subcommand is used to set user-specific information required by an applet on a smartcard. For example, the Sun applet requires a user name to be set on the card. This subcommand is also used to personalize information for non-Java cars.

5. Enabling and disabling the smartcard desktop login (`-c {enable | disable}`)

OPTIONS

The following options are supported:

`-a` *application*	Specify application name for the configuration parameter. Parameters may differ depending on the application. If no application name is specified, then `ocf` is the default application.
`-A` *aid*	Specify a unique alphanumeric string that identifies the applet. The *aid* argument must be a minimum of 5 characters and can be a maximum of 16 characters in length. If an applet with an identical *aid* already exists on the card, a load will result in an error.
`-c`	Specify subcommand name. Valid options are: `admin`, `load`, `bin2capx`, `init`, `enable` and `disable`.
`-d` *device*	Specify device on which the reader is connected (for example, `/dev/cua/a`).
`-D`	Disable a system from using smartcards.
`-E`	Export the keys to a file.
`-i` *filename*	Specify input file name.
`-I`	Import from a file.
`-j` *classname*	Specify fully-qualified class name.
`-k` *keytype*	Specify type of key (for example, `challenge_response`, `pki`.)
`-l`	Specify debug level (0–9), signifying level of debug information displayed.
`-L`	List all properties configurable in an applet.
`-n` *readername*	Specify reader name as required by the driver.
`-o` *filename*	Specify output file name.
`-p` *propfile*	Specify properties file name. This file could contain a list of property names and value pairs, in the format *propertyname=value*.
`-P` *pin*	Specify pin used to validate to the card.
`-r` *userfriendlyreadername*	Specify user-defined reader name where the card to be initialized is inserted.

`-R`	Restart the `ocf` server.
`-s` *slot*	Specify slot number. If a reader has multiple slots, this option specifies which slot to use for initialization. If a reader has only one slot, this option is not required. If no slot number is specified, by default the fist slot of the reader is used.
`-t`	Specify type of property being updated. The valid values are:
	`service` Updating a card service provider details.
	`terminal` Updating a card reader provider details.
	`debug` OCF trace level.
	`override` Override a system property of the same name.
`-T` *cardname*	Specify card name.
`-u`	Unload the applet specified by the application ID from the card. If no application ID is specified, all applets are unloaded from the card.
`-v`	Verbose mode (displays helpful messages).
`-x`	Specify action to be taken. Valid values are: `add`, `delete`, or `modify`.

EXAMPLES

EXAMPLE 1 Viewing the Values of All Properties

Enter the following command to view the values of all the properties that are set:

```
% smartcard -c admin
```

EXAMPLE 2 Viewing the Values of Specific Properties

Enter the following command to view the values of specific properties:

```
% smartcard -c admin language country
```

EXAMPLE 3 Adding a Card Service

Enter the following command to add a card service factory for a CyberFlex card, available in the package `com.sun.services.cyberflex`, to the properties:

```
% smartcard -c admin -t service \
 -j com.sun.services.cyberflex.CyberFlexCardServiceFactory -x add
```

EXAMPLE 4 Adding a Reader

Enter the following command to add a SCM reader, available in the package `com.sun.services.scm`, to the properties on device `/dev/cua/a` and assign it a name of `SCM`:

```
% smartcard -c admin -t terminal \
-j com.sun.opencard.terminal.scm.SCMStc.SCMstcCardTerminalFactory \
-x add -d /dev/cua/a -r SCM -n SunSCRI
```

EXAMPLE 5 Deleting a Reader

Enter the following command to delete the SCM reader, added in the previous example, from the properties:

```
% smartcard -c admin -t terminal -r SCM -x delete
```

EXAMPLE 6 Changing the Debug Level

Enter the following command to change the debug level for all of the com.sun package to 9:

```
% smartcard -c admin -t debug -j com.sun -l 9  -x modify
```

EXAMPLE 7 Setting the Default Card for an Application

Enter one of the following commands to set the default card for an application (`dtlogin`) to be CyberFlex.

If the property default card does not exist, enter the following command:

```
 % smartcard -c admin -a dtlogin -x add defaultcard=CyberFlex
```

If the property default card exists, enter the following command:

```
% smartcard -c admin -a dtlogin -x modify defaultcard=CyberFlex
```

EXAMPLE 8 Exporting Keys for a User into a File

Enter the following command to export the challenge-response keys for a user into a file:

```
% smartcard -c admin -k challenge_response -E -o /tmp/mykeys
```

EXAMPLE 9 Importing Keys from a File

Enter the following command to import the challenge-response keys for a user from a file:

```
% smartcard -c admin -k challenge_response -I -i /tmp/mykeys
```

EXAMPLE 10 Downloading an Applet into a Java Card

Enter the following command to download an applet into a Java card or to configure a PayFlex (non-Java) card inserted into a SCM reader for the capx file supplied in the `/usr/share/lib/smartcard` directory:

```
% smartcard -c load -r SCM \
 -i /usr/share/lib/smartcard/SolarisAuthApplet.capx
```

EXAMPLE 11 Downloading an Applet Binary

Enter the following command to download an applet binary from some place other that the capx file supplied with Solaris8 into an IButton (the AID and input file are mandatory, the remaining parameters are optional):

```
% smartcard -c load -A A000000062030400  -i newapplet.jib
```

EXAMPLE 12 Downloading an Applet on a CyberFlex Access Card

On a CyberFlex Access Card, enter the following command to download an applet newapplet.bin at fileID 2222, instanceID 3333 using the specified verifyKey and a heap size of 2000 bytes:

```
% smartcard -c load -A newAID -i newapplet.bin \
fileID=2222 instanceID=3333 verifyKey=newKey \
MAC=newMAC heapsize=2000
```

EXAMPLE 13 Configuring a PayFlex Card

Enter the following command to configure a PayFlex (non-Java) card with specific AID, transport key, and initial pin:

```
% smartcard -c load aid-A00000006203400 \
pin=242424246A617661 transportKey=4746584932567840
```

EXAMPLE 14 Unloading an Applet from a Card

Enter the following command to unload an applet from iButton:

```
% smartcard -c load -u
```

EXAMPLE 15 Displaying Usage of `smartcard -c load`

Enter the following command to display the usage of the `smartcard -c load` command:

```
% smartcard -c load
```

EXAMPLE 16 Displaying All Configurable Parameters for an Applet

Enter the following command to display all the configurable parameters for an applet with aid 123456 residing on a card inserted into an SM reader:

```
% smartcard -c init -r SM -A 123456 -L
```

EXAMPLE 17 Changing the Pin

Enter the following command to change the pin for the SolarisAuthApplet residing on a card or to change the pin for a PayFlex (non-Java) card inserted into an SM reader:

```
% smartcard -c init -A A000000062030400 -P oldpin pin=newpin
```

EXAMPLE 18 Displaying All Configurable Parameters for the SolarisAuthApplet.

Enter the following command to display all the configurable parameters for the SolarisAuthApplet residing on a card inserted into an SM reader:

```
% smartcard -c init -A A000000062030400 -L
```

EXAMPLE 19 Setting a Property to a Value on a Smartcard

Enter the following command to set properties called "user" to the value "james" and “application” to the value “login” on a card inserted into an SM reader that has a pin "testpin”:

```
% smartcard -c init -A A000000062030400 -r CyberFlex -P testpin \
application=login user=james
```

EXAMPLE 20 Converting an Applet for the CyberFlex Card into `capx` Format.

Enter the following command to convert an applet for the CyberFlex card into the `capx` format required for downloading the applet into the card:

```
% smartcard -c bin2capx \
-i /usr/share/lib/smartcard/SolarisAuthApplet.bin \
-T CyberFlex -o /home/CorporateCard.capx -v memory=128 heapsize=12
```

EXAMPLE 21 Converting an Applet for the IButton Card into `capx` Format

Enter the following command to convert an applet for the IButton card into the `capx` format required for downloading the applet into the button:

```
% smartcard -c bin2capx -i /usr/share/lib/smartcard/SolarisAuthApplet.jib \
-T IButton -o /home/CorporateCard.capx -v
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWocf

SEE ALSO

`ocfserv`(1M), `attributes`(5), `smartcard`(5)

NOTES

The command line options contain only alphanumeric input.

NAME

smattrpop – populate security attribute databases in a name service

SYNOPSIS

smattrpop [-c] [-f] [-m] [-p *policy*] [-r] -s *scope* -t *scope* [-v] *database*

DESCRIPTION

The smattrpop command updates the auth-attr(4), exec_attr(4), prof_attr(4), and user_attr(4) role-based access control databases in a target NIS, NIS+, LDAP, or local /etc files name service from the corresponding databases in a source name service or files.

This command processes the table entries from the source database and merges each source entry field into the same field in the corresponding table entry in the target database. If a source entry does not exist in the target database, the entry is created. If the source entry exists in the target database, the fields are merged or replaced according to the command options.

Any errors encountered while updating the target entry are reported to stdout, and the command continues with the next source database entry.

OPTIONS

The following options are supported:

-c
: Performs cross-table checking. If you specify this option and a check error occurs, a message identifying the check error is written to stdout.

  The target entry values are checked against entries in related databases:

  - auths values — Each value must exist as the name of an authorization in the auth-attr(4) database.
  - profiles values — Each value must exist as a name of a profile in the prof_attr(4) database.
  - roles values — Each value must exist as the name of a role identity in the user_attr(4) database.
  - For each exec_attr(4) entry in the source database, the name must exist as the name of a profile in the prof_attr(4) database.

-f
: Specifies that the value in each field in the source entry replaces the value in the corresponding field in the target entry, if the source entry field has a non-empty value.

-m
: For the auths, profiles, and roles attributes, specifies that the values in each field in the source entry are merged with the values in the corresponding target entry field. If a source value does not exist in the target field, the value is appended to the set of target values. If the target field is empty, the source values replace the target field. The attribute values that merge depend on the database being updated:

  - prof_attr(4) — the auths and profiles attribute values are merged.

- `user_attr`(4) — the `auths`, `profiles`, and `roles` attribute values are merged.
- `exec_attr`(4) — the `uid`, `gid`, `euid`, and `egid` values are merged.

`-p` *policy*
: Specifies the value of the policy field in the `exec_attr`(4) database. Valid values are `suser` (standard Solaris superuser) and `tsol` (Trusted Solaris). If you specify this option, only the entries in the source `exec_attr` database with the specified policy are processed. If you omit this option, all entries in the source `exec_attr` database are processed.

`-r`
: Specifies that role identities in the `user_attr`(4) database in the source name service are processed. If you omit this option, only the normal user entries in the `user_attr` source database are processed.

`-s` *scope*
: Specifies the source name service or local file directory for database updates, using the following syntax:

  type`:/`*server*`/`*domain*

  where *type* indicates the type of name service. Valid values for *type* are:

  - `file` — local files
  - `nis` — NIS name service
  - `nisplus` — NIS+ name service
  - `ldap` — LDAP name service

  server indicates the local host name of the Solaris system on which the `smattrpop` command is executed, and on which both the source and target databases exist.

  domain specifies the management domain name for the name service.

  You can use two special cases of *scope* values:

  - To indicate the databases in the `/etc/security` local system directory, use the scope `file:`/*server*, where *server* is the name of the local system.
  - To load from databases in an arbitrary directory on the Solaris server, use the scope `file:`/*server*/*pathname*, where where *server* is the name of the local system and *pathname* is the fully-qualified directory path name to the database files.

`-t` *scope*
: Specifies the target name service or local file directory for database updates, using the following syntax:

  type`:/`*server*`/`*domain*

where *type* indicates the type of name service. Valid values for *type* are:

- `file` — local files
- `nis` — NIS name service
- `nisplus` — NIS+ name service
- `ldap` — LDAP name service

server indicates the local host name of the Solaris system on which the `smattrpop` command is executed, and on which both the source and target databases exist.

domain specifies the management domain name for the name service.

You can use two special cases of *scope* values:

- To indicate the databases in the `/etc/security` local system directory, use the scope `file:`*/server*, where *server* is the name of the local system.
- To update to databases in an arbitrary directory on the Solaris server, use the scope `file:`*/server/pathname*, where where *server* is the name of the local system and *pathname* is the fully-qualified directory path name to the database files.

`-v` Specifies that verbose messages are written. A message is written to `stdout` for each entry processed.

OPERANDS

The following operands are supported:

database Populates one or all databases. You can specify either the name of the database you want to process (for example, `auth_attr`), or `all` to process all databases. If you specify `all`, the databases are processed in the following order:

1. `auth_attr`(4)
2. `prof_attr`(4)
3. `exec_attr`(4)
4. `user_attr`(4)

EXAMPLES

EXAMPLE 1 Populating all tables in the NIS name service

The following example merges the values from all four attribute databases in the `/etc/security` directory of the local system into the corresponding tables in the NIS domain, `east.sun.com`. The command is executed on the master server, `hoosier`, for the NIS domain and the source files are in the `/etc` and `/etc/security` directories on the NIS master server. No cross-table checking is performed. A summary message indicating the number of entries processed and updated for each table is written to `stdout`.

EXAMPLE 1 Populating all tables in the NIS name service *(Continued)*

```
/usr/sadm/bin/smattrpop -s file:/hoosier \
         -t nis:/hoosier/east.sun.com all
```

EXAMPLE 2 Updating the authorization table in the NIS+ name service

This example merges new authorization data from a local system file in the `auth_attr` text format into the existing `auth_attr` database in the NIS+ domain, `east.sun.com`. The command is executed on the NIS+ master server, `foobar`. Values from the source `auth_attr` file replace the corresponding field values in the `NIS+` tables for each entry. A message is written to `stdout` for each entry processed. Database cross-checking is performed and any check error is written to `stdout`. A summary message indicating the number of entries processed and updated for the `auth_attr` database is written to `stdout`.

```
/usr/sadm/bin/smattrpop -c -f -v -s file:/foobar/var/temp \
         -t nisplus:/foobar/East.Sun.COM auth_attr
```

ENVIRONMENT VARIABLES

See `environ`(5) for a description of the `JAVA_HOME` environment variable, which affects the execution of the `smattrpop` command. If this environment variable is not specified, the `/usr/java` location is used. See `smc`(1M).

EXIT STATUS

Any errors encountered while updating the target entry are reported to `stdout`. The following exit values are returned:

`0` The specified tables were updated. Individual entries may have encountered checking errors.

`1` A syntax error occurred in the command line.

`2` A fatal error occurred and the tables were not completely processed. Some entries may have been updated before the failure.

FILES

`/etc/security/auth_attr`	Authorization description database. See `auth-attr`(4).
`/etc/security/exec_attr`	Execution profiles database. See `exec_attr`(4).
`/etc/security/prof_attr`	Profile description database. See `prof_attr`(4).
`/etc/user_attr`	Extended user attribute database. See `user_attr`(4).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmga

SEE ALSO smc(1M), smexec(1M), smprofile(1M), auth_attr(4), exec_attr(4), prof_attr(4), user_attr(4), attributes(5), environ(5)

NAME

smc – start the Solaris Management Console

SYNOPSIS

smc [*subcommand*] [*args*]

smc [*subcommand*] [*args*] -T *tool_name* [- – *tool_args*]

DESCRIPTION

The smc command starts the Solaris Management Console. The Solaris Management Console is a graphical user interface that provides access to Solaris system administration tools. It relies on Solaris Management Console servers running on one or more computers to perform modifications and report data. Each of these servers is a repository for code which the console can retrieve after the user of the console has authenticated himself or herself to the server.

The console can also retrieve toolboxes from the server. These toolboxes are descriptions of organized collections of tools available on that and possibly other servers. Once one of these toolboxes is loaded, the console will display it and the tools referenced in it.

The console can also run in a terminal (non-graphically), for use over remote connections or non-interactively from a script.

For information on the use of the graphical console, and for more detailed explanations of authentication, tools, and toolboxes, please refer to the Solaris Management Console online help available under the "Help" menu in the Solaris Management Console. To enable an NIS/NIS+ map to be managed from the Solaris Management Console, you must use the smc edit command to create a new toolbox for that map and enter the information about your NIS/NIS+ server where necessary. For instructions on creating a new toolbox, in the Solaris Management Console Help menu, select "Contents," then "About the Solaris Management Console Editor," then "To Create a Toolbox."

subcommands

smc *subcommand*s are:

`open` The default subcommand for the Solaris Management Console is `open`. This will launch the console and allow you to run tools from the toolboxes you load. It does not need to be specified explicitly on the command line.

`edit` The `edit` subcommand will also launch the console, like the `open` subcommand. However, after loading a toolbox, you will not be able to run the referenced tools. Instead, you will be able to edit that toolbox, that is, add, remove, or modify any tools or folders in that toolbox.

OPTIONS

The following options are supported. These letter options can also be specified by their equivalent option words preceded by a double dash. For example, you can use either `-D` or `- –domain` with the *domain* argument.

If *tool_args* are specified, they must be preceded by the `- –` option and separated from the double dashes by a space.

- –auth-data *file*
  Specifies a file which the console can read to collect authentication data. When running the Solaris Management Console non-interactively, the console will still need to authenticate itself with the server to retrieve tools. This data can either be passed on the command line using the `-u`, `-p`, `-r`, and `-l` options (which is insecure, because any user can see this data), or it can be placed in a file for the console to read. For security reasons, this file should be readable only by the user running the console, although the console does not enforce this restriction.

  The format of *file* is:

  `hostname=`*host name*
  `username=`*user name*
  `password=`*password for user name*
  `rolename=`*role name*
  `rolepassword=`*password for role name*

  Only one set of `hostname-username-password-rolename-rolepassword` may be specified in any one file. If the rolename is not specified, no role will be assumed.

-B | - –toolbox *toolbox*
  Loads the specified toolbox. *toolbox* can be either a fully-qualified URL or a filename. If you specify an HTTP URL as, for example,

  `http://`*host_name*`:`*port*`/. . .`

  it must point to a *host_name* and *port* on which an Solaris Management Console server is running. If you omit *port*, the default port, `898`, is used. This option overrides the `-H` option.

-D | - –domain *domain*
  Specifies the default domain that you want to manage. The syntax of *domain* is *type*:/*host_name*/*domain_name*, where *type* is `nis`, `nisplus`, `dns`, `ldap`, or `file`; *host_name* is the name of the machine that serves the domain; and *domain_name* is the name of the domain you want to manage. (*Note:* Do *not* use `nis+` for `nisplus`.) This option applies only to a single tool run in the terminal console.

  If you do not specify this option, the Solaris Management Console assumes the `file` default domain on whatever server you choose to manage, meaning that changes are local to the server. Toolboxes can change the domain on a tool-by-tool basis; this option specifies the domain for all other tools.

-h | - –help
  Prints a usage statement about the `smc` command and its subcommands to the terminal window. To print a usage statement for one of the subcommands, enter `-h` after the subcommand.

-H | - –hostname *host_name:port*
  Specifies the *host_name* and *port* to which you want to connect. If you do not specify a *port*, the system connects to the default port, `898`. If you do not specify *host_name:port*, the Solaris Management Console connects to the local host on port

898. You may still have to choose a toolbox to load into the console. To override this behavior, use the -B option (see above), or set your console preferences to load a "home toolbox" by default.

-J*java_option*
Specifies an option that can be passed directly to the Java runtime (see java(1). Do not enter a space between -J and the argument. This option is most useful for developers.

-l | - –rolepassword *role_password*
Specifies the password for the *role_name*. If you specify a *role_name* but do not specify a *role_password*, the system prompts you to supply a *role_password*. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

-p | - –password *password*
Specifies the password for the *user_name*. If you do not specify a password, the system prompts you for one. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

-r | - –rolename *role_name*
Specifies a role name for authentication. If you are running the Solaris Management Console in a terminal and you do not specify this option, no role is assumed. The GUI console may prompt you for a role name, although you may not need to assume a role.

-s | - –silent
Disables informational messages printed to the terminal.

-t
Runs the Solaris Management Console in terminal mode. If this option is not given, the Solaris Management Console will automatically run in terminal mode if it cannot find a graphical display.

- –trust
Trusts all downloaded code implicitly. Use this option when running the terminal console non-interactively and you cannot let the console wait for user input.

-T | - –tool *tool_name*
Runs the tool with the Java class name that corresponds to *tool_name*. If you do not specify this option and the Solaris Management Console is running in terminal mode, the system prompts you. If the Solaris Management Console is running in graphical mode, the system either loads a toolbox or prompts you for one (see options -H and -B).

-u | - –username *user_name*
Specifies the user name for authentication. If you do not specify this option, the user identity running the console process is assumed.

-v | - –version
Prints the version of the Solaris Management Console to the terminal. In the graphical console, this information can be found in the About box, available from the Help menu.

`-y | - –yes`
Answers `yes` to all yes/no questions. Use this option when running the terminal console non-interactively and you cannot let the console wait for user input.

EXAMPLES

EXAMPLE 1 Printing a Usage Statement

The following prints a usage statement about the `smc` command to the terminal window:

```
smc --help
```

EXAMPLE 2 Passing an Option to Java

The following passes an option through to the Java VM, which sets the `com.example.boolean` system property to `true`. This system property is only an example; the Solaris Management Console does not use it.

```
smc -J-Dcom.example.boolean=true
```

ENVIRONMENT VARIABLES

See `environ`(5) for a description of the following environment variable that affects the execution of the `smc` command:

`JAVA_HOME` If you do not specify this environment variable, your `PATH` is searched for a suitable java. Otherwise, the `/usr/j2se` location is used.

EXIT STATUS

The following exit values are returned. Other error codes may be returned if you specify a tool (using `-T` *tool_name*) that has its own error codes. See the documentation for the appropriate tool.

`0` Successful completion.

`1` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmcc

SEE ALSO

`auths`(1), `java`(1), `profiles`(1), `roles`(1), `smcconf`(1M), `attributes`(5), `environ`(5), `X`(7)

NAME

smccompile – build class list and compile Solaris Management Console service beans for remote use

SYNOPSIS

/usr/sadm/bin/smccompile `-c` *beanname*

/usr/sadm/bin/smccompile `-j tool | service [-n` *altjarname*`]` *jarfile*

/usr/sadm/bin/smccompile `-j library [-n` *altjarname*`] ALLTOOL | ALLSERVICE | ALL |` *attachedBeanname jarfile*

DESCRIPTION

The `smccompile` command is used by developers of tools, services, and libraries for the Solaris Management Console. For information regarding the Solaris Management Console, see `smc`(1M).

`smccompile` compiles service class files given by the bean name for use with the Solaris Management Console. This step builds the extra proxy and stub classes for services to be used with Solaris Management Console tools. Solaris Management Console requires running `smccompile -c` before creating service jar files, and `smccompile -j` after creating tool, service, and library jars.

`smccompile`, in conjunction with `smcregister`(1M), is intended to replace the `smcconf` command as the preferred interface for managing the Solaris Management Console repository as well as toolboxes from within scripts, due to significant performance enhancements over `smcconf`.

OPTIONS

The following options are supported:

ALL	Specify that the library being registered to or unregistered from the repository is for use by all tools and services.
ALLSERVICE	Specify that the library being registered to or unregistered from the repository is for use by all services.
ALLTOOL	Specify that the library being registered to or unregistered from the repository is for use by all tools.
attachedBeaname	Specify the name of a registered jar to which the library jarfile should be attached to (or detached from). This is typically the same as *altjarname* (if provided) or *jarfile* used to register the jar to which this library is being attached or detached. An attached library means the library is only available for use by the tool or service to which it is being attached.
beanname	The full package path of the bean name to be compiled. An example bean name is: `com.mycompany.myproduct.MyService`.
`-c`	Compile and build service class files for the specified bean name. This step builds the extra proxy and stub classes for services to be used with Solaris Management Console tools. You must run `smccompile` with this option before creating service type jar files.

`-j`	Build a list of classes in text format, suitable as input to `smcregister` for registration with the Solaris Management Console repository. The output is written to standard out and should be redirected to a file. You must run `smccompile` with this option after creating any tool, service, or library jar.
jarfile	Specify the full path to the jar file to be registered. The name must be in the form *beanname*`.jar`, where *beanname* is the package path to the bean. If it is not, an alternate name must be given in that form using the `-n` option.
`-n` *altjarname*	Rename the jarfile in the repository to *altjarname*. Typically. this is the full bean name. For example, if the jarfile was `MyTool.jar`, then *altjarname* might be `com.mycompany.myproduct.MyTool.jar`. It is recommended that an *altjarname* containing the full package path be used. You must use this same name when registering the jar with `smcregister`.

EXAMPLES

EXAMPLE 1 Compiling a Service

The following command takes a Solaris Management Console service and builds its proxy and stub classes to make the service usable by Solaris Management Console tools:

```
/usr/sadm/bin/smccompile -c com.mycompany.myproject.MyServiceImpl
```

EXAMPLE 2 Building a Class List for a Service

The following command builds the class list file (`classlist.txt`) for a service suitable for use with the `smcregister`(1M) command:

```
/usr/sadm/bin/smccompile -j service \
     -n com.mycompany.myproject.MyServiceImpl.jar \
     ${HOME}/workarea/MyServiceImpl.jar > classlist.txt
```

The following command does the same thing without specifying an alternate name:

```
/usr/sadm/bin/smccompile -j service \
     ${HOME}/workarea/com.mycompany.myproject.MyServiceImpl.jar > classlist.txt
```

EXAMPLE 3 Building a Class List for a Tool

The following command builds the class list file (`classlist.txt`) for a tool suitable for use with the `smcregister`(1M) command:

```
/usr/sadm/bin/smccompile -j tool \
     -n com.mycompany.myproject.MyTool.jar \
     ${HOME}/workarea/MyTool.jar > classlist.txt
```

The following command does the same thing without specifying an alternate name:

EXAMPLE 3 Building a Class List for a Tool *(Continued)*

```
/usr/sadm/bin/smccompile -j tool \
     ${HOME}/workarea/com.mycompany.myproject.MyTool.jar > classlist.txt
```

EXAMPLE 4 Building a Class List for a Library Attached to All Tools

The following command builds the class list file (`classlist.txt`) for a library suitable for use with the `smcregister`(1M) command, and is attached to all tools:

```
/usr/sadm/bin/smccompile -j library \
     -n com.mycompany.myproject.MyLibrary.jar \
     ALLTOOL ${HOME}/workarea/MyLibrary.jar > classlist.txt
```

The following command does the same thing without specifying an alternate name:

```
/usr/sadm/bin/smccompile -j library \
    ALLTOOL \
    ${HOME}/workarea/com.mycompany.myproject.MyLibrary.jar > classlist.txt
```

EXAMPLE 5 Building a Class List for a Library Attached to a Specific Tool

The following command builds the class list file (`classlist.txt`) for a library suitable for use with the `smcregister`(1M) command, and is attached to a specific tool:

```
/usr/sadm/bin/smccompile -j library \
     -n com.mycompany.myproject.MyLibrary.jar \
     com.mycompany.myproject.MyTool.jar \
     ${HOME}/workarea/MyLibrary.jar > classlist.txt
```

The following command does the same thing without specifying an alternate name:

```
/usr/sadm/bin/smccompile -j library \
     com.mycompany.myproject.MyTool.jar \
     ${HOME}/workarea/com.mycompany.myproject.MyLibrary.jar > classlist.txt
```

ENVIRONMENT VARIABLES

See `environ`(5) for descriptions of the following environment variables that affect the execution of `smccompile`:

JAVA_HOME — If you do not specify this environment variable, your PATH is searched for a suitable java. Otherwise, the `/usr/j2se` location is used.

EXIT STATUS

The following exit values are returned:

`0` — Successful completion.

`1` — An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWMc

SEE ALSO `smc`(1M), `smcconf`(1M), `smcregister`(1M), `attributes`(5), `environ`(5)

NOTES All standard shell quoting rules apply.

NAME

smcconf – configure the Solaris Management Console

SYNOPSIS

/usr/sadm/bin/smcconf [-h] [-v] toolbox [*action*] [*target*] [*parameters*] [*options*]

/usr/sadm/bin/smcconf [-h] [-v] repository [*action*] [*target*] [*parameters*] [*options*]

DESCRIPTION

The smcconf command configures the Solaris Management Console. See smc(1M). This command enables you to add to, remove from, and list the contents of the toolboxes and bean repository.

Using smcconf to edit toolboxes is not as feature-rich as using the graphical editor in Solaris Management Console. The command line interface is intended for use in packaging scripts that do not require user interaction. To edit all the properties of a toolbox or to modify the hierarchy of folders in a toolbox, you must use the specialized graphical editor, that is, smc edit. See smc(1M).

smcregister is intended to replace the smcconf command as the preferred interface for managing the Solaris Management Console repository as well as toolboxes from within scripts, due to significant performance enhancements over smcconf. See smcregister(1M), smccompile(1M), and the Solaris Management Console SDK Guide at /usr/sadm/lib/smc/docs/sdkguide/index.html for details.

OPTIONS

The following options are supported:

-h Prints out a usage summary for the command.

-v Verbose option. Displays the debugging output at any time.

toolbox configuration

action Legal values are:

add
Adds a *target* to the toolbox. Specify the path to the toolbox using the -B *toolboxpath* option and, optionally, provide locale information with the -L *locale* option.

remove
Removes a *target* from the toolbox. Specify the path to the toolbox using the -B *toolboxpath* option and, as an alternative, provide locale information with the -L *locale* option.

create
Creates a new toolbox with no tools in it. The only *target* recognized is toolbox.

list

Lists the contents of the toolbox. No *target* is recognized. If you specify a *parameter*, it is taken as the path to a toolbox and the contents of that toolbox are listed. If you do not specify a *parameter*, the contents of the default toolbox are listed.

target Legal values are:

`tool`
If the *action* is specified as `add`, this target adds a native Solaris Management Console tool from the toolbox. The required *parameter* is the full Java classname of the tool you are adding. If you specify a folder name with the `-F` option, the tool is placed inside that folder (the folder will not be created if it does not already exist). Otherwise, the tool is appended to the end of the toolbox and not placed inside any folder.

If the *action* is specified as `remove`, this target removes a native Solaris Management Console tool from the toolbox. The required *parameter* is the full Java classname of the tool you want to remove. If you specify a folder name with the `-F` option, any tool with the given name in that folder will be removed. If no folder name is specified, all tools with the given name in the toolbox are removed.

For the tool to appear in the console, the tool must also be registered in the repository. See the `repository configuration` section below for more information. If a tool is referenced in a toolbox but is not registered, it will not appear in the console when the toolbox is loaded.

Removing a tool from a toolbox does not remove the tool from the server repository.

`tbxURL`
If the *action* is specified as `add` or `remove`, this target adds to or removes from the toolbox a link to another toolbox. The required *parameter* is the URL to the other toolbox.

The properties of addition and removal are the same as for the `tool` *target*.

`toolbox`
If the *action* is specified as `create`, this target creates a skeleton toolbox with no tools. The required *parameter*s are: the toolbox name, description, and small and large icon paths. These must be followed by the `-B` *toolboxpath* and `-D` *scope* options.

`legacy`
If the *action* is specified as `add` or `remove`, this target adds or removes legacy applications (command-line, X-windows, and web-based) to or from the toolbox. The `-N`, `-T`, `-E`, and `-B` options are required. The `-A` option is optional. Placement in the toolbox with the `-F` option follows the same rules as for the `tool` and `tbxURL` targets. See `NOTES` for more information about legacy applications.

`folder`
If the *action* is specified as `add`, this target adds a folder to the toolbox. The required *parameter*s are: the folder name, description, and small and large icon paths.

If the *action* is specified as `remove`, this target removes a folder from the toolbox. If the folder to be removed is itself inside a folder, the containing folder must be specified with the `-F` option.

parameters Specifies values that might be required, depending on the combination of *action* and *target*.

options Supported options for various *action* and *target* combinations for the `toolbox` configuration are:

`-A` *parameters*
Specifies the parameters to pass to the legacy application. This option is available only for the `legacy` target.

`-B` *toolboxpath*
Specifies the path of the toolbox that is being modified. If this option is not given, the modifications will be performed on the default toolbox, "`This Computer`".

`-D` *scope*
Specifies the scope (domain) in which the tool should be run. The legal values for *scope* are `file`, `nis`, `nisplus`, `dns`, and `ldap`. This can also be specified for a folder or a toolbox.

In the former case, all tools in that folder and its subfolders are run in that scope; in the latter, all tools in the toolbox are run in that scope.

`-E` *appPath*
Specifies the absolute executable path of the legacy application. This option is available only for the `legacy` target.

`-F` *folder*
Specifies the full path of the container folder. If this option is not given, the default folder is the 'root' folder of the toolbox.

`-H` [*host_name*][`:`*port*]
Specifies the host and port from which a tool should be loaded. If *host_name* is not given, the default host is used. The default host is `localhost`, if the toolbox is loaded from the local file system, or the host from which the toolbox is loaded if loaded from a remote Solaris Management Console server. If `:port` is not given, the default port will be used. If this option is not given at all, both the default host and the default port are used.

`-L` *locale*
Specifies the locale of the toolbox that is being modified. The default is the `C locale`.

`-N` *appName*
Specifies the name of the legacy application being registered. This is the name that appears in the console. This option is available only for the `legacy` target.

`-P` *key*:*value*
Specifies the key/value pairs that define parameters to a tool. Multiple key/value pairs can be specified at a time.

`-T` *appType*
Specifies the legacy application type. Legal values are `CLI`, `XAPP`, or `HTML`. This option is available only for the `legacy` target.

repository configuration

The Solaris Management Console repository stores information about the registered tools and services, as well as libraries (for instance, resource jars) and properties attached to tools or services.

action Legal values are:

`add`
Adds information to the repository. If the `-f` option is given to `add`, the information overwrites any information of the same name already in the repository. If the `-f` option is not given, an error might be returned if the information is already in the repository.

`remove`
Removes information from the repository.

`list`

Lists the contents of the repository:

- All registered tools
- All registered services
- All libraries attached to all tools
- All libraries attached to all services
- All libraries attached to all tools and services

target Legal values are:

`bean`
If the *action* is specified as `add`, this target will add a tool or service bean (which kind is determined by the contents of the bean) to the repository. The required *parameter* is the path to the jar file that contains the bean to be added.

If the *action* is specified as `remove`, this target will remove a tool or service bean from the repository. The required *parameter* is the full Java classname of the desired bean.

`library`
If the *action* is specified as `add`, this target adds a "library" jar file to a tool or service bean. The two required *parameter*s are the full Java classname of the desired bean and the path to the jar file to be attached. The bean name can also be one of the "pseudo-beans," `ALL`, `ALLTOOL`, or `ALLSERVICE`, in which case the library is attached, respectively, to all beans, all tools, or all services in the repository.

If the *action* is specified as `remove`, this target detaches a "library" jar file from a tool or service bean. The two required *parameter*s are the full Java classname of the desired bean and the name of the jar file that is attached. As with the `add` action, the three "pseudo-beans" `ALL`, `ALLTOOL`, or `ALLSERVICE` can be used.

`property`
If the *action* is specified as `add`, this target defines a property on a tool or service. One or more key/value pairs must be specified in the form,

`-P` *key*=*value*Following this property list is a "pseudo-bean name," *pseudoBeanName*, as defined for the `library` target, on which the properties are defined. Optionally, a library name can follow the "pseudo-bean" name, in which case the properties are defined on the library that is attached to the named bean.

If the *action* is specified as `remove`, this target undefines a property on a tool or service. The key/value pairs, "pseudo-bean" name, and optional library are specified for the `add` action.

EXAMPLES

EXAMPLE 1 Adding Legacy Applications to a Toolbox

The following command adds to the default toolbox the command line interface (CLI) application, `/usr/bin/ls`, with arguments `-al -R`, giving it the name, `Directory Listing`:

```
/usr/sadm/bin/smcconf toolbox add legacy -N "Directory Listing" \
     -T CLI -E /usr/bin/ls -A "-al -R"
```

EXAMPLE 2 Adding a Folder to a Toolbox

The following command adds to the standard `Management Tools` toolbox a folder with the name, `New Folder`, the description, `This is a new folder`, and the small and large icons, `folder_s.gif` and `folder_l.gif`:

EXAMPLE 2 Adding a Folder to a Toolbox *(Continued)*

```
/usr/sadm/bin/smcconf toolbox add folder "New Folder" \
     "This is a new folder" folder_s.gif folder _l.gif \
     -B /var/sadm/smc/toolboxes/smc/smc.tbx
```

EXAMPLE 3 Adding a Native Solaris Management Console Tool to a Toolbox

The following command adds a native Solaris Management Console tool to the default toolbox. The Java classname of the tool is `HelloWorld.client.HelloTool` (the name, description, and icons visible in the console are provided by the tool itself). When loaded, it is run in the NIS domain, `syrinx`, which is hosted by the machine, `temple`, and is retrieved from port `2112` on the machine from which the toolbox was loaded:

```
/usr/sadm/bin/smcconf toolbox add tool HelloWorld.client.HelloTool \
     -D nis:/temple/syrinx -H :2112
```

EXAMPLE 4 Adding an Solaris Management Console Tool to the Repository

The following command adds the Java bean found in `HelloWorld.jar` to the repository. The jar file contains information that the bean is a tool:

```
/usr/sadm/bin/smcconf repository add bean HelloWorld.jar
```

EXAMPLE 5 Removing an Solaris Management Console Service from the repository

The following command removes a Java bean from the repository. Although the name of the bean implies that it is a service, that is merely a convention; the repository knows whether a particular registered bean is a tool or a service:

```
/usr/sadm/bin/smcconf repository remove bean \
     HelloWorld.server.HelloService
```

EXAMPLE 6 Attaching a Library to a Tool

The following command adds the library jar file, `HelloWorld_fr.jar` (probably a French localized version of the `HelloTool`'s resources) to the bean, `HelloWorld.client.HelloTool`:

```
/usr/sadm/bin/smcconf repository add library \
     HelloWorld.client.HelloTool HelloWorld_fr.jar
```

EXAMPLE 7 Attaching a Library to all Tools

The following command adds the library jar file, `widgets.jar`, to all tools in the repository. The library probably contains a widget set that might be useful to any registered tools:

```
/usr/sadm/bin/smcconf repository add library ALLTOOL widgets.jar
```

ENVIRONMENT VARIABLES

See `environ`(5) for descriptions of the following environment variables that affect the execution of the `smcconf` command:

`JAVA_HOME` If you do not specify this environment variable, your PATH is searched for a suitable java. Otherwise, the `/usr/j2se` location is used.

`DISPLAY` If you do not set this environment variable, set it to null, or set it to an X(7) display to which you are not authorized to connect, the Solaris Management Console starts in terminal mode instead of graphical mode.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmc

SEE ALSO

`jar`(1), `java`(1), `javac`(1), `smc`(1M), `smccompile`(1M), `smcregister`(1M), `attributes`(5), `environ`(5)

NOTES

All standard shell quoting rules apply.

Legacy applications (X-windows, command-line, and web-based applications) are handled differently from “native” Solaris Management Console tools. Legacy tools are handled by an instantiation of a native Solaris Management Console tool, `LegacyAppLauncher`, which, through the toolbox, is given the necessary information to run the legacy application: path, options, and so forth. Thus, you do not register a legacy application into the repository as you would a native Solaris Management Console tool. Instead, legacy applications appear only in toolboxes.

NAME

smcregister – configure the Solaris Management Console

SYNOPSIS

/usr/sadm/bin/smcregister [-h] *tool* [-n *altjarname*] *jarfile classlistfile xmlfile*

/usr/sadm/bin/smcregister [-h] *service* [-n *altjarname*] *jarfile classlistfile xmlfile* [*native_lib_list*]

/usr/sadm/bin/smcregister [-h] *library* [-n *altjarname*] *jarfile classlistfile* | none ALLTOOL | ALLSERVICE | ALL | *Attachedbeanname*

/usr/sadm/bin/smcregister [-h]tool | service-u *jarfile*

/usr/sadm/bin/smcregister [-h] library -u *jarfile* ALL | ALLTOOL | ALLSERVICE | *Attachedbeanname*

/usr/sadm/bin/smcregister [-h] *toolbox* [-D] [*action*] [-f] [*target*] [*parameters*] [*options*]

/usr/sadm/bin/smcregister [-h] *property key value* ALL | ALLTOOL | ALLSERVICE | *Attachedbeanname*

/usr/sadm/bin/smcregister [-h] *property* -u *key* ALL | ALLTOOL | ALLSERVICE | *Attachedbeanname*

/usr/sadm/bin/smcregister [-h] *repository list*

/usr/sadm/bin/smcregister [-h] *scripts regscript unregscript*

DESCRIPTION

The smcregister command configures the Solaris Management Console. For information regarding the Solaris Management Console, see smc(1M). This command enables you to add to, remove from, and list the contents of toolboxes and the Solaris Management Console repository.

smcregister also allows you to register scripts to perform registrations and unregistrations. Typically, a package containing one or more tools or services posts tool and service registrations immediately after installation. On Solaris, this is by way of invocations of smcregister from within a package post-install script. Similarly, unregistrations would be posted from within a package pre-remove script. These are per-machine registrations - that is, registration requests must be posted on each machine on which the Solaris Management Console server will be running. However, due to the way that diskless clients are installed, registration requests cannot be made at install time. Therefore, packages should include and install registration and unregistration scripts, and then register these scripts during installation by way of the scripts subcommand. These scripts should contain tool, toolbox, service, library or property configurations in any of its forms as listed in this man page. While these scripts function very much like package post-install and pre-remove scripts, do not assume the normal package environment is available. However, PATH can assumed to be /usr/sbin:/usr/bin

Using `smcregister` to edit toolboxes is not as feature-rich as using the Solaris Management Console's graphical editor. The command line interface is intended for use in packaging scripts that do not require user interaction. To edit all the properties of a toolbox or to modify the hierarchy of folders in a toolbox, you must use the specialized graphical editor, that is, `smc edit`. See `smc`(1M).

`smcregister` is intended to replace the `smcconf` command as the preferred interface for managing the Solaris Management Console repository as well as toolboxes from within scripts, due to significant performance enhancements over `smcconf`.

OPTIONS

The following options are supported:

`-h` Prints out a usage summary for the command.

Scripts Configuration

regscript — The full path of a script containing registration commands. The script is executed upon the next restart of the Solaris Management Console server after the package containing the script is installed.

unregscript — The full path of a script containing unregistration commands. The script is executed upon the next restart of the Solaris Management Console server after the package containing the script is removed.

Toolbox Configuration

action — Legal values are:

add
: Adds a target to the toolbox. Specify the path to the toolbox using the `-B toolboxpath` option and, optionally, provide locale information with the `-L locale` option.

create
: Creates a new toolbox with no tools in it. The only target recognized is `toolbox`.

list
: Lists the contents of the toolbox. No target is recognized. If you specify a parameter, it is taken as the path to a toolbox and the contents of that toolbox are listed. If you do not specify a parameter, the contents of the default toolbox are listed.

remove
: Removes a target from the toolbox. Specify the path to the toolbox using the `-B toolboxpath` option and, optionally, provide locale information with the `-L locale` option.

`-D` — Defers execution of the `toolbox` command until the Solaris Management Console server is restarted. This is a convenient option for use in packaging scripts during install and un-install. Additionally, the command runs much faster than if run interactively (without `-D`).

target — Legal values are:

folder
: If the action is specified as `add`, this target adds a folder to the toolbox. There are four required parameters: the folder name, description, and small and large icon paths. If the action is specified as remove, this target removes a folder from the toolbox. If the folder to be removed is itself inside a folder, the containing folder must be specified with the `-F` option.

legacy
: If the action is specified as `add` or `remove`, this target adds or removes legacy applications (command line, `X-windows`, and web-based) to or from the toolbox. The `-N`, `-T`, `-E`, and `-B` options are required, and the `-A` option is optional. Placement in the toolbox with the `-F` option follows the same rules as for the `tool` and `tbxURL` targets.See `NOTES` for more information about legacy applications.

tbxURL
: If the action is specified as `add` or `remove`, this target adds to or removes from the toolbox a link to another toolbox. The required parameter is the URL to the other toolbox. The properties of addition and removal are the same as for the tool target.

tool
: If the action is specified as `add`, this target adds a native Solaris Management Console tool from the toolbox. The required parameter is the full Java classname of the tool you are adding. If you specify a folder name with the `-F` option, the tool is placed inside that folder (the folder will not be created if it does not already exist). Otherwise, the tool is appended to the end of the toolbox and not placed inside any folder. If the action is specified as remove, this target removes a native Solaris Management Console tool from the toolbox. The required parameter is the full Java classname of the tool you wish to remove. If you specify a folder name with the `-F` option, any tool with the given name in that folder will be removed. If no folder name is specified, all tools with the given name in the toolbox will be removed. For the tool to show up in the console, the tool must also be registered in the repository. See the repository configuration section below for more information. If a tool is referenced in a toolbox but is not registered, it will not appear in the console when the toolbox is loaded. Removing a tool from a toolbox does not remove the tool from the server repository.

toolbox
: If the action is specified as `create`, this target creates a skeleton toolbox with no tools. There are four required parameters: the toolbox name, description, and small and large icon paths. These must be followed by the `-B` *toolboxpath* and `-D` *scope* options.

parameters Specifies values that may be required depending on the combination of action and target.

options Supported options for various action and target combinations for the toolbox configuration are:

`-A`
Specifies the parameters to pass to the legacy application. This option is available only for the legacy target.

`-B`
Specifies the path of the toolbox that is being modified. If this option is not given, the modifications will be performed on the default toolbox, `This Computer`.

`-D`
Specifies the scope (domain) in which the tool should be run. The legal values for scope are `file`, `nis`, `nisplus`, `dns`, and `ldap`. This may also be specified for a folder or a toolbox. In the former case, all tools in that folder and its subfolders will be run in that scope; in the latter, all tools in the toolbox will be run in that scope.

`-E`
Specifies the absolute executable path of the legacy application. This option is available only for the legacy target.

`-f`
If the `-f` option is given to add, the information will overwrite any information of the same name already in the toolbox. If the `-f` option is not given, an error may be returned if the information is already in the toolbox.

`-F` *folder*
Specifies the full path of the container folder. If this option is not given, the default folder is the *root* folder of the toolbox.

`-H` [*host_name*][:*port*]
Specifies the host and port from which a tool should be loaded. If host_name is not given, the default host (localhost, if the toolbox is loaded from the local filesystem, or the host from which the toolbox is loaded if loaded from a remote Solaris Management Console server) will be used. If `:`*port* is not given, the default port will be used. If this option is not given at all, both the default host and the default port will be used.

`-L` *locale*
Specifies the locale of the toolbox which is being modified. The default is the `C` locale.

`-N` *appName*
Specifies the name of the legacy application being registered. This is the name that will appear in the console. This option is available only for the `legacy` target.

`-P` *key:value*
Specifies the key/value pairs that define parameters to a tool. Multiple key/value pairs can be specified at a time.

`-T` *appType*
Specifies the legacy application type. Legal values are `CLI`, `XAPP`, or `HTML`. This option is available only for the legacy target.

Tool, Service, and Library Configuration

See `NOTES` for more information about registration and unregistration of tools, services, and libraries.

ALL	Specify that the library being registered to or unregistered from the repository is for use by all tools and services.
ALLSERVICE	Specify that the library being registered to or unregistered from the repository is for use by all services.
ALLTOOL	Specify that the library being registered to or unregistered from the repository is for use by all tools.
attachedBeanname	The name of a registered jar to which the library jarfile should be attached to (or detached from). This is typically the same as *altjarname* (if provided) or *jarfile* used to register the jar to which this library is being attached or detached. An attached library means the library is only available for use by the tool or service to which it is being attached.
classlistfile	The classlist text file generated from the smccompile(1M) command. Library registration does not require that a classlist file be specified. Instead, you can substitute the keyword `none` in place of the classlist path argument to `smcregister`, in which case one will be generated automatically. Generating the classlist automatically during server startup will cause the next server restart to take longer, so it is strongly suggested that developers always provide a classlist file with their libraries. Auto-generation is more appropriately used to register 3rd-party library jars.
jarfile	The full path to the jar file to be registered/unregistered. The name must be in the form *beanname*`.jar`, where *beanname* is the package path to the bean. If it is not, an alternate name must be given in that form using the -n option.
`-n` *altjarname*	Rename the jarfile in the repository to *altjarname*. This would typically be the full bean name. For example, if the jarfile was `MyTool.jar`, then *altjarname* might be com.mycompany.myproduct.MyTool.jar. It is recommended that an *altjarname* containing the full package path be used.

native_lib_list — List of up to 4 native libraries that can be associated with a service bean.

`-u` — The operation will be to un-register the jar with the Solaris Management Console repository. The jarfile argument must be identical to the *altjarname* used to register the jar (if provided), or jarfile.

xmlfile — The `xml` descriptor file that describes this jarfile. Every tool or services must have one. See the `Solaris Management Console SDK Guide` located at `/usr/sadm/lib/smc/docs/sdkguide/index.html`.

Repository Configuration

The Solaris Management Console repository stores information about the registered tools and services, as well as libraries (for instance, resource jars) and properties attached to tools or services.

list — Lists the contents of the repository:

- All registered tools
- All registered services
- All libraries attached to all tools
- All libraries attached to all services
- All libraries attached to all tools and services

Property Configuration

See `NOTES` for more information about registration and unregistration of properties. If registering a property, this defines a property on a tool or service. Only one key value pair at a time can be registered.

beanname — The name of a registered jar on which the properties will be defined. Optionally, a library name may follow the bean name, in which case the properties are defined on the library that is attached to the named bean.

If unregistering a property, this undefines a property from a tool or service. Only one key value pair at a time can be registered. The key, beanname, and optional library are specified as for registering a property.

EXAMPLES

EXAMPLE 1 Adding Legacy Applications to a Toolbox

The following command adds to the default toolbox the Command Line Interface (CLI) application, `/usr/bin/ls` with arguments `-al -R`, giving it the name, Directory Listing:

```
/usr/sadm/bin/smcregister toolbox add legacy -N "Directory Listing" \
     -T CLI -E /usr/bin/ls -A "-al -R"
```

Use this variation to defer execution of this command until the Solaris Management Console server is restarted:

```
/usr/sadm/bin/smcregister toolbox -D add legacy -N "Directory Listing" \
     -T CLI -E /usr/bin/ls -A "-al -R"
```

EXAMPLE 2 Adding a Folder to a Toolbox

The following command adds to the standard `Management Tools` toolbox a folder with the name, New Folder, the description, This is a new folder, and the small and large icons, `folder_s.gif` and `folder_l.gif`:

```
/usr/sadm/bin/smcregister toolbox add folder "New Folder" \
     "This is a new folder" folder_s.gif folder _l.gif \
     -B /var/sadm/smc/toolboxes/smc/smc.tbx
```

EXAMPLE 3 Adding a Native Solaris Management Console Tool to a Toolbox

The following command adds a native Solaris Management Console tool to the default toolbox. The Java classname of the tool is `com.mycompany.myproject.client.MyTool` (the name, description, and icons visible in the console are provided by the tool itself). When loaded, it will be run in the NIS domain, syrinx, which is hosted by the machine, temple, and will be retrieved from port 2112 on the machine from which the toolbox was loaded.

```
/usr/sadm/bin/smcregister toolbox add tool \
     com.mycompany.myproject.client.MyTool \
     -D nis:/temple/syrinx -H :2112
```

EXAMPLE 4 Adding an Solaris Management Console Tool to the Repository

The following command adds the Java bean found in MyTool.jar to the repository. The xml file contains information about the tool. The classlist file would have been generated by `smccompile -j`:

```
/usr/sadm/bin/smcregister tool -n com.mycompany.myproject.client.MyTool.jar \
     ${HOME}/workarea/MyTool.jar \
     ${HOME}/workarea/MyTool_classlist.txt \
     ${HOME}/workarea/MyTool.xml
```

Use this variation to add an Solaris Management Console tool to the repository without specifying an alternate name:

```
/usr/sadm/bin/smcregister tool  \
     ${HOME}/workarea/com.mycompany.myproject.client.MyTool.jar \
     ${HOME}/workarea/MyTool_classlist.txt \
     ${HOME}/workarea/MyTool.xml
```

EXAMPLE 5 Adding an Solaris Management Console Service to the Repository

The following command adds the Java bean found in MyServiceImpl.jar to the repository. The xml file contains information about the service. The classlist file would have been generated by `smccompile -j`. The extra proxy and stub classes included in the jar would have been generated by `smccompile -c`:

```
/usr/sadm/bin/smcregister service \
     -n com.mycompany.myproject.server.MyServiceImpl.jar \
     ${HOME}/workarea/MyServiceImpl.jar \
```

EXAMPLE 5 Adding an Solaris Management Console Service to the Repository *(Continued)*

```
     ${HOME}/workarea/MyServiceImpl_classlist.txt \
     ${HOME}/workarea/MyServiceImpl.xml
```

Use this variation to add a Solaris Management Console service to the repository without specifying an alternate name:

```
/usr/sadm/bin/smcregister service  \
     ${HOME}/workarea/com.mycompany.myproject.server.MyServiceImpl.jar \
     ${HOME}/workarea/MyServiceImpl_classlist.txt \
     ${HOME}/workarea/MyServiceImpl.xml
```

EXAMPLE 6 Removing an Solaris Management Console Tool From the Repository

The following command removes a Java tool bean from the repository:

```
/usr/sadm/bin/smcregister tool \
     -u com.mycompany.myproject.client.MyTool.jar
```

EXAMPLE 7 Removing an Solaris Management Console Service From the Repository

The following command removes a Java service bean from the repository:

```
/usr/sadm/bin/smcregister service \
     -u com.mycompany.myproject.server.MyServiceImpl.jar
```

EXAMPLE 8 Attaching a Library to a Specific Tool

The following command adds the library jar file, MyTool_fr.jar (probably a French localized version of the MyTool's resources) to the bean, com.mycompany.myproject.client.MyTool:

```
/usr/sadm/bin/smcregister library  \
     -n MyTool_fr.jar \
     ${HOME}/workarea/MyTool_fr.jar \
     ${HOME}/workarea/MyTool_fr_classlist.txt \
     com.mycompany.myproject.client.MyTool
```

EXAMPLE 9 Attaching a Library to All Tools

The following command adds the library jar file, widgets.jar, to all tools in the repository. The library probably contains a widget set which might be useful to any registered tools. The classlist file would have been generated by `smccompile -j`.

```
/usr/sadm/bin/smcregister library \
     ${HOME}/workarea/lib/widgets.jar\
     ${HOME}/workarea/lib/widgets_classlist.txt \
     ALLTOOL
```

EXAMPLE 9 Attaching a Library to All Tools *(Continued)*

Alternatively, to add a 3rd-party library jar to all tools, replace the classlist file with `none`:

```
/usr/sadm/bin/smcregister library \
     /opt/lib/XYZwidgets.jar none ALLTOOL
```

EXAMPLE 10 Detaching a Library from All Tools

The following command removes the Java library bean from the repository:

```
/usr/sadm/bin/smcregister library -u MyTool_fr.jar ALLTOOL
```

EXAMPLE 11 Detaching a Library from a Specific Tool

The following command detaches the library jar file, MyTool_fr.jar (probably a French localized version of the MyTool's resources) from the bean `com.mycompany.myproject.client.MyTool`, and removes it from the repository:

```
/usr/sadm/bin/smcregister library -u MyTool_fr.jar \
     com.mycompany.myproject.client.MyTool
```

EXAMPLE 12 Registering Scripts

The following command registers the following scripts containing registration and unregistration commands. MyProduct_reg.sh will be executed upon the next server restart after the file is installed by the owning package. MyProduct_unreg.sh will be executed upon the next server restart after the file is removed by the owning package:

```
/usr/sadm/bin/smcregister scripts \
     /usr/sadm/lib/myProduct/MyProduct_reg.sh \
     /usr/sadm/lib/myProduct/MyProduct_unreg.sh
```

ENVIRONMENT VARIABLES

See `environ`(5) for descriptions of the following environment variables that affect the execution of `smcregister`:

JAVA_HOME — If you do not specify this environment variable, your `PATH` is searched for a suitable java. Otherwise, the `/usr/j2se` location is used.

EXIT STATUS

The following exit values are returned:

`0` — Successful completion.

`1` — An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmc

SEE ALSO `smc`(1M), `smcconf`(1M), `smccompile`(1M), `attributes`(5), `environ`(5)

NOTES All standard shell quoting rules apply.

Legacy applications (`X-windows`, command-line, and web-based applications) are handled differently from native Solaris Management Console tools. Legacy tools are handled by an instantiation of a native Solaris Management Console tool, `LegacyAppLauncher`, which, through the toolbox, is given the necessary information to run the legacy application: path, options, and so forth. Thus, you do not register a legacy application into the repository as you would a native Solaris Management Console tool. Instead, legacy applications appear only in toolboxes.

Registration and unregistration of tools, services, libraries, and properties do not take effect until the Solaris Management Console server is restarted. Run `/etc/init.d/init.wbem stop` followed by `/etc/init.d/init.wbem start`

NAME

smcron – manage jobs in the crontab database

SYNOPSIS

/usr/sadm/bin/smcron *subcommand* [*auth_args*] - – [*subcommand_args*]

DESCRIPTION

The smcron command manages jobs in the crontab(1) database.

subcommands

smcron *subcommand*s are:

add	Adds a job to the crontab(1) database. To add a job, the administrator must have the solaris.jobs.user authorization. To add a job to another user's crontab file, the administrator must have the solaris.jobs.admin authorization.
delete	Deletes a job from the crontab(1) database. To delete a job, the administrator must have the solaris.jobs.user authorization. To delete a job from another user's crontab file, the administrator must have the solaris.jobs.admin authorization.
list	Lists one or more jobs in the crontab(1) database. To list all jobs, the administrator must have the solaris.jobs.user authorization. To list a job in another user's crontab file, the administrator must have the solaris.jobs.admin authorization. No authorization is needed to list a user's own jobs.
modify	Modifies a job in the crontab(1) database. To modify a job, the administrator must have the solaris.jobs.user authorization. To modify a job in another user's crontab file, the administrator must have the solaris.jobs.admin authorization.

OPTIONS

The smcron authentication arguments, *auth_args*, are derived from the smc(1M) arg set and are the same regardless of which subcommand you use. The smcron command requires the Solaris Management Console to be initialized for the command to succeed (see smc(1M)). After rebooting the Solaris Management Console server, the first Solaris Management Console connection might time out, so you might need to retry the command.

The subcommand-specific options, *subcommand_args*, must come after the *auth_args* and must be separated from them by the - – option.

auth_args

The valid *auth_args* are -D, -H, -l, -p, -r, and -u; they are all optional. If no *auth_args* are specified, certain defaults will be assumed and the user may be prompted for additional information, such as a password for authentication purposes. These letter options can also be specified by their equivalent option words preceded by a double dash. For example, you can use either -D or - –domain with the *domain* argument.

-D | - –domain *domain*
Specifies the default domain that you want to manage. smcron accepts only file for this option. file is also the default value.

The `file` default domain means that changes are local to the server. Toolboxes can change the domain on a tool-by-tool basis; this option specifies the domain for all other tools.

`-H | - –hostname` *host_name:port*
Specifies the *host_name* and *port* to which you want to connect. If you do not specify a *port*, the system connects to the default port, `898`. If you do not specify *host_name:port*, the Solaris Management Console connects to the local host on port `898`. You may still have to choose a toolbox to load into the console. To override this behavior, use the `smc`(1M) `-B` option, or set your console preferences to load a "home toolbox" by default.

`-l | - –rolepassword` *role_password*
Specifies the password for the *role_name*. If you specify a *role_name* but do not specify a *role_password*, the system prompts you to supply a *role_password*. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-p | - –password` *password*
Specifies the password for the *user_name*. If you do not specify a password, the system prompts you for one. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-r | - –rolename` *role_name*
Specifies a role name for authentication. If you do not specify this option, no role is assumed.

`-u | - –username` *user_name*
Specifies the user name for authentication. If you do not specify this option, the user identity running the console process is assumed.

`- –`
This option is required and must always follow the preceding options. If you do not enter the preceding options, you must still enter the `- –` option.

subcommand_args

For the time-related subcommands described below, `-m`, `-M`, `-t`, and `-w`, you can enter multiple arguments, separated only by commas. `smcron` will construct `crontab` entries appropriate for your arguments. See EXAMPLES.

Note: Descriptions and other arg options that contain white spaces must be enclosed in double quotes.

- For subcommand `add`:

`-c` *command*	Specifies the command that you want to run.
`-h`	(Optional) Displays the command's usage statement.
`-m` *day_of_month*	(Optional) Specifies the day of the month you want to run the job. Valid values are 1–31. If you specify both `-t` and `-m` options, the job executes one day per month at the time specified by `-t`.

`-M` *month* (Optional) Specifies the month that you want to run the job. Valid values are 1–12. If you specify both `-t` and `-M` options, the job executes during the specified month at the time specified by `-t`.

`-n` *name* Specifies the unique name of the job.

`-o` *owner* (Optional) Specifies the user name that is the owner of the job. If you do not specify this option, the user name specified by the `-U` option is assumed.

`-t` *time_of_day* Specifies the time (in *hh*:*mm*) that you want to execute the command. If no other time-related options are specified (`-m`, `-M`, or `-w`), the job executes every day at the time specified by `-t`. If you specify both `-t` and `-w` options, the job executes one day per week at the time specified by `-t`. If you specify both `-t` and `-m` options, the job executes one day per month at the time specified by `-t`. If you specify both `-t` and `-M` options, the job executes each day during the specified month at the time specified by `-t`.

`-w` *day_of_week* (Optional) Specifies the day of the week you want to execute the command. Valid values are as follows:

- `0`=Sunday
- `1`=Monday
- `2`=Tuesday
- `3`=Wednesday
- `4`=Thursday
- `5`=Friday
- `6`=Saturday

If you specify both `-t` and `-w` options, the job executes one day per week at the time specified by `-t`.

- For subcommand `delete`:

  `-h` (Optional) Displays the command's usage statement.

  `-n` *name* Specifies the unique name of the job.

  `-o` *owner* (Optional) Specifies the user name that is the owner of the job. If you do not specify this option, the user name specified by the `-U` option is assumed.

- For subcommand `list`:

  `-f` *n*|*s*|*v* (Optional) Specifies the format of the output. See EXAMPLES for examples of each output type.

  - *n* — Displays the data in native format, as it appears in the `crontab`(1) database.
  - *s* — Default format. Displays the data in summary format.
  - *v* — Displays the data in verbose format.

`-h` (Optional) Displays the command's usage statement.

`-o` *owner* (Optional) Lists all jobs for the specified owner (user name). If you do not specify this option, all jobs in the `crontab`(1) database are listed.

- For subcommand `modify`:

  `-c` *command* (Optional) Specifies the command that you want to run.

  `-h` (Optional) Displays the command's usage statement.

  `-m` *day_of_month* (Optional) Specifies the day of the month you want to run the job. Valid values are 1–31. If you specify both `-t` and `-m` options, the job executes one day per month at the time specified by `-t`.

  `-M` *month* (Optional) Specifies the month that you want to run the job. Valid values are 1–12. If you specify both `-t` and `-M` options, the job executes during the specified month at the time specified by `-t`.

  `-n` *name* Specifies the current unique name of the job.

  `-N` *new_name* (Optional) Specifies the new unique name of the job.

  `-o` *owner* (Optional) Specifies the user name that is the owner of the job. If you do not specify this option, the user name specified by the `-U` option is assumed.

  `-O` *new_owner* (Optional) Specifies the new owner of the job.

  `-t` *time_of_day* (Optional) Specifies the time (in *hh*:*mm*) that you want to execute the command. If no other time-related options are specified (`-m`, `-M`, or `-w`), then the job executes every day at the time specified by `-t`. If you specify both `-t` and `-w` options, the job executes one day per week at the time specified by `-t`. If you specify both `-t` and `-m` options, the job executes one day per month at the time specified by `-t`. If you specify both `-t` and `-M`, then the job executes each day during the specified month at the time specified by `-t`.

  `-w` *day_of_week* (Optional) Specifies the day of the week you want to execute the command. Valid values are as follows:

  - `0`=Sunday
  - `1`=Monday
  - `2`=Tuesday
  - `3`=Wednesday
  - `4`=Thursday
  - `5`=Friday
  - `6`=Saturday

  If you specify both `-t` and `-w` options, the job executes one day per week at the time specified by `-t`.

EXAMPLES

EXAMPLE 1 Adding a Job

The following adds a new job, owned by `root`, that removes the old log files from `/tmp` daily at 1:30 AM.

```
./smcron add -H myhost -u root -p mypassword -- -n "Remove old logs" \
             -t 1:30 -c "rm /tmp/*.log" -o root
```

EXAMPLE 2 Deleting a Job

The following deletes the job `Remove old logs` owned by `root`:

```
./smcron delete -H myhost -u root -p mypassword -- \
          -n "Remove old logs" -o root
```

EXAMPLE 3 Listing Jobs in Native Format

The following lists all jobs in native, or `crontab`(1), format:

```
./smcron list -H myhost -u root -p mypassword -- -f n
MINUTE HOUR DATE MONTH DAY COMMAND

10 3 * * O,4 /etc/cron.d/logchecker
10 3 * * o /usr/lib/newsyslog
15 3 * * 0 /usr/lib/fs/nfs/nfsfind
1 2 * * * [ -x /usr/sbin/rtc ] && /usr/sbin/rtc -c > /dev/null 2>&1
```

EXAMPLE 4 Listing Jobs in Standard Format

The following lists all jobs owned by `lp` in standard format:

```
./smcron list -H myhost -u root -p mypassword -- -f s -o lp
NAME::OWNER::SCHEDULE::COMMAND

NoName_1765663371::lp::Weekly on Sundays at 3:13 AM::cd /var/lp/logs;
   if [ -f requests ]; then if [ -f requests.1 ]; then /bin/mv requests.1
   requests.2; fi; /usr/bin/cp requests requests.1; > requests; fi
NoName_512822673::lp::Weekly on Sundays at 4:15 AM::cd /var/lp/logs;
   if [ -f lpsched ]; then if [ -f lpsched.1 ]; then /bin/mv lpsched.1
   lpsched.2; fi; /usr/bin/cp lpsched lpsched.1; >lpsched; fi
```

EXAMPLE 5 Listing jobs in verbose format

The following lists all jobs in verbose format:

```
./smcron list -H myhost -u root -p mypassword -- -f v
NAME::OWNER::SCHEDULE::NEXT_RUN::STATUS::COMMAND

NoName_1075488942::root::Advanced::::Finished on Feb 10 3:10 with code 1
::/etc/cron.d/logchecker
databackup::root::Weekly on Sundays at 3:10 AM::3/19/00 3:10 AM
::Finished on Sep 19 3:10::/usr/lib/newsyslog
runlog::root::Daily at 2:01 AM::3/14/00 2:01 AM::Finished on Feb 11
```

EXAMPLE 5 Listing jobs in verbose format *(Continued)*

```
 2:01 AM::/usr/sbin/rtc
```

EXAMPLE 6 Changing a Job

The following modifies the job `Remove old logs` owned by `root` to execute daily at 2:00 AM:

```
./smcron modify -H myhost -u root -p mypassword -- -n "Remove old logs" \
          -o root -t 2:00
```

EXAMPLE 7 Specifying Multiple Time Arguments

`smcron` allows you to specify a range of times for all of its time-related subcommands, `-m`, `-M`, `-t`, and `-w`. For example, the following command:

```
# smcron add -u root -p xxxx -- -n cronjob1 -w 1-4,5 \
-t 12:00,13:15,14:30 -c ls
```

...creates the following entry in `crontab`:

```
0,15,30 12,13,14 * * 1,2,3,4,6 ls #cronjob1
```

This job would run on Monday through Thursday and Saturday at the following times:

```
12:00 12:15 12:30
13:00 13:15 13:30
14:00 14:15 14:30
```

ENVIRONMENT VARIABLES

See `environ`(5) for a description of the `JAVA_HOME` environment variable, which affects the execution of the `smcron` command. If this environment variable is not specified, the `/usr/java` location is used. See `smc`(1M).

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` Invalid command syntax. A usage message displays.

`2` An error occurred while executing the command. An error message displays.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmga

SEE ALSO | `crontab`(1), `cron`(1M), `smc`(1M), `attributes`(5), `environ`(5)

NAME

smdiskless – manage diskless client support for a server

SYNOPSIS

/usr/sadm/bin/smdiskless *subcommand* [*auth_args*] - – [*subcommand_args*]

DESCRIPTION

The smdiskless command manages diskless client support for a server.

smdiskless *subcommand*s are:

add	Adds a new diskless client to a server. There are two usages for this command. The user can either specify all the optional arguments directly on the command line, or provide a sysidcfg(4) formatted file as input. A future enhancement will allow specifying both a sysidcfg(4) formatted file and optional arguments, which will override the values in the sysidcfg(4) file.
delete	Deletes an existing diskless client from the system databases and removes any server support associated with the host, depending on the *os_server* type.
list	Lists existing diskless clients served by *os_server*.
modify	Modifies the specified attributes of the diskless client *os_server*.

OPTIONS

The smdiskless authentication arguments, *auth_args*, are derived from the smc(1M) arg set and are the same regardless of which subcommand you use. The smdiskless command requires the Solaris Management Console to be initialized for the command to succeed (see smc(1M)). After rebooting the Solaris Management Console server, the first Solaris Management Console connection might time out, so you might need to retry the command.

The subcommand-specific options, *subcommand_args*, must come after the *auth_args* and must be separated from them by the - – option.

auth_args

The valid *auth_args* are -D, -H, -l, -p, -r, and -u; they are all optional. If no *auth_args* are specified, certain defaults will be assumed and the user may be prompted for additional information, such as a password for authentication purposes. These letter options can also be specified by their equivalent option words preceded by a double dash. For example, you can use either -D or - –domain.

Note – smdiskless supports the --auth-data *file* option, which enables you to specify a file the console can read to collect authentication data. See smc(1M) for a description of this option.

-D | - –domain *domain*
Specifies the default domain that you want to manage. The syntax of *domain* is *type*:/*host_name*/*domain_name*, where *type* is nis, nis+, dns, ldap, or file; *host_name* is the name of the machine that serves the domain; and *domain_name* is the name of the domain you want to manage. (*Note:* Do *not* use nis+ for nisplus.)

If you do not specify this option, the Solaris Management Console assumes the `file` default domain on whatever server you choose to manage, meaning that changes are local to the server. Toolboxes can change the domain on a tool-by-tool basis; this option specifies the domain for all other tools.

`-H | - –hostname` *host_name:port*
Specifies the *host_name* and *port* to which you want to connect. If you do not specify a *port*, the system connects to the default port, `898`. If you do not specify *host_name:port*, the Solaris Management Console connects to the local host on port `898`. You may still have to choose a toolbox to load into the console. To override this behavior, use the `smc`(1M) `-B` option, or set your console preferences to load a "home toolbox" by default.

`-l | - –rolepassword` *role_password*
Specifies the password for the *role_name*. If you specify a *role_name* but do not specify a *role_password*, the system prompts you to supply a *role_password*. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-p | - –password` *password*
Specifies the password for the *user_name*. If you do not specify a password, the system prompts you for one. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-r | - –rolename` *role_name*
Specifies a role name for authentication. If you do not specify this option, no role is assumed.

`-u | - –username` *user_name*
Specifies the user name for authentication. If you do not specify this option, the user identity running the console process is assumed.

`- –`
This option is required and must always follow the preceding options. If you do not enter the preceding options, you must still enter the `- –` option.

subcommand_args

Note: Descriptions and other arg options that contain white spaces must be enclosed in double quotes.

- For subcommand `add`:

`-h`	(Optional) Displays the command's usage statement.
`-i` *IP_address*	Specifies the IP address for the host in the form of `129.9.200.1`.
`-e` *ethernet_addr*	Specifies the Ethernet address.
`-n` *host*	Specifies the client name.
`-o` *os_server*	(optional) Specifies the name of the host where the OS service filesystems reside. If this option is not specified, the host will be the same as that specified

in the `smc`(1M) `-D` option. This option is useful in the event that the name service server and the OS server are not the same machine.

`-x` os=*platform*

Specifies the operating system. The syntax for *platform* is as follows:

instruction_set.*implementation*.Solaris_*version*

where

- *instruction_set* is one of `sparc` or `i386`
- *implementation* is the implementation architecture, that is, `i86pc`, `sun4c` (for Solaris 2.6 and 7 only), `sun4m`, and `sun4u`.
- *version* is the Solaris version number. The supported *version* numbers are `2.6`, `2.7` (for Solaris 7), `8`, and `9`. Examples are:

  sparc.sun4c.Solaris_2.7
  sparc.sun4u.Solaris_8

`-x` root=*pathname*

(Optional) Specifies the absolute path of the directory in which to create the root directory for diskless clients. The default (and recommended) *pathname* is `/export/root/`*client_name*.

`-x` swap=*pathname*

(Optional) Specifies the absolute path of the directory in which to create the swap file for diskless clients. The default (and recommended) *pathname* is `/export/swap/`*client_name*.

`-x` swapsize=*size*

(Optional) Specifies the size, in megabytes, of the swap file for diskless clients. The default swap size is `24M`.

`-x` dump=*pathname*

(Optional) Specifies the absolute path of the dump directory for diskless clients. The default (and recommended) *pathname* is `/export/dump/`*client_name*.

`-x` dumpsize=*size*

(Optional) Specifies the size, in megabytes, of the dump file for diskless clients. The default swap size is `24M`.

`-x` pw=*Y*

(Optional) Prompts for the system's root password. The default is not to prompt. The following options are used to configure workstations on first boot by `sysidtool`(1M). They can either be specified on the command line, or in a `sysidcfg`(4) formatted file. *Note:* Use the `sysidcfg`(4) file to:

- Add a DNS client.
- Specify use of the LDAP name service.
- Specify a security policy.

The keywords and functions supported by `sysidtool` and `sysidcfg` vary among Solaris releases. Consult the man pages for your operating system release (`uname -r`) to determine the level of support available.

`-x` tz=*timezone*
(Optional) Specifies the path of a timezone file, relative to `/usr/share/lib/zoneinfo`. The default is the server's timezone.

`-x` ns=NIS | NIS+ | NONE
(Optional) Specifies the client's nameservice. This is one of NIS, NIS+, or `NONE` . Use a `sysidcfg`(4) file to specify DNS or LDAP. The default `ns` value is `NONE`, which results in the use of the `files` source in `nsswitch.conf`. See `nsswitch.conf`(4) for a description of the `files` source.

`-x` nameserver=*hostname*
(Optional) Specifies the nameserver's hostname. The default is the server's nameserver.

`-x` domain=*domain*
(Optional) Specifies the client's domain. The default is the server's domain.

`-x` nameserver_ipaddress=*ip_address*
(Optional) Specifies the nameserver's IP address.

`-x` netmask=*ip_address*
(Optional) Specifies the client's IP address netmask. The default is the server's netmask.

`-x` locale=*locale*
(Optional) Specifies the client's system locale. The default is the C locale.

`-x` terminal=*term*
(Optional) Specifies the workstation's terminal type, typically, `sun` or `xterms`.

`-x` passwd=*root_password*
(Optional) Specifies the system's root password. The default is no password.

`-x` sysidcfg=*path_to_sysidcfg_file*
(Optional) Specifies the file to be placed in the `/etc` directory of the diskless client. On first boot, `/etc/.UNCONFIGURED` exists and `sysidtool`(1M) will run. If a file called `/etc/sysidcfg` exists, `sysidtool`(1M) reads this file and uses the information for system configuration.

- For subcommand `delete`:

`-h` (Optional) Displays the command's usage statement.

`-n` *host* Specifies the hostname of the diskless client to delete. This host is deleted from relevant tables and OS Services for this client are deleted.

`-o` *os_server* (Optional) Specifies the name of the host where the OS service filesystems reside. If this option is not specified, the host will be the same as that specified in the `smc`(1M) `-D` option. This option is useful in the event that the name service server and the OS server are not the same machine.

- For subcommand `list`:

  `-h` (Optional) Displays the command's usage statement.

  `-o` *os_server* (Optional) Specifies the name of the host where the OS service filesystems reside. If this option is not specified, the host will be the same as that specified in the `smc`(1M) `-D` option. This option is useful in the event that the name service server and the OS server are not the same machine.

- For subcommand `modify`:

  `-e` *ethernet_addr* Changes the specified diskless client's ethernet address to *ethernet_addr*.

  `-h` (Optional) Displays the command's usage statement.

  `-n` *host* Specifies the host name of the diskless client to modify.

  `-o` *os_server* (Optional) Specifies the name of the host where the OS service filesystems reside. If this option is not specified, the host will be the same as that specified in the `smc`(1M) `-D` option. This option is useful in the event that the name service server and the OS server are not the same machine.

  `-x` tz=*timezone* (Optional) Changes the specified diskless client's timezone.

EXAMPLES

EXAMPLE 1 Creating a new diskless client

The following command adds a new diskless client named `client1` which will run Solaris 9 on a `sun4u` machine:

```
example% /usr/sadm/bin/smdiskless add -- -i 129.9.200.1 \
            -e 8:0:11:12:13:14 -n client1 -x os=sparc.sun4u.Solaris_9 \
            -x root=/export/root/client1 -x swap=/export/swap/client1 \
            -x swapsize=32 -x tz=US/Eastern -x locale=en_US
```

EXAMPLE 2 Deleting an existing diskless client

The following command deletes the diskless client named `client1` from the OS server named `osserver`, where the OS server is using NIS+ and the NIS+ server is `nisplusserve`:

```
example% /usr/sadm/bin/smdiskless delete \
            -D nisplus:/nisplusserver/my.domain.com -- \
            -o osserver -n client1
```

EXAMPLE 3 Listing the diskless clients served by a host

The following command lists the diskless clients running on the OS server, `osserver`:

```
example% /usr/sadm/bin/smdiskless list -D file:/osserver/osserver -- \
            -o osserver
```

EXAMPLE 4 Modifying the attributes of the diskless client host

The following command modifies the ethernet address for the client named `client1` on the OS server, `osserver`, to be `8:0:11:12:13:15`:

```
example% /usr/sadm/bin/smdiskless modify -D file:/osserver/osserver -- \
            -o osserver -n client1 -e 8:0:11:12:13:15
```

ENVIRONMENT VARIABLES

See `environ`(5) for a description of the `JAVA_HOME` environment variable, which affects the execution of the `smdiskless` command. If this environment variable is not specified, the `/usr/java1.2` location is used. See `smc`(1M).

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` Invalid command syntax. A usage message displays.

`2` An error occurred while executing the command. An error message displays.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWdclnt

SEE ALSO

`smc`(1M), `smosservice`(1M), `sysidtool`(1M), `nsswitch.conf`(4), `sysidcfg`(4), `attributes`(5), `environ`(5)

NAME

smexec – manage entries in the exec_attr database

SYNOPSIS

/usr/sadm/bin/smexec *subcommand* [*auth_args*] - – [*subcommand_args*]

DESCRIPTION

The smexec command manages an entry in the exec_attr(4) database in the local /etc files name service or a NIS or NIS+ name service.

subcommands

smexec *subcommand*s are:

add — Adds a new entry to the exec_attr(4) database. To add an entry to the exec_attr database, the administrator must have the solaris.profmgr.execattr.write authorization.

delete — Deletes an entry from the exec_attr(4) database. To delete an entry from the exec_attr database, the administrator must have the solaris.profmgr.execattr.write authorization.

modify — Modifies an entry in the exec_attr(4) database. To modify an entry in the exec_attr database, the administrator must have the solaris.profmgr.execattr.write authorization.

OPTIONS

The smexec authentication arguments, *auth_args*, are derived from the smc(1M) arg set and are the same regardless of which subcommand you use. The smexec command requires the Solaris Management Console to be initialized for the command to succeed (see smc(1M)). After rebooting the Solaris Management Console server, the first Solaris Management Console connection might time out, so you might need to retry the command.

The subcommand-specific options, *subcommand_args*, must come after the *auth_args* and must be separated from them by the - – option.

auth_args

The valid *auth_args* are -D, -H, -l, -p, -r, and -u; they are all optional. If no *auth_args* are specified, certain defaults will be assumed and the user may be prompted for additional information, such as a password for authentication purposes. These letter options can also be specified by their equivalent option words preceded by a double dash. For example, you can use either -D or - –domain with the *domain* argument.

-D | - –domain *domain*
Specifies the default domain that you want to manage. The syntax of *domain* is *type*:/*host_name*/*domain_name*, where *type* is nis, nisplus, dns, ldap, or file; *host_name* is the name of the machine that serves the domain; and *domain_name* is the name of the domain you want to manage. (*Note:* Do *not* use nis+ for nisplus.)

If you do not specify this option, the Solaris Management Console assumes the file default domain on whatever server you choose to manage, meaning that changes are local to the server. Toolboxes can change the domain on a tool-by-tool basis; this option specifies the domain for all other tools.

-H | - –hostname *host_name:port*
Specifies the *host_name* and *port* to which you want to connect. If you do not specify a *port*, the system connects to the default port, 898. If you do not specify

host_name:port, the Solaris Management Console connects to the local host on port `898`. You may still have to choose a toolbox to load into the console. To override this behavior, use the `smc`(1M) `-B` option, or set your console preferences to load a "home toolbox" by default.

`-l | - –rolepassword` *role_password*
Specifies the password for the *role_name*. If you specify a *role_name* but do not specify a *role_password*, the system prompts you to supply a *role_password*. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-p | - –password` *password*
Specifies the password for the *user_name*. If you do not specify a password, the system prompts you for one. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-r | - –rolename` *role_name*
Specifies a role name for authentication. If you do not specify this option, no role is assumed.

`-u | - –username` *user_name*
Specifies the user name for authentication. If you do not specify this option, the user identity running the console process is assumed.

`- –`
This option is required and must always follow the preceding options. If you do not enter the preceding options, you must still enter the `- –` option.

subcommand_args

Note: Descriptions and other arg options that contain white spaces must be enclosed in double quotes.

- For subcommand `add`:

`-c` *command_path*	Specifies the full path to the command associated with the new `exec_attr` entry.
`-g` *egid*	(Optional) Specifies the effective group ID that executes with the command.
`-G` *gid*	(Optional) Specifies the real group ID that executes with the command.
`-h`	(Optional) Displays the command's usage statement.
`-n` *profile_name*	Specifies the name of the profile associated with the new `exec_attr` entry.
`-t` *type*	Specifies the type for the command. Currently, the only acceptable value for *type* is `cmd`.
`-u` *euid*	(Optional) Specifies the effective user ID that executes with the command.

`-U` *uid* (Optional) Specifies the real user ID that executes with the command.

- For subcommand `delete`:

  `-c` *command_path* Specifies the full path to the command associated with the `exec_attr` entry.

  `-h` (Optional) Displays the command's usage statement.

  `-n` *profile_name* Specifies the name of the profile associated with the `exec_attr` entry.

  `-t` *type* Specifies the type `cmd` for command. Currently, the only acceptable value for *type* is `cmd`.

- For subcommand `modify`:

  `-c` *command_path* Specifies the full path to the command associated with the `exec_attr` entry that you want to modify.

  `-g` *egid* (Optional) Specifies the new effective group ID that executes with the command.

  `-G` *gid* (Optional) Specifies the new real group ID that executes with the command.

  `-h` (Optional) Displays the command's usage statement.

  `-n` *profile_name* Specifies the name of the profile associated with the `exec_attr` entry.

  `-t` *type* Specifies the type `cmd` for command. Currently, the only acceptable value for *type* is `cmd`.

  `-u` *euid* (Optional) Specifies the new effective user ID that executes with the command.

  `-U` *uid* (Optional) Specifies the new real user ID that executes with the command.

EXAMPLES

EXAMPLE 1 Creating an exec_attr database entry

The following creates a new `exec_attr` entry for the `User Manager` profile on the local file system. The entry type is `cmd` for the command `/usr/bin/cp`. The command has an effective user ID of `0` and an effective group ID of `0`.

```
./smexec add -H myhost -p mypasswd -u root -- -n "User Manager" \
          -t cmd -c /usr/bin/cp -u 0 -g 0
```

EXAMPLE 2 Deleting an exec_attr database entry

The following example deletes an `exec_attr` database entry for the `User Manager` profile from the local file system. The entry designated for the command `/usr/bin/cp` is deleted.

EXAMPLE 2 Deleting an exec_attr database entry *(Continued)*

```
./smexec delete -H myhost -p mypasswd -u root -- -n "User Manager" \
          -t cmd -c /usr/bin/cp
```

EXAMPLE 3 Modifying an exec_attr database Entry

The following modifies the attributes of the `exec_attr` database entry for the `User Manager` profile on the local file system. The `/usr/bin/cp` entry is modified to execute with the real user ID of `0` and the real group ID of `0`.

```
./smexec modify -H myhost -p mypasswd -u root -- -n "User Manager" \
          -t cmd -c /usr/bin/cp -U 0 -G 0
```

ENVIRONMENT VARIABLES

See `environ`(5) for a description of the `JAVA_HOME` environment variable, which affects the execution of the `smexec` command. If this environment variable is not specified, the `/usr/java` location is used. See `smc`(1M).

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` Invalid command syntax. A usage message displays.

`2` An error occurred while executing the command. An error message displays.

FILES

The following file is used by the `smexec` command:

`/etc/security/exec_attr` Execution profiles database. See `exec_attr`(4).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmga

SEE ALSO

`smc`(1M), `exec_attr`(4), `attributes`(5), `environ`(5)

NAME

smgroup – manage group entries

SYNOPSIS

/usr/sadm/bin/smgroup *subcommand* [*auth_args*] - – [*subcommand_args*]

DESCRIPTION

The smgroup command manages one or more group definitions in the group database for the appropriate files in the local /etc files name service or a NIS or NIS+ name service.

The following smgroup *subcommand*s are supported

add	Adds a new group entry. To add an entry, the administrator must have the solaris.admin.usermgr.write authorization.
delete	Deletes a group entry. You can delete only one entry at a time. To delete an entry, the administrator must have the solaris.admin.usermgr.write authorization. *Note:* You cannot delete the system groups with IDs less than 100, or the groups 60001, 60002, or 65534.
list	Lists one or more group entries in the form of a three-column list, containing the group name, group ID, and group members, separated by colons (:). To list entries, the administrator must have the solaris.admin.usermgr.read authorization.
modify	Modifies a group entry. To modify an entry, the administrator must have the solaris.admin.usermgr.write authorization.

OPTIONS

The smgroup authentication arguments, *auth_args*, are derived from the smc(1M) arg set and are the same regardless of which subcommand you use. The smgroup command requires the Solaris Management Console to be initialized for the command to succeed (see smc(1M)). After rebooting the Solaris Management Console server, the first Solaris Management Console connection might time out, so you might need to retry the command.

The subcommand-specific options, *subcommand_args*, must come after the *auth_args* and must be separated from them by the - – option.

auth_args

The valid *auth_args* are -D, -H, -l, -p, -r, and -u; they are all optional. If no *auth_args* are specified, certain defaults will be assumed and the user may be prompted for additional information, such as a password for authentication purposes. These letter options can also be specified by their equivalent option words preceded by a double dash. For example, you can use either -D or - –domain.

The following *auth_args* are supported:

-D | - –domain *domain*
Specifies the default domain that you want to manage. The syntax of *domain* is *type*:/*host_name*/*domain_name*, where *type* is nis, nisplus, dns, ldap or file; *host_name* is the name of the machine that serves the domain; and *domain_name* is the name of the domain you want to manage. (*Note:* Do *not* use nis+ for nisplus.)

If you do not specify this option, the Solaris Management Console assumes the `file` default domain on whatever server you choose to manage, meaning that changes are local to the server. Toolboxes can change the domain on a tool-by-tool basis; this option specifies the domain for all other tools.

`-H | - –hostname` *host_name:port*
Specifies the *host_name* and *port* to which you want to connect. If you do not specify a *port*, the system connects to the default port, `898`. If you do not specify *host_name:port*, the Solaris Management Console connects to the local host on port `898`. You may still have to choose a toolbox to load into the console. To override this behavior, use the `smc`(1M) `-B` option, or set your console preferences to load a "home toolbox" by default.

`-l | - –rolepassword` *role_password*
Specifies the password for the *role_name*. If you specify a *role_name* but do not specify a *role_password*, the system prompts you to supply a *role_password*. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-p | - –password` *password*
Specifies the password for the *user_name*. If you do not specify a password, the system prompts you for one. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-r | - –rolename` *role_name*
Specifies a role name for authentication. If you do not specify this option, no role is assumed.

`-u | - –username` *user_name*
Specifies the user name for authentication. If you do not specify this option, the user identity running the console process is assumed.

`- –`
This option is required and must always follow the preceding options. If you do not enter the preceding options, you must still enter the `- –` option.

subcommand_args

Descriptions and other argument options that contain white spaces must be enclosed in double quotes.

The `add` subcommand supports the following *subcommand_args*:

`-g` *gid*
(Optional) Specifies the group ID for the new group. The group ID must be a non-negative decimal integer with a maximum value of 2MB (2,147,483,647). Group IDs 0–99 are reserved for the system and should be used with care. If you do not specify a *gid*, the system automatically assigns the next available *gid*. To maximize interoperability and compatibility, administrators are recommended to assign groups using the range of GIDs below 60000 where possible.

`-h`
(Optional) Displays the command's usage statement.

`-m` *group_member1* `-m` *group_member2* . . .
(Optional) Specifies the new members to add to the group.

`-n` *group_name*
Specifies the name of the new group. The group name must be unique within a domain, contain 2–32 alphanumeric characters, begin with a letter, and contain at least one lowercase letter.

The `delete` subcommand supports the following *subcommand_args*:

`-h` (Optional) Displays the command's usage statement.

`-n` *group_name* Specifies the name of the group you want to delete.

The `list` subcommand supports the following *subcommand_args*

`-h` (Optional) Displays the command's usage statement.

`-n` *group_name* (Optional) Specifies the name of the group you want to list. If you do not specify a group name, all groups are listed.

The `modify` subcommand supports the following *subcommand_args*

`-h`
(Optional) Displays the command's usage statement.

`-m` *group_member1* `-m` *group_member2* . . .
(Optional) Specifies the new members to add to the group. Note that *group_member* overwrites the existing member list in the group file.

`-n` *group_name*
Specifies the name of the group you want to modify.

`-N` *new_group*
(Optional) Specifies the new group name. The group name must be unique within a domain, contain 2–32 alphanumeric characters, begin with a letter, and contain at least one lowercase letter.

EXAMPLES

EXAMPLE 1 Creating a Test Group

The following creates the `test_group` group entry with a group ID of `123` and adds `test_member1` and `test_member2` to the group:

```
./smgroup add -H myhost -p mypasswd -u root -- -n test_group \
          -m test_member1 -m test_member2 -g 123
```

EXAMPLE 2 Deleting a Group

The following deletes `test_group`:

```
./smgroup delete -H myhost -p mypasswd -u root -- -n test_group
```

EXAMPLE 3 Displaying All Groups

The following displays all groups in a three-column list showing the group name, group ID, and group members:

EXAMPLE 3 Displaying All Groups *(Continued)*

```
./smgroup list -H myhost -p mypasswd -u root --
```

EXAMPLE 4 Displaying a Group

The following displays the `group_1` data in a three-column list showing the group name, group ID, and group members:

```
./smgroup list -H myhost -p mypasswd -u root -- -n group_1
```

EXAMPLE 5 Renaming a Group

The following renames a group from `finance` to `accounting`:

```
./smgroup modify -H myhost -p mypasswd -u root  -- \
          -n finance -N accounting
```

ENVIRONMENT VARIABLES

See `environ`(5) for a description of the `JAVA_HOME` environment variable, which affects the execution of the `smgroup` command. If this environment variable is not specified, the `/usr/java` location is used. See `smc`(1M).

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` Invalid command syntax. A usage message displays.

`2` An error occurred while executing the command. An error message displays.

FILES

The following files are used by the `smgroup` command:

`/etc/group` Group file. See `group`(4).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmga

SEE ALSO

`smc`(1M), `group`(4), `attributes`(5), `environ`(5)

NAME

smlog – manage and view WBEM log files

SYNOPSIS

/usr/sadm/bin/smlog *subcommand* [*auth_args*] -- [*subcommand_args*]

DESCRIPTION

The `smlog` command manages WBEM log files and allows a user to view WBEM log file records.

Subcommands

The `smlog` command supports the following subcommands:

backup — Backs up the entries in the current WBEM log file. The backup command then creates a new log file and makes this log file the current log file.

delete — Deletes an existing (backed up) WBEM log file.

list — Lists the names of all the WBEM log files available for viewing.

view — Allows the user to view the contents of the specified WBEM log file.

OPTIONS

The `smlog` authentication arguments, *auth_args*, are derived from the `smc`(1M) `arg` set and are the same regardless of which subcommand you use. The `smlog` command requires the Solaris Management Console to be initialized for the command to succeed (see `smc`(1M)). After rebooting the Solaris Management Console server, the first Solaris Management Console connection might time out, so you might need to retry the command.

The subcommand-specific options, *subcommand_args*, must come after the *auth_args* and must be separated from them by the `- –` option.

auth_args

The valid *auth_args* are `-D`, `-H`, `-l`, `-p`, `-r`, and `-u`; they are all optional. If no *auth_args* are specified, certain defaults will be assumed and the user may be prompted for additional information, such as a password for authentication purposes. These letter options can also be specified by their equivalent option words preceded by a double dash. For example, you can use either `-D` or `- –domain` with the *domain* argument.

`-D | - –domain` *domain*
Specifies the default domain that you want to manage. `smlog` accepts only `file` for this option. `file` is also the default value.

The `file` default domain means that changes are local to the server. Toolboxes can change the domain on a tool-by-tool basis; this option specifies the domain for all other tools.

`-H | - –hostname` *host_name:port*
Specifies the *host_name* and *port* to which you want to connect. If you do not specify a *port*, the system connects to the default port, `898`. If you do not specify *host_name:port*, the Solaris Management Console connects to the local host on port `898`. You may still have to choose a toolbox to load into the console. To override this behavior, use the `smc -B` option (see `smc`(1M)), or set your console preferences to load a “home toolbox” by default.

`-l | - –rolepassword` *role_password*
Specifies the password for the *role_name*. If you specify a *role_name* but do not specify a *role_password*, the system prompts you to supply a *role_password*. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-p | - –password` *password*
Specifies the password for the *user_name*. If you do not specify a password, the system prompts you for one. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-r | - –rolename` *role_name*
Specifies a role name for authentication. If you do not specify this option, no role is assumed.

`-u | - –username` *user_name*
Specifies the user name for authentication. If you do not specify this option, the user identity running the console process is assumed.

`- –`
This option is required and must always follow the preceding options. If you do not enter the preceding options, you must still enter the `- –` option.

subcommand_args

Descriptions and other `arg` options that contain white spaces must be enclosed in double quotes.

The `backup` subcommand supports the following *subcommand_args*:

`-h` Displays the command's usage statement.

This *subcommand_arg* is optional.

The `delete` subcommand supports the following *subcommand_args*

`-h` Displays the command's usage statement.

This *subcommand_arg* is optional.

`-n` *name* Specifies the name of the log file you want to delete.

The `list` subcommand supports the following *subcommand_args*:

`-h` Displays the command's usage statement.

This *subcommand_arg* is optional.

The `view` subcommand supports the following *subcommand_args*:

`-h` Displays the command's usage statement.

This *subcommand_arg* is optional.

`-n` *name* Specifies the name of the log file you want to view.

`-v` Displays the data in verbose format.

This *subcommand_arg* is optional.

EXAMPLES

EXAMPLE 1 Listing WBEM Log Files

The following command lists all available WBEM log files:

```
./smlog list -H myhost -p mypasswd -u root --

Log.01/03/2001.14:38:29
Log.01/04/2001.16:34:59
Log.01/08/2001.14:13:33
Log.01/11/2001.18:39:53
Log.01/12/2001.10:31:31
Log.12/21/2000.17:41:11
```

EXAMPLE 2 Displaying a WBEM Log File

The following command displays the contents of a log file:

```
./smlog view -H myhost -p mypasswd -u root -- -n Log.01/04/2001.16:34:59

Date and Time       Client     User   Source            Severity       Cate
gory        Summary
1/5/01 5:22:47 PM   hostname1  root   Solaris_OsService Informational  Appl
ication log No services found.
1/5/01 5:21:46 PM   hostname1  root   Solaris_OsService Informational  Appl
ication log No services found.
```

The `smlog` output wraps when it exceeds `80` characters.

ENVIRONMENT VARIABLES

See `environ`(5) for a description of the `JAVA_HOME` environment variable, which affects the execution of the `smlog` command. If this environment variable is not specified, the `/usr/java1.2` location is used. See `smc`(1M).

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` Invalid command syntax. A usage message displays.

`2` An error occurred while executing the command. An error message displays.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmga

SEE ALSO

`smc`(1M), `attributes`(5), `environ`(5)

NAME

smmaillist – manage email alias entries

SYNOPSIS

/usr/sadm/bin/smmaillist *subcommand* [*auth_args*] - – [*subcommand_args*]

DESCRIPTION

The `smmaillist` command manages one or more email alias entries for the appropriate files in the local `/etc` files name service or a NIS or NIS+ name service.

subcommands

`smmaillist` *subcommand*s are:

`add` Creates a new email alias definition and adds it to the appropriate files. To add an entry, the administrator must have the `solaris.admin.usermgr.write` authorization.

`delete` Deletes an email alias entry. You can delete only one entry at a time. To delete an entry, the administrator must have the `solaris.admin.usermgr.write` authorization. *Note:* You cannot delete Postmaster or Mailer-Daemon aliases.

`list` Lists one or more email alias entries. To list an entry, the administrator must have the `solaris.admin.usermgr.read` authorization.

`modify` Modifies an email alias entry. To modify an entry, the administrator must have the `solaris.admin.usermgr.write` authorization.

OPTIONS

The `smmaillist` authentication arguments, *auth_args*, are derived from the `smc`(1M) arg set and are the same regardless of which subcommand you use. The `smmaillist` command requires the Solaris Management Console to be initialized for the command to succeed (see `smc`(1M)). After rebooting the Solaris Management Console server, the first Solaris Management Console connection might time out, so you might need to retry the command.

The subcommand-specific options, *subcommand_args*, must come after the *auth_args* and must be separated from them by the `- –` option.

auth_args

The valid *auth_args* are `-D`, `-H`, `-l`, `-p`, `-r`, and `-u`; they are all optional. If no *auth_args* are specified, certain defaults will be assumed and the user may be prompted for additional information, such as a password for authentication purposes. These letter options can also be specified by their equivalent option words preceded by a double dash. For example, you can use either `-D` or `- –domain` with the *domain* argument.

`-D | - –domain` *domain*
Specifies the default domain that you want to manage. The syntax of *domain* is *type*:/*host_name*/*domain_name*, where *type* is `nis`, `nisplus`, `dns`, `ldap`, or `file`; *host_name* is the name of the machine that serves the domain; and *domain_name* is the name of the domain you want to manage. (*Note:* Do *not* use `nis+` for `nisplus`.)

If you do not specify this option, the Solaris Management Console assumes the `file` default domain on whatever server you choose to manage, meaning that changes are local to the server. Toolboxes can change the domain on a tool-by-tool basis; this option specifies the domain for all other tools.

`-H | - –hostname` *host_name:port*
Specifies the *host_name* and *port* to which you want to connect. If you do not specify a *port*, the system connects to the default port, `898`. If you do not specify *host_name:port*, the Solaris Management Console connects to the local host on port `898`. You may still have to choose a toolbox to load into the console. To override this behavior, use the `smc`(1M) `-B` option, or set your console preferences to load a "home toolbox" by default.

`-l | - –rolepassword` *role_password*
Specifies the password for the *role_name*. If you specify a *role_name* but do not specify a *role_password*, the system prompts you to supply a *role_password*. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-p | - –password` *password*
Specifies the password for the *user_name*. If you do not specify a password, the system prompts you for one. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-r | - –rolename` *role_name*
Specifies a role name for authentication. If you do not specify this option, no role is assumed.

`-u | - –username` *user_name*
Specifies the user name for authentication. If you do not specify this option, the user identity running the console process is assumed.

`- –`
This option is required and must always follow the preceding options. If you do not enter the preceding options, you must still enter the `- –` option.

subcommand_args

Note: Descriptions and other arg options that contain white spaces must be enclosed in double quotes.

- For subcommand `add`:

  `-a` *address1* `-a` *address2* . . . (Optional) Specifies the new email address. See `sendmail`(1M).

  `-h` (Optional) Displays the command's usage statement.

  `-n` *alias_name* Specifies the name of the alias you want to add. See `sendmail`(1M).

- For subcommand `delete`:

  `-h` (Optional) Displays the command's usage statement.

  `-n` *alias_name* Specifies the alias you want to delete.

- For subcommand `list`:

`-h`	(Optional) Displays the command's usage statement.
`-n` *alias_name*	(Optional) Specifies the name of the alias you want to display. If you do not specify an alias, all aliases are listed.

- For subcommand `modify`:

`-a` *address1* `-a` *address2* . . .	(Optional) Specifies new email address(es) to replace the existing one(s). See `sendmail`(1M).
`-h`	(Optional) Displays the command's usage statement.
`-n` *alias_name*	(Optional) Specifies the name of the alias you want to modify.
`-N` *new_alias_name*	Specifies the new alias name. Use only when renaming an alias. See `sendmail`(1M).

EXAMPLES

EXAMPLE 1 Creating an alias

The following creates the `coworkers` alias and adds the following member list: `bill@machine1`, `sue@machine2`, and `me@machine3` to the alias.

```
./smmaillist add -H myhost -p mypasswd -u root -- -n coworkers \
          -a bill@machine1 -a sue@machine2 -a me@machine3
```

EXAMPLE 2 Deleting a mail alias

The following deletes the `my_alias` alias:

```
./smmaillist delete -H myhost -p mypasswd -u root -- -n my_alias
```

EXAMPLE 3 Displaying members of a mail alias

The following displays the list of members belonging to the `my_alias` alias:

```
./smmaillist list -H myhost -p mypasswd -u root -- -n my_alias
```

EXAMPLE 4 Displaying members of all mail aliases

The following displays the list of members belonging to all mail aliases:

```
./smmaillist list -H myhost -p mypasswd -u root --
```

EXAMPLE 5 Renaming a mail alias

The following renames the `current_name` mail alias to `new_name`:

```
./smmaillist modify -H myhost -p mypasswd -u root -- \
          -n current_name -N new_name
```

EXAMPLE 6 Redefining an address list

The following changes the recipients of the alias `my_alias` to `bill@machine1`. Any previous recipients are deleted from the alias.

```
./smmaillist modify -H myhost -p mypasswd -u root -- \
          -n my_alias -a bill@machine1
```

ENVIRONMENT VARIABLES

See `environ`(5) for a description of the `JAVA_HOME` environment variable, which affects the execution of the `smmaillist` command. If this environment variable is not specified, the `/usr/java` location is used. See `smc`(1M).

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` Invalid command syntax. A usage message displays.

`2` An error occurred while executing the command. An error message displays.

FILES

The following files are used by the `smmaillist` command:

`/var/mail/aliases` Aliases for `sendmail`(1M). See `aliases`(4).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmga

SEE ALSO

`sendmail`(1M), `smc`(1M), `aliases`(4), `attributes`(5), `environ`(5)

NAME

smmultiuser – manage bulk operations on user accounts

SYNOPSIS

/usr/sadm/bin/smmultiuser *subcommand* [*auth_args*] - – [*subcommand_args*]

DESCRIPTION

The `smmultiuser` command allows bulk operations on user entries in the local `/etc` filesystem or a NIS or NIS+ name service, using either an input file or piped input. *Note:* Both input files and piped input contain a cleartext (non-encrypted) password for each new user entry.

subcommands

`smmultiuser` *subcommand*s are:

`add` Adds multiple user entries to the appropriate files. To add an entry, the administrator must have the `solaris.admin.usermgr.write` authorization.

`delete` Deletes one or more user entries from the appropriate files. To delete an entry, the administrator must have the `solaris.admin.usermgr.write` authorization.

`modify` Modifies existing user entries in the user account database. To modify an entry, the administrator must have the `solaris.admin.usermgr.write` authorization. Here is the list of what can be modified using the `modify` subcommand:

1. UserName (only under certain conditions; see Note 2 in NOTES).
2. Password (only under certain conditions; see Note 3 in NOTES). To modify a password, the administrator must have the `solaris.admin.usermgr.pswd` authorization.
3. Description.
4. Primary Group ID.
5. Shell type.
6. FullName.

OPTIONS

The `smmultiuser` authentication arguments, *auth_args*, are derived from the `smc`(1M) arg set and are the same regardless of which subcommand you use. The `smmultiuser` command requires the Solaris Management Console to be initialized for the command to succeed (see `smc`(1M)). After rebooting the Solaris Management Console server, the first Solaris Management Console connection might time out, so you might need to retry the command.

The subcommand-specific options, *subcommand_args*, must come after the *auth_args* and must be separated from them by the `- –` option.

auth_args

The valid *auth_args* are `-D`, `-H`, `-l`, `-p`, `-r`, `- –trust`, and `-u`; they are all optional. If no *auth_args* are specified, certain defaults will be assumed and the user may be prompted for additional information, such as a password for authentication purposes. These letter options can also be specified by their equivalent option words preceded by a double dash. For example, you can use either `-D` or `- –domain`.

`-D | - –domain` *domain*
Specifies the default domain that you want to manage. The syntax of *domain* is *type*:/*host_name*/*domain_name*, where *type* is `nis`, `nisplus`, `dns`, `ldap`, or `file`; *host_name* is the name of the machine that serves the domain; and *domain_name* is the name of the domain you want to manage. (*Note:* Do *not* use `nis+` for `nisplus`.)

If you do not specify this option, the Solaris Management Console assumes the `file` default domain on whatever server you choose to manage, meaning that changes are local to the server. Toolboxes can change the domain on a tool-by-tool basis; this option specifies the domain for all other tools.

`-H | - –hostname` *host_name:port*
Specifies the *host_name* and *port* to which you want to connect. If you do not specify a *port*, the system connects to the default port, `898`. If you do not specify *host_name:port*, the Solaris Management Console connects to the local host on port `898`. You may still have to choose a toolbox to load into the console. To override this behavior, use the `smc`(1M) `-B` option, or set your console preferences to load a "home toolbox" by default.

`-l | - –rolepassword` *role_password*
Specifies the password for the *role_name*. If you specify a *role_name* but do not specify a *role_password*, the system prompts you to supply a *role_password*. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-p | - –password` *password*
Specifies the password for the *user_name*. If you do not specify a password, the system prompts you for one. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-r | - –rolename` *role_name*
Specifies a role name for authentication. If you do not specify this option, no role is assumed.

`- –trust`
Trusts all downloaded code implicitly. Use this option when running the terminal console non-interactively and you cannot let the console wait for user input.

If using piped input into any of the `smmultiuser` subcommands, it will now be necessary to use the `- –trust` option with the `-L` *logfile* option. See EXAMPLES.

`-u | - –username` *user_name*
Specifies the user name for authentication. If you do not specify this option, the user identity running the console process is assumed.

`- –`
This option is required and must always follow the preceding options. If you do not enter the preceding options, you must still enter the `- –` option.

subcommand_args
Note: Descriptions and other arg options that contain white spaces must be enclosed in double quotes.

- For subcommand `add`:

  `-h` (Optional) Displays the command's usage statement.

  `-i` *input_file* Specifies the input file containing the user account information. After the command is executed, the input file is removed. The input file must follow the `/etc/passwd` file format. If you do not specify the `-i` *input_file* option, you must include a *piped_input* operand immediately before the command. See EXAMPLES.

  `-L` *logfile* (Optional) Specifies the full pathname to the text file that stores the command's success/failure data. *Note:* This text file is an ASCII—formatted log file; it is different from and unrelated to the output of the normal logging mechanism that also occurs within the Log Viewer tool. The `-L` *logfile* option is used to dump additional logging information to a text file.

- For subcommand `delete`:

  `-h` (Optional) Displays the command's usage statement.

  `-i` *input_file* Specifies the input file containing the user account information. After the command is executed, the input file is removed. The input file must follow the `/etc/passwd` file format. If you do not specify the `-i` *input_file* option, you must include a *piped_input* operand immediately before the command. See EXAMPLES.

  `-L` *logfile* (Optional) Specifies the full pathname to the text file that stores the command's success/failure data.

- For subcommand `modify`:

  `-h` (Optional) Displays the command's usage statement.

  `-i` *input_file* Specifies the input file containing the user account information. After the command is executed, the input file is removed. The input file must follow the `/etc/passwd` file format. If you do not specify the `-i` *input_file* option, you must include a *piped_input* operand immediately before the command. See EXAMPLES. *Note:* When modifying passwords, use the piped input, since it is more secure than keeping passwords in a file. See Note 1 in NOTES.

  `-L` *logfile* (Optional) Specifies the full pathname to the text file that stores the command's success/failure data.

OPERANDS

The following operands are supported:

piped_input You must include *piped_input* if you do not specify an *input_file*. Include the piped input immediately before the command. The piped input must follow the `/etc/passwd` file format. See EXAMPLES. *Note:* Use the `- –trust` option when using piped input with the `-L` *logfile* option to avoid the user prompt from the Security Alert Manager, which normally asks the user whether the log file should be created. Without the `- –trust` option, the

piped input is improperly taken as the answer to the prompt before the user can answer "Y" or "N", and the logging operation will probably fail.

EXAMPLES

EXAMPLE 1 Creating multiple user accounts

The following reads in user account data from the /tmp/foo file and creates new user accounts on the local file system. The input file is formatted in the /etc/passwd format.

```
./smmultiuser add -H myhost -p mypasswd -u root --  -i /tmp/foo
```

EXAMPLE 2 Deleting multiple user accounts

The following reads in user account data from the /tmp/foo file and deletes the named user accounts from the local file system:

```
./smmultiuser delete -H myhost -p mypasswd -u root --  -i /tmp/foo
```

EXAMPLE 3 Creating a log file with piped input

The following example shows the use of the smc(1M) - –trust option that is required when creating a log file. It is applicable to the delete and modify subcommands also.

```
cat /tmp/users.txt | smmultiuser add --trust --  -L /tmp/mylog.txt
```

ENVIRONMENT VARIABLES

See environ(5) for a description of the JAVA_HOME environment variable, which affects the execution of the smprofile command. If this environment variable is not specified, the /usr/java location is used. See smc(1M).

EXIT STATUS

The following exit values are returned:

0 Successful completion.

1 Invalid command syntax. A usage message displays.

2 An error occurred while executing the command. An error message displays.

FILES

The following files are used by the smprofile command:

/etc/passwd Contains the file format to use for the *input_file* and *piped_input*. See passwd(4).

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmga

SEE ALSO `smc`(1M), `passwd`(4), `attributes`(5), `environ`(5)

NOTES

1. The file format used by both the `add` and `modify` subcommands is the `/etc/passwd` format. But there is an allowance for a mutated version of this file format that contains an extra field at the end of each line to be used for the Full Name. If the extra field is appended to the end of each line, it will be used for the Full Name value, but if it is omitted, it will be assumed that no FullName modification is being done. The extra field is separated with a colon (`:`), just like all the other fields.

   Example of regulation `/etc/passwd` entry:

   `rick2:x:101:10:description1:/home/rick2:/bin/sh`Example of `/etc/passwd` variant entry:

   `rick2:x:101:10:description1:/home/rick2:/bin/sh:Ricks_fullname`

2. The modifies are all done based on lookups of the user name in the user tables. If a user name can not be found in this lookup, a secondary check will be made to see if the uid *and* FullName can be found in the user tables. If they are both found, assume that a user rename has occurred. If neither can be found, assume that the user account does not exist and cannot be modified.
3. If no password is supplied, assume that there is no change to the password information. If a password is being changed, it should be supplied in cleartext as piped input, although this is not required. The password can be supplied in the input file also. Once read in, the password will be changed accordingly.

NAME

smosservice – manage OS services

SYNOPSIS

/usr/sadm/bin/smosservice *subcommand* [*auth_args*] - – [*subcommand_args*]

DESCRIPTION

The `smosservice` command manages OS services.

`smosservice` *subcommand*s are:

`add` Adds the specified OS services.

`delete` Deletes the specified OS services.

`list` Either lists all the installed OS services for the server if you do not specify a hostname, or lists the OS services for the specified diskless client if you do specify a hostname.

`patch` Manages patches on all existing diskless clients. For example, you can use this subcommand to initially establish a patch spool directory on an OS server. Then, you can apply the patch to the spool area, verifying the patch as needed. Once the patch exists in the spool area, you can apply the patch to the clone area. In addition, you can migrate the patched clone area to clients.

OPTIONS

The `smosservice` authentication arguments, *auth_args*, are derived from the `smc`(1M) arg set and are the same regardless of which subcommand you use. The `smosservice` command requires the Solaris Management Console to be initialized for the command to succeed (see `smc`(1M)). After rebooting the Solaris Management Console server, the first Solaris Management Console connection might time out, so you might need to retry the command.

The subcommand-specific options, *subcommand_args*, must come after the *auth_args* and must be separated from them by the `- –` option.

auth_args

The valid *auth_args* are `-D`, `-H`, `-l`, `-p`, `-r`, and `-u`; they are all optional. If no *auth_args* are specified, certain defaults will be assumed and the user may be prompted for additional information, such as a password for authentication purposes. These letter options can also be specified by their equivalent option words preceded by a double dash. For example, you can use either `-D` or `- –domain`.

`-D | - –domain` *domain*
Specifies the default domain that you want to manage. The syntax of *domain* is *type*:/*host_name*/*domain_name*, where *type* is `nis`, `nis+`, `dns`, `ldap`, or `file`; *host_name* is the name of the machine that serves the domain; and *domain_name* is the name of the domain you want to manage. (*Note:* Do *not* use `nis+` for `nisplus`.)

If you do not specify this option, the Solaris Management Console assumes the `file` default domain on whatever server you choose to manage, meaning that changes are local to the server. Toolboxes can change the domain on a tool-by-tool basis; this option specifies the domain for all other tools.

`-H | - –hostname` *host_name:port*
Specifies the *host_name* and *port* to which you want to connect. If you do not specify a *port*, the system connects to the default port, `898`. If you do not specify *host_name:port*, the Solaris Management Console connects to the local host on port `898`. You may still have to choose a toolbox to load into the console. To override this behavior, use the `smc`(1M) `-B` option, or set your console preferences to load a "home toolbox" by default.

`-l | - –rolepassword` *role_password*
Specifies the password for the *role_name*. If you specify a *role_name* but do not specify a *role_password*, the system prompts you to supply a *role_password*. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-p | - –password` *password*
Specifies the password for the *user_name*. If you do not specify a password, the system prompts you for one. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-r | - –rolename` *role_name*
Specifies a role name for authentication. If you do not specify this option, no role is assumed.

`-u | - –username` *user_name*
Specifies the user name for authentication. If you do not specify this option, the user identity running the console process is assumed.

`- –`
This option is required and must always follow the preceding options. If you do not enter the preceding options, you must still enter the `- –` option.

subcommand_args

Note: Descriptions and other arg options that contain white spaces must be enclosed in double quotes.

- For subcommand `add`:

`-h`	(Optional) Displays the command's usage statement.
`-o` *os_server*	(Optional) Specifies the name of the host where the OS service filesystems reside. If this option is not specified, the host will be the same as that specified in the `smc`(1M) `-D` option. This option is useful in the event that the name service server and the OS server are not the same machine.
`-x` mediapath=*path*	Specifies the full path to the Solaris CD image.
`-x` platform=*platform*	Specifies the OS service to add. The instruction architecture, machine class, OS, and version are given in the form: *instruction_set*.*machine_class*.Solaris_*os_version*for example, `sparc.sun4m.Solaris_9`

`-x` cluster=*cluster* — Specifies the Solaris cluster to install. For example, `SUNWCall`.

`-x` locale=*locale*[*locale*, . . .] — (Optional) Specifies the locales to install from the specified cluster. A comma-delimited list of locales can be specified.

- For subcommand `delete`:

  `-h` — (Optional) Displays the command's usage statement.

  `-o` *os_server* — (Optional) Specifies the name of the host where the OS service filesystems reside. If this option is not specified, the host will be the same as that specified in the `smc`(1M) `-D` option. This option is useful in the event that the name service server and the OS server are not the same machine.

  `-x` rmplatform=*platform* — Specifies the OS service to remove. The instruction architecture, machine class, OS, and version are given in the form:

  instruction_set.*machine_class*.Solaris_*os_version*for example, `sparc.all.Solaris_9`. *Note:* Only a machine class of `all` is supported.

- For subcommand `list`:

  `-h` — (Optional) Displays the command's usage statement.

  `-o` *os_server* — (Optional) Specifies the name of the host where the OS service filesystems reside. If this option is not specified, the host will be the same as that specified in the `smc`(1M) `-D` option. This option is useful in the event that the name service server and the OS server are not the same machine.

- For subcommand `patch`:

  `-a` *patch_directory/patch_ID* — Adds the specified patch, *patch_ID*, to the spool directory. *patch_directory* specifies the source path of the patch to be spooled which includes the patchid directory name. Patches are spooled to `/export/diskless/Patches/`. If the patch being added obsoletes an existing patch in the spool, the obsolete patch is moved to the archive area, `/export/diskless/Patches/Archive` (to be restored if this new patch is ever removed).

  `-h` — (Optional) Displays the command's usage statement.

  `-m` — (Optional) Synchronizes spooled patches with offline copies of each diskless client OS service on

the server. Spooled patches and applied patches are compared so that newly spooled patches can be installed and patches recently removed from the spool can be backed out. This option does not apply to patches directly to diskless client OS services or diskless clients; the `-u` option must be used to update the services and clients with the changes. Clients are not required to be down at this time, as all patching is done off line. *Note:* The server is fully available during this operation.

`-P` — Lists all currently spooled patches with an associated synopsis. The list is split up into sections detailing the patches for each OS and architecture in this format:

```
Solaris os_rel1 architecture1:
patchid Synopsis
patchid Synopsis
......
Solaris os_rel1 architecture2:
patchid Synopsis
......
```

`-r` *patchid* — Removes the specified patchid from the spool if it is not a requirement for any of the other patches in the spool. All archived patches that were obsoleted by the removed patch are restored to the spool.

`-U` — (Optional) Updates all diskless client OS services and diskless clients with any changes after synchronizing patches with the `-m` option. Clients must be brought down during this operation. Once execution has completed, each client should be booted again.

EXAMPLES

EXAMPLE 1 Creating a new OS service

The following command adds an OS service for Solaris 9 for the `sun4u` machine class where the OS server is *not* using a name service:

```
example% /usr/sadm/bin/smosservice add -- \
            -x mediapath=/net/imageserver/5.8/sparc \
            -x platform=sparc.sun4u.Solaris_9 \
            -x cluster=SUNWCXall -x locale=en_US
```

The following command adds an OS service for Solaris 9 for the `sun4u` machine class where the OS server is using NIS, the NIS server is `nisserver`, the OS server is `osserver`, and the port to which you connect on `osserver` is 898:

```
example% /usr/sadm/bin/smosservice add -D nis:/nisserver/my.domain.com -- \
            -H osserver:898 -- \
            -x mediapath=/net/imageserver/5.8/sparc \
            -x platform=sparc.sun4u.Solaris_9 \
            -x cluster=SUNWCXall -x locale=en_US \
            -o osserver
```

In the preceding example, the OS service is placed in `/export` on `osserver`, while the `hosts.byaddr`, `ethers`, and `bootparams` maps are updated on the NIS server.

EXAMPLE 2 Deleting an OS service

The following command deletes the OS service for Solaris 9 for the `sun4u` machine class where the OS server is using NIS, the NIS server is `nisserver`, and the OS server is `osserver`:

```
example% /usr/sadm/bin/smosservice delete\
            -D nis:/nisserver/my.domain.com -- \
            -x rmplatform=sparc.all.Solaris_9 \
            -o osserver
```

EXAMPLE 3 Listing installed OS services

The following command lists the OS services installed on the machine, `osserver`:

```
example% /usr/sadm/bin/smosservice list \
            -D file:/osserver/osserver -- -o osserver
```

ENVIRONMENT VARIABLES

See `environ`(5) for a description of the `JAVA_HOME` environment variable, which affects the execution of the `smosservice` command. If this environment variable is not specified, the `/usr/java1.2` location is used. See `smc`(1M).

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` Invalid command syntax. A usage message displays.

`2` An error occurred while executing the command. An error message displays.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWdclnt

SEE ALSO

`smc`(1M), `smdiskless`(1M), `attributes`(5), `environ`(5)

NAME

smpatch – manage patches

SYNOPSIS

/usr/sadm/bin/smpatch *subcommand* [*auth_args*] -- [*subcommand_args*]

DESCRIPTION

The `smpatch` command manages patch installation on single or multiple machines, analyzes patch requirements, and downloads required patches.

To run this command on multiple machines, you must first create a role on each machine that contains the necessary rights to perform patch management. In addition, all of the machines on which you want to install patches must run the same version of the Solaris operating environment, have the same hardware architecture, and have the same patches installed.

Subcommands

The following subcommands are supported:

add — Installs patches on single or multiple machines. You can either specify the patch IDs and the machine names directly on the command line, or you can specify a *patchlist_file* and a *machinelist_file* that contains the list of patches and the machines on which you want to install the patches, respectively.

analyze — Analyzes and lists the patches required for a specified machine.

You must first install the PatchPro application on the host machine before you run the analyze command. To install PatchPro, download the packages from the Sun Microsystems web site and follow the instructions to install the software on your machine.

download — Downloads patches from the SunSolve Online database to the patch directory. You can either specify the patches that you want to download, or you can specify a machine name to download the recommended patches for that machine.

You must first install the PatchPro application on the host machine before you run the download command. To install PatchPro, download the packages from the Sun Microsystems web site and follow the instructions to install the software on your machine.

remove — Removes a single patch from a system.

OPTIONS

There are two kinds of options: authentication arguments (*auth_args*) and subcommand arguments (*subcommand_args*).

Authentication Arguments

The `smpatch` authentication arguments, *auth_args*, are derived from the `smc`(1M) argument set and are the same regardless of which subcommand you use.

Valid *auth_args* are `-D`, `-H`, `-l`, `-p`, `-r`, and `-u`; they are all optional. If no *auth_args* are specified, certain defaults will be assumed and the user may be prompted for additional information, such as a password for authentication purposes.

The single letter options can also be specified by their equivalent option words preceded by a double dash. For example, you can use either `-D` or `--domain`.

Descriptions and other arg options that contain white spaces must be enclosed in double quotes.

The following authentication arguments (*auth_args*) are supported:

`-D | --domain` *domain*
Specifies the default domain that you want to manage. `smpatch` only accepts files for this option.

If you do not specify this option, the Solaris Management Console assumes the file default domain on whatever server you choose to manage, meaning that changes are local to the server. Toolboxes can change the domain on a tool-by-tool basis; this option specifies the domain for all other tools.

`-H | --hostname` *host_name*:*port*
Specifies the host and port to which you want to connect. If you do not specify a port, the system connects to the default port, `898`. If you do not specify a host (*host_name*:*port*, the Solaris Management Console connects to the local host on port `898`. You may still have to choose a toolbox to load into the console. To override this behavior, use the `smc -B` option, or set your console preferences to load a `home toolbox` by default.

`-l | --rolepassword` *role_password*
Specifies the password for the *role_name*. If you specify a *role_name* but do not specify a *role_password*, the system prompts you to supply a *role_password*. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-p | --password` *password*
Specifies the password for the *user_name*. If you do not specify a password, the system prompts you for one. Because passwords specified on the command line can be seen by any user on the system, this option is considered insecure.

`-r | --rolename` *role_name*
Specifies a role name for authentication. If you do not specify this option, no role is assumed.

`-u | --username` *user_name*
Specifies the user name for authentication. If you do not specify this option, the user identity running the console process is assumed.

`--`
This option is required and must always follow the preceding options. If you do not enter the preceding options, you must still enter the `--` option.

Subcommand Arguments

The subcommand-specific options, subcommand_args, must come after the auth_args and must be separated from them by the -- option. Enclose descriptions and *arg* options that contain white space in double quotes.

add

The add subcommand requires one of the following subcommand arguments:

`-i` *patch_id1* `-i` *patch_id2* . . .
Specifies the patch or patches that you want to install. You can specify the `-x` *idlist=patch_id_file* operand instead of specifying this option.

`-x` *idlist*=patchlist_file
Specifies a file that contains the list of patches that you want to install. You can specify the `-i` *patch_id1* option instead of specifying this operand.

The following subcommand arguments are optional for the add subcommand:

`-d` *patchdir*
Specifies the directory where the patches are located. If you do not specify this option, the default patch spool directory (`/var/sadm/spool`), located on the host specified with the `-H` authentication argument, is assumed. The patch directory has the following syntax: *machine_name*`:`*/directory_path*, where */directory_path* is a fully-qualified, shared directory. You can specify just the */directory_path* if the directory is an NFS-mounted network directory or is located on the machine on which you want to install the patches.

`-h`
Displays the command's usage statement.

`-n` *machine_name1* `-n` *machine_name2* . . .
Specifies the machine(s) on which you want to install the patches. If you do not specify this option, the machine is assumed to be be that specified with the `-H` authentication argument. You can specify the `-x` *mlist=machine_name_file* operand instead of specifying this option.

`-x` *mlist=machinelist_file*
Specifies a file that contains the list of machines to which you want to install patches. You can specify the`-n` *machine_name1* option instead of specifying this operand.

analyze

The following subcommand arguments are optional for the analyze subcommand:

`-h`	Displays the command's usage statement.
`-n` *machine_name*	Specifies the machine for which you want to display a list of patches recommended by SunSolve. If you do not specify this option, the machine is assumed to be that specified by the `-H` authentication argument.

download The `download` subcommand requires one of the following subcommand arguments:

`-i` *patch_id1* `-i` *patch_id2* . . . Specifies the patch or patches that you want to down load. You can specify the `-x` *idlist=patch_id_file* operand instead of this option or you can omit this argument in favor of `-n download_machine` option.

`-x` *idlist=patchlist_file* Specifies the file containing the list of patches that you want to download. You can specify this operand instead of specifying the `-i` *patch_id1* option.

The following subcommand arguments are optional for the `download` subcommand:

`-n` *download_machine* Specifies the machine on which you want to download the recommended patches. If you do not specify this option, the machine is assumed to be that specified by the `-H` authentication argument.

`-d` *downloaddir* Specifies the directory where the patches are downloaded. This directory must have write permission and be accessible to the download_machine. If you do not specify this option, the default patch spool directory (`/var/sadm/spool`) located on the download machine is assumed.

remove The `remove` subcommand requires the following arguments:

`-i` *patch_id* Specifies the patch that you want to remove.

The following subcommand arguments are optional for the `remove` subcommand:

`-n` *machine* Specifies the machine on which you want to remove the recommended patches. If you do not specify this option, the machine is assumed to be that specified by the `-H` authentication argument.

OPERANDS The following operands are supported:

machinelist_file Specifies the fully-qualified path and file name of the file containing the list of machine names on which you want to install the patches. The *machinelist_file* file contains one machine name per line.

patchlist_file Specifies the fully-qualified path and file name of the file containing the list of patches that you want to install. The *patchlist_file* contains one patch ID per line.

EXAMPLES

EXAMPLE 1 Installing Patches on Multiple Machines

The following example installs patches `102893-01`, `106895-09`, and `106527-05` on the machines `lab1` and `lab2`. The patches are located in the `/files/patches/s9` shared directory on the machine `fileserver`:

```
/usr/sadm/bin//smpatch add -H myhost -p mypasswd -u root -- \
-i 102893-01 -i 106895-09 -i 106527-05 \
      -d fileserver:/files/patches/s9 \
      -n lab1 -n lab2
```

EXAMPLE 2 Installing Patches Using a patch_list File

The following example specifies the `/tmp/patch/patch_file` to install patches on the machines `lab1` and `lab2`. The patches are located in the NFS network-mounted directory `/net/fileserver/export/patchspool/Solaris9`:

```
/usr/sadm/bin/smpatch add -H myhost -p mypasswd -u root -- \
     -x mlist=/tmp/patch/patch_file \
     -d /net/fileserver/export/patchspool/Solaris9 \
     -n lab1 -n lab2
```

EXAMPLE 3 Installing Patches Using a patch_list File and a machine_list File

The following example installs the patches listed in `/tmp/patch/patch_file` on the machines listed in `/tmp/patch/machine_file`. The patches are located in the default `/var/sadm/spool directory` on the default machine `myhost`.

```
/usr/sadm/bin/smpatch add -H myhost -p mypasswd -u root -- \
     -x mlist=/tmp/patch/patch_file  \
     -x mlist=/tmp/patch/machine_file
```

EXAMPLE 4 Analyzing and Downloading Patches from Sunsolve Online

The following example analyzes the `lab1` machine and downloads the assessed patches from the Sunsolve Online database to the default patch spool directory.

```
/usr/sadm/bin/smpatch download -p mypasswd -u root -- -n lab1
```

EXAMPLE 5 Downloading Patches From the Sunsolve Database

The following example downloads the `102893-01` and `106895-09` patches from the Sunsolve Online database to the `/files/patches/s8` directory located on the default machine.

```
/usr/sadm/bin/smpatch download -p mypasswd -u root -- \
          -i 102893-01 -i 106895-09 -d /files/patches/s8
```

EXAMPLE 6 Removing Patches

The following example removes patch `102893-01`:

EXAMPLE 6 Removing Patches *(Continued)*

```
/usr/sadm/bin/smpatch remove -p mypasswd -u root -- \
         -i 10283-01
```

ENVIRONMENT VARIABLES

See `environ`(5) for descriptions of the following environment variables that affect the execution of `smpatch`: `JAVA_HOME`. If this environment variable is not specified, the `/usr/java` location is used. See `smc`(1M).

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` Invalid command syntax. A usage message displays.

`2` An error occurred while executing the command. An error message displays.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmga

SEE ALSO

`arch`(1), `uname`(1), `patchadd`(1M), `patchrm`(1M), `share`(1M), `smc`(1M), `attributes`(5), `environ`(5)

NAME

smprofile – manage profiles in the prof_attr and exec_attr databases

SYNOPSIS

/usr/sadm/bin/smprofile *subcommand* [*auth_args*] - – [*subcommand_args*]

DESCRIPTION

The `smprofile` command manages one or more profiles in the `prof_attr`(4) or `exec_attr`(4) databases in the local `/etc` files name service or a NIS or NIS+ name service.

subcommands

`smprofile` *subcommand*s are:

`add` Adds a new profile (right) to the `prof_attr`(4) database. To add a profile, the administrator must have the `solaris.profmgr.write` authorization.

`delete` Deletes a profile from the `prof_attr`(4) database, deletes all associated entries from the `exec_attr`(4) database, and deletes the assigned profile from the `user_attr`(4) database. To delete a profile, the administrator must have the `solaris.profmgr.execattr.write` and `solaris.profmgr.write` authorization.

`list` Lists one or more profiles from the `prof_attr`(4) or `exec_attr`(4) databases. To list a profile, the administrator must have the `solaris.profmgr.read` authorization.

`modify` Modifies a profile in the `prof_attr`(4) database. To modify a profile, the administrator must have the `solaris.profmgr.write` authorization.

OPTIONS

The `smprofile` authentication arguments, *auth_args*, are derived from the `smc`(1M) arg set and are the same regardless of which subcommand you use. The `smprofile` command requires the Solaris Management Console to be initialized for the command to succeed (see `smc`(1M)). After rebooting the Solaris Management Console server, the first Solaris Management Console connection might time out, so you might need to retry the command.

The subcommand-specific options, *subcommand_args*, must come after the *auth_args* and must be separated from them by the `- –` option.

auth_args

The valid *auth_args* are `-D`, `-H`, `-l`, `-p`, `-r`, and `-u`; they are all optional. If no *auth_args* are specified, certain defaults will be assumed and the user may be prompted for additional information, such as a password for authentication purposes. These letter options can also be specified by their equivalent option words preceded by a double dash. For example, you can use either `-D` or `- –domain` with the *domain* argument.

`-D | - –domain` *domain*
Specifies the default domain that you want to manage. The syntax of *domain* is *type*:/*host_name*/*domain_name*, where *type* is `nis`, `nisplus`, `dns`, `ldap`, or `file`; *host_name* is the name of the machine that serves the domain; and *domain_name* is the name of the domain you want to manage. (*Note:* Do *not* use `nis+` for `nisplus`.)

If you do not specify this option, the Solaris Management Console assumes the `file` default domain on whatever server you choose to manage, meaning that changes are local to the server. Toolboxes can change the domain on a tool-by-tool basis; this option specifies the domain for all other tools.

`-H | - –hostname` *host_name:port*
Specifies the *host_name* and *port* to which you want to connect. If you do not specify a *port*, the system connects to the default port, `898`. If you do not specify *host_name:port*, the Solaris Management Console connects to the local host on port `898`. You may still have to choose a toolbox to load into the console. To override this behavior, use the `smc`(1M) `-B` option, or set your console preferences to load a "home toolbox" by default.

`-l | - –rolepassword` *role_password*
Specifies the password for the *role_name*. If you specify a *role_name* but do not specify a *role_password*, the system prompts you to supply a *role_password*. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-p | - –password` *password*
Specifies the password for the *user_name*. If you do not specify a password, the system prompts you for one. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-r | - –rolename` *role_name*
Specifies a role name for authentication. If you do not specify this option, no role is assumed.

`-u | - –username` *user_name*
Specifies the user name for authentication. If you do not specify this option, the user identity running the console process is assumed.

`- –`
This option is required and must always follow the preceding options. If you do not enter the preceding options, you must still enter the `- –` option.

subcommand_args

Note: Descriptions and other arg options that contain white spaces must be enclosed in double quotes.

- For subcommand `add`:

  `-a` *addauth1* `-a` *addauth2* . . .
  (Optional) Specifies the authorization name(s) to add to the new profile. The administrator must have the `solaris.profmgr.write` authorization and must have the corresponding "grant" authorization. A "grant" authorization is one in which the lowest component of the authorization name is replaced by the word `grant`. For example, to grant some profile the `solaris.role.write` authorization, the administrator needs that authorization and also the

`solaris.role.grant` authorization. For more information on granting authorizations, see `auth_attr`(4).

`-d` *description* — Specifies the description of the new profile.

`-h` — (Optional) Displays the command's usage statement.

`-m` *html_help* — Specifies the HTML help file name for the new profile. The help file name must be put in the `/usr/lib/help/profiles/locale/C` directory.

`-n` *name* — Specifies the name of the new profile.

`-p` *addprof1* `-p` *addprof2* . . . — (Optional) Specifies the supplementary profile name(s) to add to the new profile.

- For subcommand `delete`:

  `-h` (Optional) Displays the command's usage statement.

  `-n` *name* Specifies the name of the profile you want to delete.

- For subcommand `list`:

  `-h` — (Optional) Displays the command's usage statement.

  `-l` — (Optional) Displays the detailed output for each profile in a block of *key:value* pairs, followed by a blank line that delimits each profile block. Each *key:value* pair is displayed on a separate line. All the attributes associated with a profile from the `prof_attr` and `exec_attr` databases are displayed. If you do not specify this option, only the specified profile name(s) and associated profile description(s) are displayed.

  `-n` *name1* `-n` *name2* . . . — (Optional) Specifies the profile(s) that you want to display. If you do not specify a profile name, all profiles are displayed.

- For subcommand `modify`:

  `-a` *addauth1* `-a` *addauth2* . . . — (Optional) Specifies the authorization name(s) to add to the profile. The administrator must currently have been granted each of the specified authorizations and must have the ability to grant each of those authorizations to other users or roles. For more information on granting authorizations, see `auth_attr`(4).

  `-d` *description* — (Optional) Specifies the new description of the profile.

`-h`	(Optional) Displays the command's usage statement.
`-m` *html_help*	(Optional) Specifies the new HTML help file name of the profile. If you change this name, you must accordingly rename the help file name entered in the `/usr/lib/help/profiles/locale/C` directory.
`-n` *name*	Specifies the name of the profile you want to modify.
`-p` *addprof1* `-p` *addprof2* . . .	(Optional) Specifies the supplementary profile name(s) to add to the profile. The administrator must have the `solaris.profmgr.assign` authorization to add any profile and the `solaris.profmgr.delegate` authorization to add any profile that has been assigned to the authenticated user.
`-q` *delprof1* `-q` *delprof2* . . .	(Optional) Specifies the supplementary profile name(s) to delete from the profile. The administrator must have the `solaris.profmgr.assign` authorization to delete any profile and the `solaris.profmgr.delegate` authorization to delete any profile that has been assigned to the authenticated user.
`-r` *delauth1* `-r` *delauth2* . . .	(Optional) Specifies the authorization name(s) to delete from the profile. The administrator must have the `solaris.profmgr.write` authorization and must have the corresponding "grant" authorization. For more information about "grant" authorizations, see the `-a` option description for the `add` subcommand above.

EXAMPLES

EXAMPLE 1 Creating a new profile

The following creates a new `User Manager` profile on the local file system. The new profile description is `Manage users and groups`, and the authorizations assigned are `solaris.admin.usermgr.write` and `solaris.admin.usermgr.read`. The supplementary profile assigned is `Operator`. The help file name is `RtUserMgmt.html`.

```
./smprofile add -H myhost -p mypasswd -u root -- -n "User Manager" \
   -d "Manage users and groups" -a solaris.admin.usermgr.write \
   -a solaris.admin.usermgr.read -p Operator -m RtUserMgmt.html
```

EXAMPLE 2 Deleting a profile

The following deletes the `User Manager` profile from the local file system:

```
./smprofile delete -H myhost -p mypasswd -u root -- -n "User Manager"
```

EXAMPLE 3 Listing all profiles

The following lists all profiles and their associated profile descriptions on the local file system.

```
./smprofile list -H myhost -p mypasswd -u root --
```

EXAMPLE 4 Modifying a profile

The following modifies the `User Manager` profile on the local file system. The new profile description is `Manage world`, the new authorization assignment is `solaris.admin.usermgr.*` authorizations, and the new supplementary profile assignment is `All`. (The `-a` option argument must be enclosed in double quotes when the wildcard character (`*`) is used.)

```
./smprofile modify -H myhost -p mypasswd -u root -- -n "User Manager" \
   -d "Manage world" -a "solaris.admin.usermgr.*" -p All
```

ENVIRONMENT VARIABLES

See `environ`(5) for a description of the `JAVA_HOME` environment variable, which affects the execution of the `smprofile` command. If this environment variable is not specified, the `/usr/java` location is used. See `smc`(1M).

EXIT STATUS

The following exit values are returned:

`0`	Successful completion.
`1`	Invalid command syntax. A usage message displays.
`2`	An error occurred while executing the command. An error message displays.

FILES

The following files are used by the `smprofile` command:

`/etc/security/exec_attr`	Execution profiles database. See `exec_attr`(4).
`/etc/security/prof_attr`	Profile description database. See `prof_attr`(4).
`/etc/user_attr`	Extended user attribute database. See `user_attr`(4).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmga

SEE ALSO

`smc`(1M), `auth_attr`(4), `exec_attr`(4), `prof_attr`(4), `user_attr`(4), `attributes`(5), `environ`(5)

NAME

smrole – manage roles and users in role accounts

SYNOPSIS

/usr/sadm/bin/smrole *subcommand* [*auth_args*] - – [*subcommand_args*]

DESCRIPTION

The `smrole` command manages roles and adds or deletes users in role accounts.

subcommands

`smrole` *subcommand*s are:

`add` Adds a new role entry. To add an entry, the administrator must have the `solaris.role.write` authorization.

`delete` Deletes one or more roles. To delete an entry, the administrator must have the `solaris.role.write` authorization.

`list` Lists one or more roles. If you do not specify a role name, all roles are listed. To list an entry, the administrator must have the `solaris.admin.usermgr.read` authorization.

`modify` Adds or deletes users from a role account. To modify an entry, the administrator must have the `solaris.role.write` authorization.

OPTIONS

The `smrole` authentication arguments, *auth_args*, are derived from the `smc`(1M) arg set and are the same regardless of which subcommand you use. The `smrole` command requires the Solaris Management Console to be initialized for the command to succeed (see `smc`(1M)). After rebooting the Solaris Management Console server, the first Solaris Management Console connection might time out, so you might need to retry the command.

The subcommand-specific options, *subcommand_args*, must come after the *auth_args* and must be separated from them by the `- –` option.

auth_args

The valid *auth_args* are `-D`, `-H`, `-l`, `-p`, `-r`, and `-u`; they are all optional. If no *auth_args* are specified, certain defaults will be assumed and the user may be prompted for additional information, such as a password for authentication purposes. These letter options can also be specified by their equivalent option words preceded by a double dash. For example, you can use either `-D` or `- –domain` with the *domain* argument.

`-D | - –domain` *domain*
Specifies the default domain that you want to manage. The syntax of *domain* is *type*:/*host_name*/*domain_name*, where *type* is `nis`, `nisplus`, `dns`, `ldap`, or `file`; *host_name* is the name of the machine that serves the domain; and *domain_name* is the name of the domain you want to manage. (*Note:* Do *not* use `nis+` for `nisplus`.)

If you do not specify this option, the Solaris Management Console assumes the `file` default domain on whatever server you choose to manage, meaning that changes are local to the server. Toolboxes can change the domain on a tool-by-tool basis; this option specifies the domain for all other tools.

`-H | - –hostname` *host_name:port*
Specifies the *host_name* and *port* to which you want to connect. If you do not specify a *port*, the system connects to the default port, `898`. If you do not specify

host_name:port, the Solaris Management Console connects to the local host on port `898`. You may still have to choose a toolbox to load into the console. To override this behavior, use the `smc`(1M) `-B` option, or set your console preferences to load a "home toolbox" by default.

`-l | - –rolepassword` *role_password*
Specifies the password for the *role_name*. If you specify a *role_name* but do not specify a *role_password*, the system prompts you to supply a *role_password*. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-p | - –password` *password*
Specifies the password for the *user_name*. If you do not specify a password, the system prompts you for one. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-r | - –rolename` *role_name*
Specifies a role name for authentication. If you do not specify this option, no role is assumed.

`-u | - –username` *user_name*
Specifies the user name for authentication. If you do not specify this option, the user identity running the console process is assumed.

`- –`
This option is required and must always follow the preceding options. If you do not enter the preceding options, you must still enter the `- –` option.

subcommand_args

Note: Descriptions and other arg options that contain white spaces must be enclosed in double quotes.

- For subcommand `add`:

`-a` *adduser1* `-a` *adduser2* . . .	(Optional) Specifies the user name(s) to add to the new role. The administrator must have the `solaris.role.assign` authorization.
`-c` *comment*	(Optional) Includes a short description of the role. Consists of a string of up to 256 printable characters, excluding the colon (`:`).
`-d` *dir*	(Optional) Specifies the home directory of the new role, limited to 1024 characters.
`-F` *full_name*	(Optional) Specifies the full, descriptive name of the role. The *full_name* must be unique within a domain, and can contain alphanumeric characters and spaces. If you use spaces, you must enclose the *full_name* in double quotes.
`-G` *group1* `-G` *group2* . . .	(Optional) Specifies the new role's supplementary group membership in the system group database with the character string names of one or more

	existing groups. *Note:* You cannot assign a primary group to a role. A role's primary group is always `sysadmin` (group 14).
`-h`	(Optional) Displays the command's usage statement.
`-n` *rolename*	Specifies the name of the role you want to create.
`-p` *addprof1* `-p` *addprof2* . . .	(Optional) Specifies the profile(s) to add to the role. To assign a profile to a role, the administrator must have the `solaris.profmgr.assign` or `solaris.profmgr.delegate` authorization.
`-P` *password*	(Optional) Specifies the role's password. The *password* can contain up to eight characters. If you do not specify a password, the system prompts you for one. To set the password, the administrator must have the `solaris.admin.usermgr.pswd` authorization. *Note:* When you specify a password using the `-P` option, you type the password in plain text. Specifying a password using this method introduces a security gap while the command is running. However, if you do not specify a password (and the system prompts you for one), the echo is turned off when you type in the password.
`-s` *shell*	(Optional) Specifies the full pathname of the program used as the role's shell on login. Valid entries are `/bin/pfcsh` (C shell), `/bin/pfksh` (Korn shell), and `/bin/pfsh` (Bourne shell), the default.
`-u` *uid*	(Optional) Specifies the ID of the role you want to add. If you do not specify this option, the system assigns the next available unique ID greater than 100.
`-x` *autohome*=Y|N	(Optional) Sets the role's home directory. The home directory path in the password entry is set to `/home/`*login name*.
`-x` *perm=home_perm*	(Optional) Sets the permissions on the role's home directory. *perm* is interpreted as an octal number, and the default is `0775`.
`-x` *serv=homedir_server*	(Optional) If `-D` is `nis`, `nisplus`, or `ldap`, use this option to specify the name of the server where the user's home directory resides. Users created in a local scope must have their home directory server

created on their local machines.

- For subcommand `delete`:

  `-h` (Optional) Displays the command's usage statement.

  `-n` *rolename1* `-n` *rolename2* . . . Specifies the name of the role(s) you want to delete.

- For subcommand `list`:

  `-h` (Optional) Displays the command's usage statement.

  `-l` (Optional) Displays the output for each user in a block of *key*:*value* pairs (for example, `user name:root`), followed by a blank line that delimits each user block. Each *key*:*value* pair is displayed on a separate line. The keys are: `autohome setup`, `comment`, `home directory`, `login shell`, `primary group`, `secondary groups`, `server`, `user ID (UID)`, and `user name`.

  `-n` *role1* `-n` *role2* . . . (Optional) Specifies the role(s) that you want to list. If you do not specify a role name, all roles are listed.

- For subcommand `modify`:

  `-a` *adduser1* `-a` *adduser2* . . . (Optional) Specifies the user name(s) to add to the new role. The administrator must have the `solaris.role.assign` authorization, or must have the `solaris.role.delegate` authorization and be a member of the role being modified.

  `-c` *comment* (Optional) Includes a short description of the role. Consists of a string of up to 256 printable characters, excluding the colon (`:`).

  `-d` *dir* (Optional) Specifies the home directory of the new role, limited to 1024 characters.

  `-F` *full_name* (Optional) Specifies the full, descriptive name of the role. The *full_name* must be unique within a domain, and can contain alphanumeric characters and spaces. If you use spaces, you must enclose the *full_name* in double quotes.

  `-G` *group1* `-G` *group2* . . . (Optional) Specifies the new role's secondary group membership in the system group database with the character string names of one or more existing groups. *Note:* You cannot assign a primary group to a role. A role's primary group is always `sysadmin` (group 14).

  `-h` (Optional) Displays the command's usage statement.

`-n` *rolename*	Specifies the name of the role you want to modify.
`-N` *new_rolename*	(Optional) Specifies the new name of the role.
`-p` *addprof1* `-p` *addprof2* . . .	(Optional) Specifies the profile(s) to add to the role. To assign a profile to a role, the administrator must have the `solaris.profmgr.assign` or `solaris.profmgr.delegate` authorization.
`-P` *password*	(Optional) Specifies the role's password. The *password* can contain up to eight characters. To set the password, the administrator must have the `solaris.admin.usermgr.pswd` authorization. *Note:* When you specify a password, you type the password in plain text. Specifying a password using this method introduces a security gap while the command is running.
`-q` *delprof1* `-q` *delprof2* . . .	(Optional) Specifies the profile(s) to delete from the role.
`-r` *deluser1* `-r` *deluser2* . . .	(Optional) Specifies the user name(s) to delete from the role.
`-s` *shell*	(Optional) Specifies the full pathname of the program used as the role's shell on login. Valid entries are `/bin/pfcsh` (C shell), `/bin/pfksh` (Korn shell), and `/bin/pfsh` (Bourne shell), the default.
`-x` *autohome*=Y|N	(Optional) Sets the role's home directory. The home directory path in the password entry is set to `/home/`*login_name*.
`-x` *perm=home_perm*	(Optional) Sets the permissions on the role's home directory. *perm* is interpreted as an octal number, and the default is 0775.

EXAMPLES

EXAMPLE 1 Creating a role account

The following creates the `role1` account with a full name of `Engineering Admin` and a password of `abc123` on the local file system, and assigns `user1` and `user2` to the role. This role has `Name Service Security` and `Audit Review` rights. The system assigns the next available unique UID greater than 100.

```
./smrole add -H myhost -p mypasswd -u root -- -n role1 \
         -F "Engineering Admin" -P abc123 -a user1 -a user2 \
         -p "Name Service Security" -p "Audit Review"
```

EXAMPLE 2 Deleting role accounts

The following deletes the `role1` and `role2` accounts from the local file system.

```
./smrole delete -H myhost -p mypasswd -u root -- -n role1 -n role2
```

EXAMPLE 3 Listing role accounts

The following lists all role accounts on the local file system in summary form.

```
./smrole list -H myhost -p mypasswd -u root --
```

EXAMPLE 4 Modifying a role account

The following modifies the `role1` account so the role defaults to the Korn shell, includes the `user3` account, and does not include the `user2` account.

```
./smrole modify -H myhost -p mypasswd -u root -- -n role1 \
          -s /bin/pfksh  -a user3 -r user2
```

ENVIRONMENT VARIABLES

See `environ`(5) for a description of the `JAVA_HOME` environment variable, which affects the execution of the `smrole` command. If this environment variable is not specified, the `/usr/java` location is used. See `smc`(1M).

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` Invalid command syntax. A usage message displays.

`2` An error occurred while executing the command. An error message displays.

FILES

The following files are used by the `smrole` command:

`/etc/aliases`	Mail aliases. See `aliases`(4).
`/etc/auto_home`	Automatic mount points. See `automount`(1M).
`/etc/group`	Group file. See `group`(4).
`/etc/passwd`	Password file. See `passwd`(4).
`/etc/security/policy.conf`	Configuration file for security policy. See `policy.conf`(4).
`/etc/shadow`	Shadow password file. See `shadow`(4).
`/etc/user_attr`	Extended user attribute database. See `user_attr`(4).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmga

SEE ALSO automount(1M), smc(1M), aliases(4), group(4), passwd(4), policy.conf(4), shadow(4), user_attr(4), attributes(5), environ(5)

NAME

smrsh – restricted shell for sendmail

SYNOPSIS

smrsh -c *command*

DESCRIPTION

The smrsh program is intended as a replacement for the sh command in the prog mailer in sendmail(1M) configuration files. The smrsh program sharply limits commands that can be run using the |program syntax of sendmail. This improves overall system security. smrsh limits the set of programs that a programmer can execute, even if sendmail runs a program without going through an alias or forward file.

Briefly, smrsh limits programs to be in the directory /var/adm/sm.bin, allowing system administrators to choose the set of acceptable commands. It also rejects any commands with the characters: ,, <, >, |, ;, &, $, \r (RETURN), or \n (NEWLINE) on the command line to prevent end run attacks.

Initial pathnames on programs are stripped, so forwarding to /usr/ucb/vacation, /usr/bin/vacation, /home/server/mydir/bin/vacation, and vacation all actually forward to/var/adm/sm.bin/vacation.

System administrators should be conservative about populating /var/adm/sm.bin. Reasonable additions are utilities such as vacation(1) and procmail. Never include any shell or shell-like program (for example, perl) in the sm.bin directory. This does not restrict the use of shell or perl scrips in the sm.bin directory (using the #! syntax); it simply disallows the execution of arbitrary programs.

OPTIONS

The following options are supported:

-c *command* Where *command* is a valid command, executes *command*.

FILES

/var/adm/sm.bin directory for restricted programs

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsr, SUNWcsu

SEE ALSO

sendmail(1M), , attributes(5)

NAME

smserialport – manage serial port

SYNOPSIS

/usr/sadm/bin/smserialport *subcommand* [*auth_args*] -- [*subcommand_args*]

DESCRIPTION

The `smserialport` command manages serial ports.

Sub-commands

The following `smserialport` sub-commands (*subcommand*) are supported:

configure	Configures a serial port's basic settings for a device such as a terminal, modem or no connection.
delete	Deletes a given port. You can disable a port and prevent new services from being spawned for incoming connections, without interfering with existing services.
list	Lists all serial ports.
modify	Modifies a serial port's parameters.

OPTIONS

There are two kinds of options: authentication arguments (*args*) and sub-command arguments (*subcommand_args*).

Authentication Arguments

The `smserialport` authentication arguments, *args*, are derived from the `smc`(1M) argument set and are the same regardless of which sub-command you use.

Valid *args* are `-D`, `-H`, `-l`, `-p`, `-r`, and `-u`; they are all optional. If no *args* are specified, certain defaults will be assumed and the user may be prompted for additional information, such as a password for authentication purposes.

The single letter options can also be specified by their equivalent option words preceded by a double dash. For example, you can use either `-D` or `--domain`.

Descriptions and other arg options that contain white spaces must be enclosed in double quotes.

The following authentification arguments (*args*) are supported:

`-D | --domain` *domain*
Specifies the default domain that you want to manage. `smserialport` accepts only the *file* value for this option. *file* is also the default value.

The *file* default domain means that changes are local to the server. Toolboxes can change the domain on a tool-by-tool basis; this option specifies the *domain* for all other tools.

`-H | --hostname` *host_name*:*port*
Specifies the host and port to which you want to connect. If you do not specify a port, the system connects to the default port, `898`. If you do not specify a host (*host_name*:*port*, the Solaris Management Console connects to the local host on port `898`. You may still have to choose a toolbox to load into the console. To override this behavior, use the `smc -B` option, or set your console preferences to load a `home toolbox` by default.

`-l | --rolepassword` *role_password*
Specifies the password for the *role_name*. If you specify a *role_name* but do not specify a *role_password*, the system prompts you to supply a *role_password*. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-p | --password` *password*
Specifies the password for the *user_name*. If you do not specify a password, the system prompts you for one. Because passwords specified on the command line can be seen by any user on the system, this option is considered insecure.

`-r | --rolename` *role_name*
Specifies a role name for authentication. If you do not specify this option, no role is assumed.

`-u | --username` *user_name*
Specifies the user name for authentication. If you do not specify this option, the user identity running the console process is assumed.

`--`
This option is required and must always follow the preceding options. If you do not enter the preceding options, you must still enter the `--` option.

Sub-command Arguments

The sub-command specific options, *subcommand_args*, must come after the *args* and must be separated from them by the `--` option. Enclose descriptions and *arg* options that contain white space in double quotes.

configure

The `configure` sub-command requires the following sub-command argument:

`-n` *port_name* Specifies the name of the serial port to reconfigure.

The following sub-command arguments are optional for the `configure` sub-command:

`-b` *baudrate*
Specifies the port baud rate. The supported baud rate are `38400`, `19200`, `9600`, `4800`, `2400`, `1200`, `300` and `auto`. The default is `9600`.

`-c` *comment*
Specifies a short comment description of the service. The default is a description of the requested device type.

`-h`
Displays the command's usage statement.

`-l` *login_prompt*
Specifies the login prompt. The default is *tty*`'`*port_name*`' login:`.

`-t` *terminal_type*
Specifies the terminal type. The default is `tvi925`.

`-x` *device_name*
Specifies the device to be configured. Valid device_names are: `terminal`, `modemdialin`, `modemdialout`, `modemdialinout` or `initializeonly` for no connection. The default is `terminal`.

`-x` service=`y` | `n`
Specifies the status of service, that is `y` for enabled or `n` for disabled. The default is `y`.

delete

The `delete` sub-command requires the following sub-command arguments:

`-n` *port_name*
Specifies the name of the serial port to be disabled.

The following sub-command arguments are optional for the `delete` sub-command:

`-h`
Displays the command's usage statement.

list

The `list` sub-command does not require any sub-command arguments.

The following sub-command arguments are optional for the `list` sub-command:

`-h`
Displays the command's usage statement.

`-v`
Displays the data in verbose format.

modify

The `modify` sub-command requires the following sub-command arguments:

`-n` *port_name*
Specifies the name of the serial port to modify.

The following sub-command arguments are optional for the `modify` sub-command:

`-b` *baudrate*
Specifies the port baud rate. The supported baud rate are `38400`, `19200`, `9600`, `4800`, `2400`, `1200`, `300` and `auto`.

`-c` *comment*
A short comment description of the service.

`-h`
Displays the command usage statement.

`-l` *login_prompt*
Specifies the login prompt.

`-t` *terminal_type*
Specifies the terminal type.

`-x` bidirectional=`y` | `n`
Specifies the bi-directional port flag, `y` for set or `n` for not set. When this flag is set, the line can be used in both directions.

`-x` connect_on_carrier=`y` | `n`
Specifies if to connect on carrier, that is `y` or `n`.

`-x` initialize_only=`y` | `n`
Specifies if the service invocation. If `y` the service is invoked only once. This can be used to configure a particular device without actually monitoring it, as with software carrier.

`-x` service_program=*command*
Specifies the full pathname of the service command to invoke when a connection request is received.

`-x` service_status=`y` | `n`
Specifies the status of service, that is `y` for enabled or `n` for disabled.

`-x` software_carrier=`y` | `n`
Specifies the carrier detection. `y` for software or `n` for hardware.

`-x` timeout=*timeout*
Specifies the time to close a port if the open on the port succeeds, and no input data is received in timeout seconds. The supported *timeout* are `never`, `30`, `60` and `90`.

EXAMPLES

EXAMPLE 1 Listing Serial Ports

The following example lists the serial ports:

```
example% ./smserialport list -H myhost -u root -p mypassword --

Port Service  Baud-Rate Terminal-Type Prompt       Comment
a    enabled  9600      xterm         as           welcome
b    enabled  9600      tvi925        ttyb login:
```

EXAMPLE 2 Modifying Serial Ports

The following example contains two commands. The first command modifies serial port `b` for a baud rate of `4800`, an `xterm` as terminal type, a `b:` for login prompt and a comment. The second command lists the ports.

```
example% ./smserialport modify -H myhost -u root -p mypassword -- \
     -n b -b 4800 -t xterm -l b: -c "modified port b"
```

EXAMPLE 2 Modifying Serial Ports *(Continued)*

```
example% ./smserialport list -H myhost -u root -p mypassword --

Port Service  Baud-Rate Terminal-Type Prompt      Comment
a    enabled  9600      xterm         as          welcome
b    enabled  4800      xterm         b:          modified port b
```

EXAMPLE 3 Deleting a Serial Port

The following example contains two commands. The first command deletes serial port b. The second command lists the ports.

```
example% ./smserialport delete -H myhost -u root \
     -p mypassword -- -n b

example% ./smserialport list -H myhost -u root -p mypassword --

Port Service  Baud-Rate Terminal-Type Prompt      Comment
a    enabled  9600      xterm         as          welcome
b    disabled 9600      tvi925        ttyb login:
```

EXAMPLE 4 Configuring a Serial Port

The following example contains two commands. The first command configures serial port b for a bi-directional modem. The second command lists the ports.

```
example% ./smserialport configure -H myhost -u root  \
     -p mypassword -- -n b -x device=modemdialinout

example% ./smserialport list -H myhost -u root -p mypassword --
Port Service  Baud-Rate Terminal-Type Prompt      Comment
a    enabled  9600      xterm         as          welcome
b    enabled  9600      tvi925        ttyb login: Modem - Dial In and Out
```

ENVIRONMENT VARIABLES

See environ(5) for descriptions of the following environment variables that affect the execution of smserialport: JAVA_HOME. If this environment variable is not specified, the /usr/java location is used. See smc(1M).

EXIT STATUS

The following exit values are returned:

0 Successful completion.

1 Invalid command syntax. A usage message displays.

2 An error occurred while executing the command. An error message displays.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmga

SEE ALSO `tip`(1), `pmadm`(1M), `sacadm`(1M), `smc`(1M), `ttyadm`(1M), `ttymon`(1M), `attributes`(5), `environ`(5)

NAME

smuser – manage user entries

SYNOPSIS

/usr/sadm/bin/smuser *subcommand* [*auth_args*] - – [*subcommand_args*]

DESCRIPTION

The `smuser` command manages one or more user entries in the local `/etc` filesystem or a NIS or NIS+ target name service.

subcommands

`smuser` *subcommand*s are:

`add` Adds a new user entry to the appropriate files. You can use a template and input file instead of supplying the additional command line options. If you use a template and command line options, the command line options take precedence and override any conflicting template values. To add an entry, the administrator must have the `solaris.admin.usermgr.write` authorization.

`delete` Deletes one or more user entries from the appropriate files. To delete an entry, the administrator must have the `solaris.admin.usermgr.write` authorization. *Note:* You cannot delete the system accounts with IDs less than 100, or 60001, 60002, or 65534.

`list` Lists one more user entries from the appropriate files. To list entries, the administrator must have the `solaris.admin.usermgr.read` authorization.

`modify` Modifies a user entry in the appropriate files. To modify an entry, the administrator must have the `solaris.admin.usermgr.write` authorization.

OPTIONS

The `smuser` authentication arguments, *auth_args*, are derived from the `smc`(1M) arg set and are the same regardless of which subcommand you use. The `smuser` command requires the Solaris Management Console to be initialized for the command to succeed (see `smc`(1M)). After rebooting the Solaris Management Console server, the first Solaris Management Console connection might time out, so you might need to retry the command.

The subcommand-specific options, *subcommand_args*, must come after the *auth_args* and must be separated from them by the `- –` option.

auth_args

The valid *auth_args* are `-D`, `-H`, `-l`, `-p`, `-r`, and `-u`; they are all optional. If no *auth_args* are specified, certain defaults will be assumed and the user may be prompted for additional information, such as a password for authentication purposes. These letter options can also be specified by their equivalent option words preceded by a double dash. For example, you can use either `-D` or `- –domain` with the *domain* argument.

`-D | - –domain` *domain*
Specifies the default domain that you want to manage. The syntax of *domain* is *type*:/*host_name*/*domain_name*, where *type* is `nis`, `nisplus`, `dns`, `ldap`, or `file`; *host_name* is the name of the machine that serves the domain; and *domain_name* is the name of the domain you want to manage. (*Note:* Do *not* use `nis+` for `nisplus`.)

If you do not specify this option, the Solaris Management Console assumes the `file` default domain on whatever server you choose to manage, meaning that changes are local to the server. Toolboxes can change the domain on a tool-by-tool basis; this option specifies the domain for all other tools.

`-H | - –hostname` *host_name:port*
Specifies the *host_name* and *port* to which you want to connect. If you do not specify a *port*, the system connects to the default port, `898`. If you do not specify *host_name:port*, the Solaris Management Console connects to the local host on port `898`. You may still have to choose a toolbox to load into the console. To override this behavior, use the `smc`(1M) `-B` option, or set your console preferences to load a "home toolbox" by default.

`-l | - –rolepassword` *role_password*
Specifies the password for the *role_name*. If you specify a *role_name* but do not specify a *role_password*, the system prompts you to supply a *role_password*. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-p | - –password` *password*
Specifies the password for the *user_name*. If you do not specify a password, the system prompts you for one. Passwords specified on the command line can be seen by any user on the system, hence this option is considered insecure.

`-r | - –rolename` *role_name*
Specifies a role name for authentication. If you do not specify this option, no role is assumed.

`-u | - –username` *user_name*
Specifies the user name for authentication. If you do not specify this option, the user identity running the console process is assumed.

`- –`
This option is required and must always follow the preceding options. If you do not enter the preceding options, you must still enter the `- –` option.

subcommand_args

Note: Descriptions and other arg options that contain white spaces must be enclosed in double quotes.

- For subcommand `add`:

`-c` *comment*	(Optional) Includes a short description of the login, which is typically the user's name. Consists of a string of up to 256 printable characters, excluding the colon (:).
`-d` *dir*	(Optional) Specifies the home directory of the new user, limited to 1024 characters.
`-e` *ddmmyyyy*	(Optional) Specifies the expiration date for a login. After this date, no user can access this login. This option is useful for creating temporary logins. Specify a null value (" ") to indicate that the login is always

valid. The administrator must have the `solaris.admin.usermgr.pswd` authorization.

`-f` *inactive*
: (Optional) Specifies the maximum number of days allowed between uses of a login ID before that ID is declared invalid. Normal values are positive integers. Enter zero to indicate that the login account is always active.

`-F` *full_name*
: (Optional) Specifies the full, descriptive name of the user. The *full_name* must be unique within a domain and can contain alphanumeric characters and spaces. If you use spaces, you must enclose the *full_name* in double quotes.

`-g` *group*
: (Optional) Specifies the new user's primary group membership in the system group database with an existing group's integer ID.

`-G` *group1* `-G` *group2* . . .
: (Optional) Specifies the new user's supplementary group membership in the system group database with the character string names of one or more existing groups. Duplicates of groups specified with the `-g` and `-G` options are ignored.

`-h`
: (Optional) Displays the command's usage statement.

`-n` *login*
: Specifies the new user's login name. The login name must be unique within a domain, contain 2–32 alphanumeric characters, begin with a letter, and contain at least one lowercase letter.

`-P` *password*
: (Optional) Specifies up to an eight-character password assigned to the user account. *Note:* When you specify a password, you type the password in plain text. Specifying a password using this method introduces a security gap while the command is running. To set the password, the administrator must have the `solaris.admin.usermgr.pswd` authorization.

`-s` *shell*
: (Optional) Specifies the full pathname (limited to 1024 characters) of the program used as the user's shell on login. Valid entries are a user-defined shell, `/bin/csh` (C shell), `bin/ksh` (Korn shell), and the default, `/bin/sh` (Bourne shell).

`-t` *template*
: (Optional) Specifies a template, created using the User Manager tool, that contains a set of pre-defined user attributes. You may have entered a name service server in the template. However, when a user is actually added with this template, if a name service is

unavailable, the user's local server will be used for both the Home Directory Server and Mail Server.

`-u` *uid* — (Optional) Specifies the user ID of the user you want to add. If you do not specify this option, the system assigns the next available unique user ID greater than 100.

`-x` *autohome*=Y|N — (Optional) Sets the home directory to automount if set to `Y`. The user's home directory path in the password entry is set to `/home/`*login name*.

`-x` *mail=mail_server* — (Optional) Specifies the host name of the user's mail server, and creates a mail file on the server. Users created in a local scope must have a mail server created on their local machines.

`-x` *perm=home_perm* — (Optional) Sets the permissions on the user's home directory. *perm* is interpreted as an octal number, and the default is `0775`.

`-x` *pwmax=days* — (Optional) Specifies the maximum number of days that the user's password is valid. The administrator must have the `solaris.admin.usermgr.pswd` authorization.

`-x` *pwmin=days* — (Optional) Specifies the minimum number of days between user password changes. The administrator must have the `solaris.admin.usermgr.pswd` authorization.

`-x` *pwwarn=days* — (Optional) Specifies the number of days relative to *pwmax* that the user is warned about password expiration prior to the password expiring. The administrator must have the `solaris.admin.usermgr.pswd` authorization.

`-x` *serv=homedir_server* — (Optional) Specifies the name of the server where the user's home directory resides. Users created in a local scope must have their home directory server created on their local machines.

- For subcommand `delete`:

  `-h` — (Optional) Displays the command's usage statement.

  `-n` *login1* — Specifies the login name of the user you want to delete.

  `-n` *login2* . . . — (Optional) Specifies the additional login name(s) of the user(s) you want to delete.

- For subcommand `list`:

  `-h` — (Optional) Displays the command's usage statement.

`-l` Displays the output for each user in a block of *key:value* pairs (for example, `user name:root`) followed by a blank line to delimit each user block. Each *key:value* pair is displayed on a separate line. The keys are: `autohome setup`, `comment`, `days to warn`, `full name`,`home directory`, `home directory permissions`, `login shell`, `mail server`, `max days change`, `max days inactive`, `min days change`, `password expires`, `password type`, `primary group`, `rights`, `roles`, `secondary groups`, `server`, `user ID (UID)`, and `user name`.

`-n` *login1* Specifies the login name of the user you want to list.

`-n` *login2* . . . (Optional) Specifies the additional login name(s) of the user(s) you want to list.

- For subcommand `modify`:

`-a` *addrole1* `-a` *addrole2* . . . (Optional) Specifies the role(s) to add to the user account. To assign a role to a user, the administrator must have the `solaris.role.assign` authorization or must have the `solaris.role.delegate` authorization and be a member of each of the roles specified.

`-c` *comment* (Optional) Describes the changes you made to the user account. Consists of a string of up to 256 printable characters, excluding the colon (:).

`-d` *description* (Optional) Specifies the user's home directory, limited to 1024 characters.

`-e` *ddmmyyyy* (Optional) Specifies the expiration date for a login in a format appropriate to the locale. After this date, no user can access this login. This option is useful for creating temporary logins. Specify a null value (" ") to indicate that the login is always valid.

`-f` *inactive* (Optional) Specifies the maximum number of days allowed between uses of a login ID before the ID is declared invalid. Normal values are positive integers. Specify zero to indicate that the login account is always active.

`-F` *full_name* (Optional) Specifies the full, descriptive name of the user. The *full_name* must be unique within a domain and can contain alphanumeric characters and spaces. If you use spaces, you must enclose the *full_name* in double quotes.

`-g` *group* (Optional) Specifies the new user's primary group membership in the system group database with an existing group's integer ID.

`-G` *group1* `-G` *group2* . . .	(Optional) Specifies the new user's supplementary group membership in the system group database with the character string names of one or more existing groups. Duplicates of groups specified with the `-g` and `-G` options are ignored.
`-h`	(Optional) Displays the command's usage statement.
`-n` *name*	Specifies the user's current login name.
`-N` *new_name*	(Optional) Specifies the user's new login name. The login name must be unique within a domain, contain 2–32 alphanumeric characters, begin with a letter, and contain at least one lowercase letter.
`-p` *addprof1* `-p` *addprof2* . . .	(Optional) Specifies the profile(s) to add to the user account. To assign a profile to a user, the administrator must have the `solaris.profmgr.assign` or `solaris.profmgr.delegate` authorization.
`-P` *password*	(Optional) Specifies up to an eight-character password assigned to the user account. When you specify a password, you type the password in plain text. Specifying a password using this method introduces a security gap while the command is running.
`-q` *delprof1* `-q` *delprof2* . . .	(Optional) Specifies the profile(s) to delete from the user account.
`-r` *delrole1* `-r` *delrole2* . . .	(Optional) Specifies the role(s) to delete from the user account.
`-s` *shell*	(Optional) Specifies the full pathname (limited to 1024 characters) of the program used as the user's shell on login. Valid entries are a user-defined shell, `/bin/csh` (C shell), `bin/ksh` (Korn shell), and the default, `/bin/sh` (Bourne shell).l)
`-x` *autohome*=Y|N	(Optional) Sets up the home directory to automount if set to `Y`. The user's home directory path in the password entry is set to `/home/`*login name*.
`-x` *pwmax=days*	(Optional) Specifies the maximum number of days that the user's password is valid.
`-x` *pwmin=days*	(Optional) Specifies the minimum number of days between password changes.
`-x` *pwwarn=days*	(Optional) Specifies the number of days relative to *pwmax* that the user is warned about password

expiration before the password expires.

EXAMPLES

EXAMPLE 1 Creating a new user account

The following creates a new user account on the local file system. The account name is `user1`, and the full name is `Joe Smith`. The comment field verifies that the account is for Joe Smith. The system will assign the next available user ID greater than 100 to this account. There is no password set for this account, so when Joe Smith logs in for the first time, he will be prompted to enter a password.

```
./smuser add -H myhost -p mypasswd -u root -- -F "Joe Smith" \
             -n user1 -c "Joe's account"
```

EXAMPLE 2 Deleting a user account

The following deletes the `user1` account from the local file system:

```
./smuser delete -H myhost -p mypasswd -u root -- -n user1
```

EXAMPLE 3 Listing all user accounts

The following lists all user accounts on the local file system in summary form:

```
./smuser list -H myhost -p mypasswd -u root --
```

EXAMPLE 4 Modifying a user account

The following modifies the `user1` account to default to a Korn shell, and assigns the account to the `qa_group` secondary group.

```
./smuser modify -H myhost -p mypasswd -u root -- -n user1 \
             -s /bin/ksh -G qa_group
```

ENVIRONMENT VARIABLES

See `environ`(5) for a description of the `JAVA_HOME` environment variable, which affects the execution of the `smuser` command. If this environment variable is not specified, the `/usr/java` location is used. See `smc`(1M).

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` Invalid command syntax. A usage message displays.

`2` An error occurred while executing the command. An error message displays.

FILES

The following files are used by the `smuser` command:

`/etc/aliases` Mail aliases. See `aliases`(4).

/etc/auto_home	Automatic mount points. See automount(1M).
/etc/group	Group file. See group(4).
/etc/passwd	Password file. See passwd(4).
/etc/security/policy.conf	Configuration file for security policy. See policy.conf(4).
/etc/shadow	Shadow password file. See shadow(4).
/etc/user_attr	Extended user attribute database. See user_attr(4).

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWmga

SEE ALSO

automount(1M), smc(1M), aliases(4), group(4), passwd(4), policy.conf(4), shadow(4), user_attr(4), attributes(5), environ(5)

NAME

snmpdx – Sun Solstice Enterprise Master Agent

SYNOPSIS

```
/usr/lib/snmp/snmpdx [-hy] [-a filename] [-c config-dir] [-d debug-level]
     [-i filename] [-m GROUP -m SPLIT] [-o filename] [-p port]
     [-r filename]
```

DESCRIPTION

The Master Agent, `snmpdx`, is the main component of Solstice Enterprise Agent technology. It runs as a daemon process and listens to User Datagram Protocol (UDP) port 161 for SNMP requests. The Master Agent also opens another port to receive SNMP trap notifications from various subagents. These traps are forwarded to various managers, as determined by the configuration file.

Upon invocation, `snmpdx` reads its various configuration files and takes appropriate actions by activating subagents, determining the subtree Object Identifier (OID) for various subagents, populating its own Management Information Bases (MIBs), and so forth. The Master Agent invokes subagents, registers subagents, sends requests to subagents, receives responses from subagents, and traps notifications from subagents.

The Master Agent is invoked from a start-up script at boot time only if contents of the resource configuration file `/var/snmp/conf/snmpdx.rsrc` are non-trivial.

OPTIONS

The following options are supported:

`-a` *filename*
: Specify the full path of the access control file used by the Master Agent. The default access control file is `/etc/snmp/conf/snmpdx.acl`.

`-c` *config-dir*
: Specify the full path of the directory containing the Master Agent configuration files. The default directory is `/etc/snmp/conf`.

`-d` *debug-level*
: Debug. Levels from 0 to 4 are supported, giving various levels of debug information. The default is 0 which means no debug information is given.

`-h`
: Help. Print the command line usage.

`-i` *filename*
: Specify the full path of the enterprise-name OID map. This file contains the PID used by the Master Agent for recovery after a crash. It contains tuples of the UNIX process ID, port number, resource name, and agent name. The default file is `/var/snmp/snmpdx.st`.

`-m GROUP` | `-m SPLIT`
: Specify the mode to use for forwarding of SNMP requests.

  `GROUP`
  : Multiple variables can be included in each request from the Master Agent to the subagents. This results in, at mose, one send-request per agent.

	`SPLIT` Each variable in the incoming request results in one send-request to each subagent.
	The default is `GROUP`.
`-o` *filename*	Specify the full path of the file containing the tuple (enterprise-name, OID). For example, (Sun Microsystems, 1.3.1.6.1.4.32). The Master Agent uses this file as a base for look-up in the trap-filtering and forwarding process. The default file is `/etc/snmp/conf/enterprises.oid`.
`-p` *port*	Specify the port number. The default port number is 161.
`-r` *filename*	Specify the full path of the resource file to be used by the Master Agent. This file stores information about the subagents that the Master Agent invokes and manages. The default resource file is `/etc/snmp/conf/snmpdx.rsrc`.
`-y`	Set a recovery indicator to invoke the recovery module. The recovery process discovers which subagents in the previous session are still active; those subagents not active are re-spawned by the Master Agent.

FILES

`/var/snmp/conf/enterprises.oid`
Enterprise-name OID map

`/var/snmp/conf/snmpdx.acl`
Access control file

`/var/snmp/conf/snmpdx.rsrc`
Resource configuration file

`/var/snmp/snmpdx.st`
Master Agent status file

`/var/snmp/mib/snmpdx.mib`
Master Agent MIB file

EXIT STATUS

The following error values are returned:

`0`	Successful completion.
non-zero	An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsasnm

SEE ALSO `snmpXdmid`(1M), `attributes`(5)

NAME

snmpXdmid – Sun Solstice Enterprise SNMP-DMI mapper subagent

SYNOPSIS

/usr/lib/dmi/snmpXdmid `-s` *hostname* `[-h] [-c` *config-dir*`] [-d` *debug-level*`]`

DESCRIPTION

The `snmpXdmid` utility is a subagent in the Solstice Enterprise Agent Desktop Management Interface package. It maps the SNMP requests forwarded by the Master Agent (`snmpdx`(1M)) into one or more equivalent DMI requests. Further, it remaps the DMI response into SNMP response back to `snmpdx`. By default, `snmpXdmid` also forwards the DMI indications as SNMP traps to `snmpdx`. The feature is configurable and can be disabled by setting `TRAP_FORWARD_TO_MAGENT=0` in the `snmpXdmid` configuration file, `snmpXdmid.conf`.

This subagent runs as a daemon in the system. The subagent uses a set of .MAP files located in `/var/dmi/map` to map the SNMP Object Identifier (OID) into a corresponding DMI component. The map files are generated using the MIF-to-MIB utility, `miftomib`. They are read by `snmpXdmid` when a corresponding MIF file gets registered with the DMI Service Provider (`dmispd`(1M)).

The `snmpXdmid.conf` file is used for configuration information. Each entry in the file consists of a keyword followed by an equal sign (=), followed by a parameter string. The keyword must begin in the first position. A line beginning with a pound sign (#) is treated as a comment and the subsequent characters on that line are ignored. The keywords currently supported are:

`WARNING_TIMESTAMP`	Indication subscription expiration, warning time.	
`EXPIRATION_TIMESTAMP`	Indication subscription expiration timestamp.	
`FAILURE_THRESHOLD`	DMISP retries before dropping indication due to comm errors.	
`TRAP_FORWARD_TO_MAGENT`		
	`0`	Drop indication at the subagent level.
	non-zero	Forward indications as SNMP traps to `snmpdx`.

By default, the configuration file `snmpXdmid.conf` is located in the `/etc/dmi/conf` directory. You can specify an alternative directory with the `-c` option.

OPTIONS

The following options are supported:

`-c` *config-dir*	Specify the directory where `snmpXdmid.conf` file is located.
`-d` *debug-level*	Debug. Levels from 1 to 5 are supported, giving various levels of debug information.
`-h`	Help. Print the command line usage.
`-s` *hostname*	Specify the host on which `dmispd` is running.

FILES

`/etc/dmi/conf/snmpXdmid.conf` DMI mapper configuration file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsadmi

SEE ALSO

`dmispd`(1M), `snmpdx`(1M), `attributes`(5)

NAME

snmpXwbemd – SNMP Adapter Subagent for WBEM

SYNOPSIS

/usr/sadm/lib/wbem/snmpXwbemd [-d] [-h] [-p *port*]

DESCRIPTION

The snmpXwbemd daemon is a subagent in the Web-Based Enterprise Management (WBEM) services package.

This daemon maps the Simple Network Management Protocol (SNMP) requests forwarded by the Solstice Enterprise Agents (SEA) Master Agent snmpdx(1M) into one or more equivalent WBEM Common Information Model (CIM) properties and instances. Further, it remaps the response from the CIM Object Manager into a SNMP response, which it passes back to snmpdx(1M).

A mapping file contains the corresponding Object Identifier (OID), class name, property name, and Abstract Syntax Notation 1 (ASN.1) type for each object. You can also create your own mapping file.

OPTIONS

The following options are supported:

-d Displays all debug information.

-h Displays help by printing the correct command line usage.

-p Specifies the port number to use.

OPERANDS

The following operand is supported:

port Specifies the port number you want to use.

EXAMPLES

EXAMPLE 1 An Example of a 050SUNWwbcou.map File

This mapping file that Sun Microsystems provides contains definitions of objects, in this format:

```
#
#pragma ident    "@(#)050SUNWwbcou.map    1.0    01/04/03 SMI"
#
# Copyright (c) 2001 by Sun Microsystems, Inc.
# All rights reserved.
#
# *** Description of contents ***
#
# First non-commented non-blank line contains required Version label.
# Remaining non-commented non-blank lines are considered map entries
# used as described below:
#
# Column 1 - SNMP OID - Uniquely describes an SNMP variable
# Column 2 - CIM Class Name - CIM class associated with this variable
# Column 3 - CIM Property Name - CIM property that maps to SNMP OID variable
# Column 4 - ASN.1 type - SNMP datatype that dictates how data is mapped
#            to/from SNMP requests.  Supported types are: SnmpString, SnmpOid,
#            SnmpTimeticks, SnmpCounter, SnmpInt, SnmpGauge, SnmpIpAddress,
#            SnmpOpaque)
# Column 5 and greater are ignored
#
```

EXAMPLE 1 An Example of a `050SUNWwbcou.map` File *(Continued)*

```
Version 1.0

1.3.6.1.2.1.1.1.0 Solaris_ComputerSystem Description SnmpString
1.3.6.1.2.1.1.3.0 Solaris_OperatingSystem LastBootUpTime SnmpTimeticks
1.3.6.1.2.1.1.4.0 Solaris_ComputerSystem PrimaryOwnerContact SnmpString
1.3.6.1.2.1.1.5.0 Solaris_ComputerSystem Name SnmpString

1.3.6.1.2.1.25.1.5.0 Solaris_OperatingSystem NumberOfUsers SnmpGauge
1.3.6.1.2.1.25.1.6.0 Solaris_OperatingSystem NumberOfProcesses SnmpGauge
1.3.6.1.2.1.25.1.7.0 Solaris_OperatingSystem MaxNumberOfProcesses SnmpGauge
1.3.6.1.2.1.25.1.2.0 Solaris_OperatingSystem LocalDateTime SnmpString
```

Each definition of an object in this file contains an OID, its corresponding CIM class name, its corresponding CIM property name, and its corresponding ASN.1 type. Each of these elements is separated by a space character.

FILES

`/var/sadm/wbem/snmp/map/050SUNWwbcou.map`
The SNMP Adapter Subagent for WBEM MIB–2 mapping file that Sun Microsystems provides contains SNMP Management Information Base (MIB) definitions for the CIM instrumentation that SNMP manages.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes.

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWwbcou
CSI	Enabled
Interface Stability	Evolving
MT-Level	Safe

SEE ALSO

`snmpdx`(1M), `attributes`(5)

NAME

snoop – capture and inspect network packets

SYNOPSIS

```
snoop [-aqrCDNPSvV] [-t [r | a | d]] [-c maxcount] [-d device]
     [-i filename] [-n filename] [-o filename] [-p first [, last]] [-s snaplen]
     [-x offset [, length]] [expression]
```

DESCRIPTION

`snoop` captures packets from the network and displays their contents. `snoop` uses both the network packet filter and streams buffer modules to provide efficient capture of packets from the network. Captured packets can be displayed as they are received, or saved to a file (which is *RFC 1761*–compliant) for later inspection.

`snoop` can display packets in a single-line summary form or in verbose multi-line forms. In summary form, only the data pertaining to the highest level protocol is displayed. For example, an NFS packet will have only NFS information displayed. The underlying RPC, UDP, IP, and ethernet frame information is suppressed but can be displayed if either of the verbose options are chosen.

`snoop` requires an interactive interface.

OPTIONS

`-C` List the code generated from the filter expression for either the kernel packet filter, or `snoop`'s own filter.

`-D` Display number of packets dropped during capture on the summary line.

`-N` Create an IP address-to-name file from a capture file. This must be set together with the `-i` option that names a capture file. The address-to-name file has the same name as the capture file with `.names` appended. This file records the IP address to hostname mapping at the capture site and increases the portability of the capture file. Generate a `.names` file if the capture file is to be analyzed elsewhere. Packets are not displayed when this flag is used.

`-P` Capture packets in non-promiscuous mode. Only broadcast, multicast, or packets addressed to the host machine will be seen.

`-S` Display size of the entire ethernet frame in bytes on the summary line.

`-V` Verbose summary mode. This is halfway between summary mode and verbose mode in degree of verbosity. Instead of displaying just the summary line for the highest level protocol in a packet, it displays a summary line for each protocol layer in the packet. For instance, for an NFS packet it will display a line each for the ETHER, IP, UDP, RPC and NFS layers. Verbose summary mode output may be easily piped through `grep` to extract packets of interest. For example, to

view only RPC summary lines, enter the following:
`example# snoop -i rpc.cap -V | grep RPC`

`-a` — Listen to packets on `/dev/audio` (warning: can be noisy).

`-c` *maxcount* — Quit after capturing *maxcount* packets. Otherwise keep capturing until there is no disk left or until interrupted with Control-C.

`-d` *device* — Receive packets from the network using the interface specified by *device*, for example, `le0` or `hme0`. The program `netstat`(1M), when invoked with the `-i` flag, lists all the interfaces that a machine has. Normally, `snoop` will automatically choose the first non-loopback interface it finds.

`-i` *filename* — Display packets previously captured in *filename*. Without this option, `snoop` reads packets from the network interface. If a *filename*`.names` file is present, it is automatically loaded into the `snoop` IP address-to-name mapping table (See `-N` flag).

`-n` *filename* — Use *filename* as an IP address-to-name mapping table. This file must have the same format as the `/etc/hosts` file (IP address followed by the hostname).

`-o` *filename* — Save captured packets in *filename* as they are captured. (This *filename* is referred to as the "capture file".) The format of the capture file is RFC 1761–compliant. During packet capture, a count of the number of packets saved in the file is displayed. If you wish just to count packets without saving to a file, name the file `/dev/null`.

`-p` *first* [`,` `last`] — Select one or more packets to be displayed from a capture file. The *first* packet in the file is packet number 1.

`-q` — When capturing network packets into a file, do not display the packet count. This can improve packet capturing performance.

`-r` — Do not resolve the IP address to the symbolic name. This prevents `snoop` from generating network traffic while capturing and displaying packets. However, if the `-n` option is used, and an address is found in the mapping file, its corresponding name will be used.

`-s` *snaplen* — Truncate each packet after *snaplen* bytes. Usually the whole packet is captured. This option is useful if only

certain packet header information is required. The packet truncation is done within the kernel giving better utilization of the streams packet buffer. This means less chance of dropped packets due to buffer overflow during periods of high traffic. It also saves disk space when capturing large traces to a capture file. To capture only IP headers (no options) use a *snaplen* of 34. For UDP use 42, and for TCP use 54. You can capture RPC headers with a *snaplen* of 80 bytes. NFS headers can be captured in 120 bytes.

`-t [ r | a | d ]` Time-stamp presentation. Time-stamps are accurate to within 4 microseconds. The default is for times to be presented in `d` (delta) format (the time since receiving the previous packet). Option `a` (absolute) gives wall-clock time. Option `r` (relative) gives time relative to the first packet displayed. This can be used with the `-p` option to display time relative to any selected packet.

`-v` Verbose mode. Print packet headers in lots of detail. This display consumes many lines per packet and should be used only on selected packets.

`-x` *offset* [, *length*] Display packet data in hexadecimal and ASCII format. The *offset* and *length* values select a portion of the packet to be displayed. To display the whole packet, use an *offset* of 0. If a *length* value is not provided, the rest of the packet is displayed.

OPERANDS

expression Select packets either from the network or from a capture file. Only packets for which the expression is true will be selected. If no expression is provided it is assumed to be true.

Given a filter expression, `snoop` generates code for either the kernel packet filter or for its own internal filter. If capturing packets with the network interface, code for the kernel packet filter is generated. This filter is implemented as a streams module, upstream of the buffer module. The buffer module accumulates packets until it becomes full and passes the packets on to `snoop`. The kernel packet filter is very efficient, since it rejects unwanted packets in the kernel before they reach the packet buffer or `snoop`. The kernel packet filter has some limitations in its implementation; it is possible to construct filter expressions that it cannot handle. In this event, `snoop` tries to split the filter and do as much filtering in the kernel as possible. The remaining filtering is done by the packet filter for `snoop`. The `-C` flag can be used to view generated

code for either the packet filter for the kernel or the packet filter for `snoop`. If packets are read from a capture file using the `-i` option, only the packet filter for `snoop` is used.

A filter *expression* consists of a series of one or more boolean primitives that may be combined with boolean operators (`AND`, `OR`, and `NOT`). Normal precedence rules for boolean operators apply. Order of evaluation of these operators may be controlled with parentheses. Since parentheses and other filter expression characters are known to the shell, it is often necessary to enclose the filter expression in quotes. Refer to Example 2 for information about setting up more efficient filters.

The primitives are:

`host` *hostname*
: True if the source or destination address is that of `hostname`. The *hostname* argument may be a literal address. The keyword `host` may be omitted if the name does not conflict with the name of another expression primitive. For example, "`pinky`" selects packets transmitted to or received from the host `pinky`, whereas "`pinky and dinky`" selects packets exchanged between hosts `pinky AND dinky`.

  The type of address used depends on the primitive which precedes the `host` primitive. The possible qualifiers are "`inet`", "`inet6`", "`ether`", or none. These three primitives are discussed below. Having none of the primitives present is equivalent to "inet host hostname or inet6 host hostname". In other words, snoop tries to filter on all IP addresses associated with hostname.

inet or *inet6*
: A qualifier that modifies the `host` primitive that follows. If it is *inet*, then `snoop` tries to filter on all IPv4 addresses returned from a name lookup. If it is *inet6*, `snoop` tries to filter on all IPv6 addresses returned from a name lookup.

ipaddr, *atalkaddr*, or *etheraddr*
: Literal addresses, IP dotted, AppleTalk dotted, and ethernet colon are recognized. For example,

  - "`129.144.40.13`" matches all packets with that IP ;
  - "`2::9255:a00:20ff:fe73:6e35`" matches all packets with that IPv6 address as source or destination;
  - "`65281.13`" matches all packets with that AppleTalk address;
  - "`8:0:20:f:b1:51`" matches all packets with the ethernet address as source or destination.

An ethernet address beginning with a letter is interpreted as a hostname. To avoid this, prepend a zero when specifying the address. For example, if the ethernet address is "`aa:0:45:23:52:44`", then specify it by add a leading zero to make it "`0aa:0:45:23:52:44`".

`from` or `src`
A qualifier that modifies the following `host`, `net`, *ipaddr*, *atalkaddr*, *etheraddr*, `port` or `rpc` primitive to match just the source address, port, or RPC reply.

`to` or `dst`
A qualifier that modifies the following `host`, `net`, *ipaddr*, *atalkaddr*, *etheraddr*, `port` or `rpc` primitive to match just the destination address, port, or RPC call.

`ether`
A qualifier that modifies the following `host` primitive to resolve a name to an ethernet address. Normally, IP address matching is performed.

`ethertype` *number*
True if the ethernet type field has value *number*. Equivalent to "`ether[12:2] =` *number*".

`ip`, `ip6`, `arp`, `rarp`, `pppoed`, `pppoes`
True if the packet is of the appropriate ethertype.

`pppoe`
True if the ethertype of the packet is either `pppoed` or `pppoes`.

`broadcast`
True if the packet is a broadcast packet. Equivalent to "`ether[2:4] = 0xffffffff`".

`multicast`
True if the packet is a multicast packet. Equivalent to "`ether[0] & 1 = 1`".

`apple`
True if the packet is an Apple Ethertalk packet. Equivalent to "`ethertype 0x809b or ethertype 0x80f3`".

`decnet`
True if the packet is a DECNET packet.

`greater` *length*
True if the packet is longer than *length*.

`less` *length*
True if the packet is shorter than *length*.

`udp`, `tcp`, `icmp`, `icmp6`, `ah`, `esp`
True if the IP or IPv6 protocol is of the appropriate type.

`net` *net*
True if either the IP source or destination address has a network number of *net*. The `from` or `to` qualifier may be used to select packets for which the network number occurs only in the source or destination address.

`port` *port*
True if either the source or destination port is *port*. The *port* may be either a port number or name from `/etc/services`. The `tcp` or `udp` primitives may be used to select TCP or UDP ports only. The `from` or `to` qualifier may be used to select packets for which the *port* occurs only as the source or destination.

`rpc` *prog* [, *vers* [, `proc`]]
True if the packet is an RPC call or reply packet for the protocol identified by *prog*. The *prog* may be either the name of an RPC protocol from `/etc/rpc` or a program number. The *vers* and `proc` may be used to further qualify the program *version* and *procedure* number, for example, "`rpc nfs,2,0`" selects all calls and replies for the NFS null procedure. The `to` or `from` qualifier may be used to select either call or reply packets only.

`ldap`
True if the packet is an LDAP packet on port 389.

`gateway` *host*
True if the packet used *host* as a gateway, that is, the ethernet source or destination address was for *host* but not the IP address. Equivalent to "`ether host` *host* and not host *host*".

`nofrag`
True if the packet is unfragmented or is the first in a series of IP fragments. Equivalent to "`ip[6:2] & 0x1fff = 0`".

expr relop expr
True if the relation holds, where *relop* is one of `>`, `<`, `>=`, `<=`, `=`, `!=`, and `expr` is an arithmetic expression composed of numbers, packet field selectors, the `length` primitive, and arithmetic operators `+`, `–`, `*`, `&`, `|`, `^`, and `%`. The arithmetic operators within `expr` are evaluated before the relational operator and normal precedence rules apply between the arithmetic operators, such as multiplication before addition. Parentheses may be used to control the order of evaluation. To use the value of a field in the packet use the following syntax:

base`[expr [: size ] ]`

where `expr` evaluates the value of an offset into the packet from a *base* offset which may be `ether`, `ip`, `udp`, `tcp`, or `icmp`. The `size` value specifies the size of the field. If not given, 1 is assumed. Other legal values are 2 and 4. For example,

```
ether[0] & 1 = 1
```

is equivalent to `multicast`

```
ether[2:4] = 0xffffffff
```

is equivalent to `broadcast`.

```
ip[ip[0] & 0xf * 4 : 2] = 2049
```

is equivalent to `udp[0:2] = 2049`

```
ip[0] & 0xf > 5
```

selects IP packets with options.

```
ip[6:2] & 0x1fff = 0
```

eliminates IP fragments.

```
udp and ip[6:2]&0x1fff = 0 and udp[6:2] != 0
```

finds all packets with UDP checksums.

The `length` primitive may be used to obtain the length of the packet. For instance "`length > 60`" is equivalent to "`greater 60`", and "`ether[length – 1]`" obtains the value of the last byte in a packet.

`and`
Perform a logical AND operation between two boolean values. The AND operation is implied by the juxtaposition of two boolean expressions, for example "`dinky pinky`" is the same as "`dinky AND pinky`".

`or` or `,`
Perform a logical OR operation between two boolean values. A comma may be used instead, for example, "`dinky,pinky`" is the same as "`dinky OR pinky`".

`not` or `!`
Perform a logical NOT operation on the following boolean value. This operator is evaluated before AND or OR.

`slp`
True if the packet is an SLP packet.

`sctp`
True if the packet is a SCTP packet.

EXAMPLES

EXAMPLE 1 Using the `snoop` Command

Capture all packets and display them as they are received:

```
example# snoop
```

EXAMPLE 1 Using the `snoop` Command *(Continued)*

Capture packets with host `funky` as either the source or destination and display them as they are received:

```
example# snoop funky
```

Capture packets between `funky` and `pinky` and save them to a file. Then inspect the packets using times (in seconds) relative to the first captured packet:

```
example# snoop -o cap funky pinky
example# snoop -i cap -t r | more
```

To look at selected packets in another capture file:

```
example# snoop -i pkts -p 99,108
 99   0.0027   boutique -> sunroof     NFS C GETATTR FH=8E6C
100   0.0046   sunroof -> boutique     NFS R GETATTR OK
101   0.0080   boutique -> sunroof     NFS C RENAME FH=8E6C MTra00192 to .nfs08
102   0.0102   marmot -> viper          NFS C LOOKUP FH=561E screen.r.13.i386
103   0.0072   viper -> marmot          NFS R LOOKUP No such file or directory
104   0.0085   bugbomb -> sunroof    RLOGIN C PORT=1023 h
105   0.0005   kandinsky -> sparky    RSTAT C Get Statistics
106   0.0004   beeblebrox -> sunroof  NFS C GETATTR FH=0307
107   0.0021   sparky -> kandinsky    RSTAT R
108   0.0073   office -> jeremiah        NFS C READ FH=2584 at 40960 for 8192
```

To look at packet 101 in more detail:

```
example# snoop -i pkts -v -p101
ETHER:  ----- Ether Header -----
ETHER:
ETHER:  Packet 101 arrived at 16:09:53.59
ETHER:  Packet size = 210 bytes
ETHER:  Destination = 8:0:20:1:3d:94, Sun
ETHER:  Source      = 8:0:69:1:5f:e,  Silicon Graphics
ETHER:  Ethertype = 0800 (IP)
ETHER:
IP:   ----- IP Header -----
IP:
IP:   Version = 4, header length = 20 bytes
IP:   Type of service = 00
IP:         ..0. .... = routine
IP:         ...0 .... = normal delay
IP:         .... 0... = normal throughput
IP:         .... .0.. = normal reliability
IP:   Total length = 196 bytes
IP:   Identification 19846
IP:   Flags = 0X
IP:   .0.. .... = may fragment
IP:   ..0. .... = more fragments
IP:   Fragment offset = 0 bytes
IP:   Time to live = 255 seconds/hops
IP:   Protocol = 17 (UDP)
IP:   Header checksum = 18DC
IP:   Source address = 129.144.40.222, boutique
IP:   Destination address = 129.144.40.200, sunroof
```

EXAMPLE 1 Using the snoop Command *(Continued)*

```
IP:
UDP:  ----- UDP Header -----
UDP:
UDP:  Source port = 1023
UDP:  Destination port = 2049 (Sun RPC)
UDP:  Length = 176
UDP:  Checksum = 0
UDP:
RPC:  ----- SUN RPC Header -----
RPC:
RPC:  Transaction id = 665905
RPC:  Type = 0 (Call)
RPC:  RPC version = 2
RPC:  Program = 100003 (NFS), version = 2, procedure = 1
RPC:  Credentials: Flavor = 1 (Unix), len = 32 bytes
RPC:     Time = 06-Mar-90 07:26:58
RPC:     Hostname = boutique
RPC:     Uid = 0, Gid = 1
RPC:     Groups = 1
RPC:  Verifier   : Flavor = 0 (None), len = 0 bytes
RPC:
NFS:  ----- SUN NFS -----
NFS:
NFS:  Proc = 11 (Rename)
NFS:  File handle = 000016430000000100080000305A1C47
NFS:                597A00000008000020463l4AFC450000
NFS:  File name = MTra00192
NFS:  File handle = 000016430000000100080000305A1C47
NFS:                597A00000008000020463l4AFC450000
NFS:  File name = .nfs08
NFS:
```

To view just the NFS packets between `sunroof` and `boutique`:

```
example# snoop -i pkts rpc nfs and sunroof and boutique
1   0.0000   boutique -> sunroof    NFS C GETATTR FH=8E6C
2   0.0046    sunroof -> boutique   NFS R GETATTR OK
3   0.0080   boutique -> sunroof    NFS C RENAME FH=8E6C MTra00192 to .nfs08
```

To save these packets to a new capture file:

```
example# snoop -i pkts -o pkts.nfs rpc nfs sunroof boutique
```

To view encapsulated packets, there will be an indicator of encapsulation:

```
example# snoop ip-in-ip
sunroof -> boutique ICMP Echo request    (1 encap)
```

If -V is used on an encapsulated packet:

```
example# snoop -V ip-in-ip
sunroof -> boutique  ETHER Type=0800 (IP), size = 118 bytes
sunroof -> boutique  IP  D=129.144.40.222 S=129.144.40.200 LEN=104, ID=27497
sunroof -> boutique  IP  D=10.1.1.2 S=10.1.1.1 LEN=84, ID=27497
sunroof -> boutique  ICMP Echo request
```

EXAMPLE 2 Setting Up A More Efficient Filter

To set up a more efficient filter, the following filters should be used toward the end of the expression, so that the first part of the expression can be set up in the kernel: `greater`, `less`, `port`, `rpc`, `nofrag`, and `relop`. The presence of `OR` makes it difficult to split the filtering when using these primitives that cannot be set in the kernel. Instead, use parentheses to enforce the primitives that should be `OR`'d.

To capture packets between `funky` and `pinky` of type `tcp` or `udp` on `port` 80:

```
example# snoop funky and pinky and port 80 and tcp or udp
```

Since the primitive `port` cannot be handled by the kernel filter, and there is also an `OR` in the expression, a more efficient way to filter is to move the `OR` to the end of the expression and to use parentheses to enforce the `OR` between `tcp` and `udp`:

```
example# snoop funky and pinky and (tcp or udp) and port 80
```

EXIT STATUS

`0` Successful completion.

`1` An error occurred.

FILES

`/dev/audio`	Symbolic link to the system's primary audio device.
`/dev/null`	The null file.
`/etc/hosts`	Host name database.
`/etc/rpc`	RPC program number data base.
`/etc/services`	Internet services and aliases.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWrcmdc

SEE ALSO

`netstat`(1M), `hosts`(4), `rpc`(4), `services`(4), `attributes`(5), `audio`(7I), `bufmod`(7M), `dlpi`(7P), `le`(7D), `pfmod`(7M), `tun`(7M)

Callaghan, B. and Gilligan, R. *RFC 1761, Snoop Version 2 Packet Capture File Format*. Network Working Group. February 1995.

WARNINGS

The processing overhead is much higher for realtime packet interpretation. Consequently, the packet drop count may be higher. For more reliable capture, output raw packets to a file using the `-o` option and analyze the packets off-line.

Unfiltered packet capture imposes a heavy processing load on the host computer, particularly if the captured packets are interpreted realtime. This processing load further increases if verbose options are used. Since heavy use of `snoop` may deny computing resources to other processes, it should not be used on production servers. Heavy use of `snoop` should be restricted to a dedicated computer.

`snoop` does not reassemble IP fragments. Interpretation of higher level protocol halts at the end of the first IP fragment.

`snoop` may generate extra packets as a side-effect of its use. For example it may use a network name service (NIS or NIS+) to convert IP addresses to host names for display. Capturing into a file for later display can be used to postpone the address-to-name mapping until after the capture session is complete. Capturing into an NFS-mounted file may also generate extra packets.

Setting the `snaplen` (`-s` option) to small values may remove header information that is needed to interpret higher level protocols. The exact cutoff value depends on the network and protocols being used. For NFS Version 2 traffic using UDP on 10 Mb/s ethernet, do not set `snaplen` less than 150 bytes. For NFS Version 3 traffic using TCP on 100 Mb/s ethernet, `snaplen` should be 250 bytes or more.

`snoop` requires information from an RPC request to fully interpret an RPC reply. If an RPC reply in a capture file or packet range does not have a request preceding it, then only the RPC reply header will be displayed.

NAME

soconfig – configure transport providers for use by sockets

SYNOPSIS

/sbin/soconfig `-f` *file*

/sbin/soconfig *family type protocol* [*path*]

DESCRIPTION

The `soconfig` utility configures the transport provider driver for use with sockets. It specifies how the family, type, and protocol parameters in the `socket`(3SOCKET) call are mapped to the name of a transport provider such as `/dev/tcp`. This utility can be used to add an additional mapping or remove a previous mapping.

The `init`(1M) utility uses `soconfig` with the `sock2path`(4) file during the booting sequence.

OPTIONS

The following options are supported:

`-f` *file* Set up the `soconfig` configuration for each driver according to the information stored in `file`. A `soconfig` file consists of lines of at least the first three fields listed below, separated by spaces:

family type protocol path

These fields are described in the `OPERANDS` section below.

An example of `file` can be found in the `EXAMPLES` section below.

OPERANDS

The following operands are supported:

family The protocol family as listed in the `/usr/include/sys/socket.h` file, expressed as an integer.

type The socket type as listed in the `/usr/include/sys/socket.h` file, expressed as an integer.

protocol The protocol number as specified in the family-specific `include` file, expressed as an integer. For example, for `AF_INET` this number is specified in `/usr/include/netinet/in.h`. An unspecified protocol number is denoted with the value zero.

path The string that specifies the path name of the device that corresponds to the transport provider. If this parameter is specified, the configuration will be added for the specified family, type, and protocol. If this parameter is not specified, the configuration will be removed.

EXAMPLES

EXAMPLE 1 Using `soconfig`

The following example sets up `/dev/tcp` for family `AF_INET` and type `SOCK_STREAM`:

```
example# soconfig 2 2 0 /dev/tcp
```

EXAMPLE 1 Using `soconfig` *(Continued)*

The following is a sample file used with the `-f` option. Comment lines begin with a number sign (#):

```
#     Family   Type  Protocol    Path
      2        2        0       /dev/tcp
      2        2        6       /dev/tcp

      2        1        0       /dev/udp
      2        1        17      /dev/udp

      1        2        0       /dev/ticotsord
      1        1        0       /dev/ticlts

      2        4        0       /dev/rawip
```

FILES

`/etc/sock2path` file containing mappings from sockets to transport providers

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsr

SEE ALSO

`init`(1M), , `sock2path`(4), `attributes`(5)

Network Interfaces Programmer's Guide

NAME

soladdapp – add an application to the Solstice application registry

SYNOPSIS

/usr/snadm/bin/soladdapp [-r *registry*] -n *name* -i *icon* -e *executable* [*args*]

DESCRIPTION

soladdapp adds an application to the Solstice application registry. After it is added, the application is displayed in the Solstice Launcher main window (see solstice(1M)).

OPTIONS

-r *registry*	Define the full path name of the Solstice registry file.
-n *name*	Define the name of the tool to be registered.
-i *icon*	Define the full path name of the tool icon.
-e *executable*	Define the full path name of the tool.
args	Specify any arguments to use with the tool.

When executed without options, soladdapp uses /opt/SUNWadm/etc/.solstice_registry (the default registry path).

RETURN VALUES

0	on success
1	on failure
2	if the registry is locked
3	if the entry is a duplicate.

EXAMPLES

EXAMPLE 1 A sample display of the soladdapp command.

The following adds an application called Disk Manager to the Solstice application registry for display in the Solstice Launcher main window.

```
# soladdapp -r /opt/SUNWadm/etc/.solstice_registry -n "Disk Manager"
-i /opt/SUNWdsk/etc/diskmgr.xpm -e /opt/SUNWdsk/bin/diskmgr
```

FILES

/opt/SUNWadm/etc/.solstice_registry
The default registry path.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsadml

SEE ALSO

soldelapp(1M), solstice(1M), attributes(5)

NOTES

Globally registered applications are used by local and remote users sharing the software in a particular /opt directory. They can be added only using soladdapp.

NAME

soldelapp – remove an application from the Solstice application registry

SYNOPSIS

/usr/snadm/bin/soldelapp [-r *registry*] -n *name*

DESCRIPTION

soldelapp removes an application from the Solstice application registry. After removal, the application is no longer displayed in the Solstice Launcher main window (see solstice(1M)).

OPTIONS

-r *registry* Define the full path name of the Solstice registry file.

-n *name* Define the name of the tool to be removed.

When executed without options, soldelapp uses /opt/SUNWadm/etc/.solstice_registry (the default registry path).

RETURN VALUES

0 on success

1 on failure

2 if the registry is locked

3 if *name* is not found in the registry

4 if the named registry or default registry is not found

EXAMPLES

EXAMPLE 1 A sample display of the soldelapp command.

The following removes an application called Disk Manager from the Solstice application registry and the Solstice Launcher main window.

```
# soldelapp -r /opt/SUNWadm/etc/.solstice_registry -n "Disk Manager"
```

FILES

/opt/SUNWadm/etc/.solstice_registry
The default registry file.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsadml

SEE ALSO

soladdapp(1M), solstice(1M), attributes(5)

NOTES

Globally registered applications are used by local and remote users sharing the software in a particular /opt directory. They can be removed only using soldelapp.

NAME

solstice – access system administration tools with a graphical user interface

SYNOPSIS

/bin/solstice

DESCRIPTION

`solstice` used on a system presents the `Solstice Launcher`, a graphical user interface that provides access to the Solstice AdminSuite product family of system administration tools. The tools that appear in the launcher depend on what Solstice products you installed on your system.

Help is available by using the Help button.

USAGE

The `Solstice Launcher` allows you to do the following tasks:

Launch applications
Use the `Solstice Launcher` to launch system administration tools.

Register applications
Use the `Solstice Launcher` to add and register applications locally with the launcher.

Remove applications
Use the `Solstice Launcher` to remove locally registered applications.

Customize application properties
Use the `Solstice Launcher` to show, hide, or remove applications in the launcher, reorder the icons, change the launcher window width, modify applications properties, and add applications.

FILES

`/$HOME/.solstice_registry`
Local registry information.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsadml

SEE ALSO

`soladdapp`(1M), `soldelapp`(1M), `attributes`(5)

NOTES

The `Solstice Launcher` adds or removes local applications that are private to the user (not local to the system) only. The properties of globally registered applications that are used by local and remote users sharing the software from a particular `/opt` directory cannot be modified from the `Solstice Launcher`. To register global applications for use by local and remote users, use the `soladdapp`(1M) command. To remove globally registered applications, use the `soldelapp`(1M) command.

NAME

sppptun – PPP tunneling driver utility

SYNOPSIS

sppptun plumb

sppptun plumb *protocol device*

sppptun unplumb *interface*

sppptun query

DESCRIPTION

The `sppptun` utility is used to configure and query the Solaris PPP tunneling device driver, `/dev/sppptun`. Currently, only PPP over Ethernet (PPPoE) is supported, so the `plumb` and `unplumb` arguments are used to specify Ethernet interfaces that are to be used for PPPoE, and the `query` option lists the plumbed interfaces.

The use of `sppptun` to add interfaces is similar to the use of `ifconfig`(1M) to add interfaces to IP. The plumbing is done once for each interface, preferably at system start-up time, and is not normally manipulated on a running system. If multiple instances of PPP are run over a single interface, they share the plumbing to that interface. Plumbing for each session is not required (and not possible for PPPoE).

The proper way to plumb interfaces for PPPoE is to list the interfaces, one per line, in the `/etc/ppp/pppoe.if` file.

USAGE

`sppptun plumb`
When specified with no additional arguments, the plumb argument lists the protocols that are supported by the utility. These are the strings that are used as the *protocol* argument below.

`sppptun plumb` *protocol device*
This plumbs a new interface into the driver. The *protocol* parameter is `pppoe` for the PPP-carrying "Session Stage" connection or `pppoed` for the PPPoE "Discovery Stage" connection. Both connections must be present for each Ethernet interface that is to be used for PPPoE. The *device* parameter is the path name of the Ethernet interface to use (use `ifconfig`(1M) to list available devices). If the path begins with `/dev/`, then this portion may be omitted.

`sppptun unplumb` *interface*
This removes an existing interface from the driver and terminates any PPP sessions that were using the interface. The *interface* parameter is the name of the interface as reported when the interface was plumbed.

`sppptun query`
Displays the canonical names of all interfaces plumbed into the `/dev/sppptun` device driver.

EXAMPLES

EXAMPLE 1 Setting up to Use PPPoE on `hme0`

Plumb the `hme0` interface.

EXAMPLE 1 Setting up to Use PPPoE on `hme0` *(Continued)*

```
# sppptun plumb pppoed hme0
hme0:pppoed
# sppptun plumb pppoe hme0
hme0:pppoe
```

Remove the `hme0` interface.

```
# sppptun unplumb hme0:pppoed
# sppptun unplumb hme0:pppoe
```

EXAMPLE 2 Script to Remove All Plumbed Interfaces

```
#!/bin/sh
for intf in `sppptun query`
do
        sppptun unplumb $intf
done
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` One or more errors occurred.

FILES

`/etc/ppp/pppoe.if`	list of Ethernet interfaces to be plumbed at boot time
`/usr/sbin/sppptun`	executable command
`/dev/sppptun`	Solaris PPP tunneling device driver

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWpppdt

SEE ALSO

`pppd`(1M), `pppoec`(1M), `pppoed`(1M), `sppptun`(7M)

RFC 2516, Method for Transmitting PPP Over Ethernet (PPPoE), Mamakos et al, February 1999

NAME

spray – spray packets

SYNOPSIS

/usr/sbin/spray [-c *count*] [-d *delay*] [-l *length*] [-t *nettype*] *host*

DESCRIPTION

`spray` sends a one-way stream of packets to *host* using RPC, and reports how many were received, as well as the transfer rate. The *host* argument can be either a name or an Internet address.

`spray` is not useful as a networking benchmark, as it uses unreliable connectionless transports, UDP for example. `spray` can report a large number of packets dropped when the drops were caused by `spray` sending packets faster than they can be buffered locally, that is, before the packets get to the network medium.

OPTIONS

`-c` *count* — Specify how many packets to send. The default value of *count* is the number of packets required to make the total stream size 100000 bytes.

`-d` *delay* — Specify how many microseconds to pause between sending each packet. The default is `0`.

`-l` *length* — The *length* parameter is the numbers of bytes in the Ethernet packet that holds the RPC call message. Since the data is encoded using XDR, and XDR only deals with 32 bit quantities, not all values of *length* are possible, and `spray` rounds up to the nearest possible value. When *length* is greater than 1514, then the RPC call can no longer be encapsulated in one Ethernet packet, so the *length* field no longer has a simple correspondence to Ethernet packet size. The default value of *length* is `86 bytes`, the size of the RPC and UDP headers.

`-t` *nettype* — Specify class of transports. Defaults to `netpath`. See `rpc`(3NSL) for a description of supported classes.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWrcmdc

SEE ALSO

`rpc`(3NSL), `attributes`(5)

NAME

ssaadm – administration program for SPARCstorage Array and SPARCstorage RSM disk systems

SYNOPSIS

ssaadm [-v] [-e] *subcommand* [*subcommand_option*...] | *pathname*...

DESCRIPTION

The ssaadm program is an administrative command that manages the SPARCstorage Array and SPARCstorage RSM disk systems (henceforth called SPARCstorage systems). ssaadm performs a variety of control and query tasks depending on the command line arguments and options used.

The command line must contain a *subcommand* (listed under USAGE) and at least one *pathname*. Commands specific to either a SPARCstorage Array or a SPARCstorage RSM state that fact. It may also contain options and other parameters depending on the subcommand. The subcommand is applied to each of the pathnames on the command line.

pathname specifies the SPARCstorage system controller or a disk in the SPARCstorage system. The controller name is specified by its physical name, for example,

```
/devices/. . . /. . . /SUNW,soc@3,0/SUNW,
     pln@axxxxxxx,xxxxxxxx:ctlr
```

or by a name of the form

c*N*

where *N* is the logical controller number. ssaadm uses the c*N* name to find an entry in the /dev/rdsk directory of a disk that is attached to the SPARCstorage system controller. The /dev/rdsk entry is then used to determine the physical name of the SPARCstorage system controller. A disk in the SPARCstorage system is specified by its logical or physical device name, for example,

```
/dev/rdsk/c1t0d0s2
```

or

```
/devices/. . . /. . . /SUNW,soc@3,0/SUNW,
     pln@axxxxxxx,xxxxxxxx/ssd@0,0:c,raw
```

See disks(1M) for more information on logical names for disks and controllers.

OPTIONS

The following options are supported:

-e — Expert mode. This is required for the expert mode subcommands listed below.

-v — Verbose mode.

Subcommands and their options are described below. Expert mode subcommands are listed separately.

OPERANDS

The following operands are supported:

pathname The SPARCstorage system controller or a disk in the SPARCstorage system.

Subcommands

`display` [`-p`] *pathname* . . .

Display configuration information for the specified units or display performance information for the specified SPARCstorage Array controller. If *pathname* specifies the controller, the configuration information is displayed for all disks in the SPARCstorage Array. For each drive that has fast write enabled, (`FW`) are displayed after the drive identification.

`-p`

Display performance information for the specified SPARCstorage Array controller. The accumulation of the performance statistics must be enabled using the `perf_statistics` subcommand before displaying the performance information. If not enabled, all of the I/Os per second are displayed as zeros.

The performance display reports the following information:

`BUSY`

How busy the controller in the SPARCstorage Array is, expressed as a percentage.

`IOPS`

The total I/Os per second for the SPARCstorage Array.

entries for each disk

The total number of I/Os per second.

`download -f` *filename pathname*
`download -w` *wwn pathname*

Download an image to the SPARCstorage Array controller.

`-f`

Download the prom image specified by *filename* to the SPARCstorage Array controller FEPROMs. When the download is complete, the SPARCstorage Array must be reset in order to use the downloaded code. Note that the `download` subcommand modifies the FEPROM on the SPARCstorage Array and should be used *with caution*.

`-w`

Change the SPARCstorage Array controller's World Wide Name. *wwn* is a 12 digit hex number, leading zeros required. The new SPARCstorage Array controller's image have the least significant 6 bytes of the 8-byte World Wide Name modified to *wwn*.

`fast_write` [`-s`] `-c` *pathname*
`fast_write` [`-s`] `-d` *pathname*
`fast_write` [`-s`] `-e` *pathname*

Enable or disable the use of the NVRAM to enhance the performance of writes in the SPARCstorage Array. *pathname* may refer to the SPARCstorage Array controller or to an individual disk.

`-c`
Enable fast writes for synchronous writes only.

`-d`
Disable fast writes.

`-e`
Enable fast writes.

`-s`
Save the state that is currently being requested so it persists across power-cycles.

`fc_s_download` [`-f` *fcode-file*]
Download the fcode contained in the file *fcode-file* into *all* the FC/S Sbus Cards. This subcommand is interactive and expects user confirmation before downloading the fcode. When invoked without the [`-f` *fcode-file*] option, the current version of the fcode in each FC/S Sbus card is printed. Note that the `fc_s_download` subcommand should be used *only in single-user mode*; otherwise the FC/S card could be reset.

`insert_device` *pathname*
Guide user through hot insertion of a disk device.

This subcommand only applies to the RSM. See `NOTES` for hot plugging limitations.

`perf_statistics -d` *pathname*
`perf_statistics -e` *pathname*
Enable or disable the accumulation of performance statistics for the specified SPARCstorage Array controller. The accumulation of performance statistics must be enabled before using the `display -p` subcommand. This subcommand can be issued only to the SPARCstorage Array controller.

`-d`
Disable the accumulation of performance statistics.

`-e`
Enable the accumulation of performance statistics.

`purge` *pathname*
Purge any fast write data from NVRAM for one disk, or all disks if the controller is specified. This option should be used with caution, usually only when a drive has failed.

`release` *pathname*
Release a reservation held on the specified controllers or disks. When HA (High_Availability) Software is running on a system, do not use this subcommand to release a disk on an SSA. Doing so could cause problems for the HA software.

`remove_device` *pathname*
Guide user through hot removal of a disk device.

This subcommand only applies to the RSM. See `NOTES` for hot plugging limitations.

`replace_device` *pathname*
Guide user through hot replacement of a disk device.

This subcommand only applies to the RSM. See `NOTES` for hot plugging limitations.

`reserve` *pathname*
Reserve the specified controllers or disks for exclusive use by the issuing host. When HA (High_Availability) Software is running on a system, do not use this subcommand to reserve a disk on an SSA. Doing so could cause problems for the HA software.

`set_boot_dev` [`-y`] *pathname*
Set the boot-device variable in the PROM to the physical device name specified by *pathname* which can be a block special device or the pathname of the directory on which the boot file system is mounted. This subcommand normally runs interactively and requests confirmation for setting the default boot device in the PROM. The `-y` option can be used to run it in non-interactive mode, in which case no confirmation is requested or required.

`start` [`-t` *tray-number*] *pathname*
Spin up the specified disks. If *pathname* specifies the controller, this action applies to all disks in the SPARCstorage Array.

`-t`
Spin up all disks in the tray specified by *tray-number*. *pathname* must specify the controller.

`stop` [`-t` *tray-number*] *pathname*
Spin down the specified disks. If *pathname* specifies the controller, this action applies to all disks in the SPARCstorage Array.

`-t`
Spin down all disks in the tray specified by *tray-number*. *pathname* must specify the controller.

`sync_cache` *pathname*
Flush all outstanding writes for the specified disk from NVRAM to the media. If *pathname* specifies the controller, this action applies to all disks in the SPARCstorage Array.

SCSI Enclosure Services (SES) Subcommands

The SPARCstorage RSM tray is addressed by the using the logical or physical path of the SES device or specifying the controller followed by the tray number if that controller has multiple trays. The controller is addressed by `c`*N* or the physical path to the SPARCstorage Array's controller.

See `ses`(7D) for more information about environmental sensor cards and associated devices.

These subcommands also work with RSM trays directly attached to wide differential SCSI controllers.

`alarm` *pathname* | `controller` *tray-number*
Display the current state of the audible alarm.

`alarm_on` *pathname* | `controller` *tray-number*
`alarm_off` *pathname* | `controller` *tray-number*
Enable or disable the audible alarm for this enclosure.

`alarm_set` *pathname* | `controller` *tray-number* [*seconds*]
Set the audible alarm setting to seconds.

`env_display` *pathname* | `controller` *tray-number*
Display the environmental information for the specified unit.

`led` *pathname*
Display the current state of the led for the specified disk.

`led_on` *pathname*
`led_off` *pathname*
Turn on or off the led for this disk.

`power_off` *pathname* | `controller` *tray-number*
Power down this RSM. The RSM will need to be powered back on manually.

This subcommand does not work with RSMs directly attached to wide differential SCSI controllers.

Expert Mode Subcommands

See `NOTES` for limitations of these subcommands. Only users that are knowledgeable about the systems they are managing should use the expert mode subcommands.

For the following subcommands that work on a bus if a disk is specified then the bus that disk attached to is used.

`bus_getstate pathname`	Get and display the state of the specified bus.
`bus_quiesce pathname`	Quiesce the specified bus.
`bus_reset pathname`	Reset the specified bus.
`bus_resetall pathname`	Reset the specified bus and all devices on that bus.
`bus_unquiesce pathname`	Unquiesce the specified bus.
`dev_getstate pathname`	Get the state (`online` or `offline`) of the specified device.
`dev_reset pathname`	Reset the specified device.
`offline pathname`	Turn the specified disk offline.
`online pathname`	Turn the specified disk online.

EXAMPLES

EXAMPLE 1 Using `ssaadm` to remove a disk on an SSA

An example of using the expert mode hot plugging subcommands to hot remove a disk on a SSA follows. See `NOTES` for hot plugging limitations.

The first step reserves the SCSI device so that it can't be accessed via its second SCSI bus:

```
example# ssaadm reserve /dev/dsk/c1t8d0s2
```

The next two steps take the disk to be removed offline then quiesce the bus:

```
example# ssaadm -e offline /dev/dsk/c1t8d0s2
example# ssaadm -e bus_quiesce /dev/dsk/c1t8d0s2
```

The user then removes the disk and continues by unquiescing the bus, putting the disk back online, then releasing it:

```
example# ssaadm -e bus_unquiesce /dev/dsk/c1t8d0s2
example# ssaadm -e online /dev/dsk/c1t8d0s2
example# ssaadm release /dev/dsk/c1t8d0s2
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

non-zero An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWssaop

SEE ALSO

`disks`(1M), `luxadm`(1M), `attributes`(5), `ses`(7D)

SPARCstorage Array User's Guide

NOTES

Currently, only some device drivers support hot plugging. If hot plugging is attempted on a disk or bus where it is not supported, an error message of the form:

```
ssaadm: can't acquire "PATHNAME": No such file or directory
```

is displayed.

Do not quiesce any bus containing a disk with the `root`, `usr`, or `swap` partitions to avoid possible system deadlock.

NAME

sshd – secure shell daemon

SYNOPSIS

```
sshd [-diqQ46] [-b bits] [-f config_file] [-g login_grace_time] [-h
     host_key_file] [-k key_gen_time] [-p port] [-u len] [-V client_protocol_id]
```

DESCRIPTION

The `sshd` (Secure Shell daemon) is the daemon program for `ssh`(1). Together these programs replace `rlogin` and `rsh`, and provide secure encrypted communications between two untrusted hosts over an insecure network. The programs are intended to be as easy to install and use as possible.

`sshd` is the daemon that listens for connections from clients. It forks a new daemon for each incoming connection. The forked daemons handle key exchange, encryption, authentication, command execution, and data exchange.

This implementation of `sshd` supports both SSH protocol versions 1 and 2 simultaneously. Because of security weaknesses in the v1 protocol, it is recommended that sites run only v2, if possible. Support for v1 is provided to help sites with existing `ssh` v1 clients and servers to transition to v2. Support for v1 might not be available in a future release of Solaris.

`sshd` works as described in the following subsections.

SSH Protocol Version 1

Each host has a host-specific RSA key (normally 1024 bits) used to identify the host. Additionally, when the daemon starts, it generates a server RSA key (normally 768 bits). This key is normally regenerated every hour if it has been used, and is never stored on disk.

Whenever a client connects the daemon responds with its public host and server keys. The client compares the RSA host key against its own database to verify that it has not changed. The client then generates a 256–bit random number. It encrypts this random number using both the host key and the server key, and sends the encrypted number to the server. Both sides then use this random number as a session key which is used to encrypt all further communications in the session. The rest of the session is encrypted using a conventional cipher, currently Blowfish or 3DES, with 3DES being used by default. The client selects the encryption algorithm to use from those offered by the server.

Next, the server and the client enter an authentication dialog. The client tries to authenticate itself using `.rhosts` authentication, `.rhosts` authentication combined with RSA host authentication, RSA challenge-response authentication, or password-based authentication.

Rhosts authentication is normally disabled because it is fundamentally insecure, but can be enabled in the server configuration file if desired. System security is not improved unless `rshd`(1M), `rlogind`(1M), `rexecd`(1M), and `rexd`(1M) are disabled (thus completely disabling `rlogin`(1) and `rsh`(1) into the machine).

SSH Protocol Version 2

Version 2 works similarly to version 1: Each host has a host-specific DSA/RSA key. However, when the daemon starts, it does not generate a server key. Forward security is provided through a Diffie-Hellman key agreement. This key agreement results in a

shared session key. The rest of the session is encrypted using a symmetric cipher, currently Blowfish, 3DES, or AES. The client selects the encryption algorithm to use from those offered by the server. Additionally, session integrity is provided through a cryptographic message authentication code (`hmac-sha1` or `hmac-md5`).

Protocol version 2 provides a public key based user authentication method (PubKeyAuthentication) and conventional password authentication.

Command Execution and Data Forwarding

If the client successfully authenticates itself, a dialog for preparing the session is entered. At this time the client can request things like allocating a pseudo-tty, forwarding X11 connections, forwarding TCP/IP connections, or forwarding the authentication agent connection over the secure channel.

Finally, the client either requests a shell or execution of a command. The sides then enter session mode. In this mode, either side may send data at any time, and such data is forwarded to/from the shell or command on the server side, and the user terminal on the client side.

When the user program terminates and all forwarded X11 and other connections have been closed, the server sends command exit status to the client, and both sides exit.

`sshd` can be configured using command-line options or the configuration files `/etc/ssh/ssh/ssh_config` and `~/.ssh/config`, both described in `ssh_config`(4). Command-line options override values specified in the configuration file.

`sshd` rereads its configuration file when it receives a hangup signal, `SIGHUP`.

Host Access Control

The `sshd` daemon uses TCP Wrappers to restrict access to hosts. It uses the service name of `sshd` for `hosts_access()`. For more information on TCP Wrappers see `tcpd(1M)` and `hosts_access(3)` man pages, which are part of the `SUNWsfman` package (they are not SunOS man pages). TCP wrappers binaries, including `libwrap`, are in `SUNWtcpd`, a required package for `SUNWsshdu`, the package containing `sshd`.

OPTIONS

The options for `sshd` are as follows:

`-b` *bits*
Specifies the number of bits in the server key (the default is 768).

`-d`
Debug mode. The server sends verbose debug output to the system log, and does not put itself in the background. The server also will not fork and will only process one connection. This option is only intended for debugging for the server. Multiple `-d` options increase the debugging level. Maximum is 3.

`-f` *configuration_file*
Specifies the name of the configuration file. The default is `/etc/ssh/sshd_config`. `sshd` refuses to start if there is no configuration file.

`-g` *login_grace_time*
Gives the grace time for clients to authenticate themselves (the default is 300 seconds). If the client fails to authenticate the user within this number of seconds, the server disconnects and exits. A value of zero indicates no limit.

`-h` *host_key_file*
Specifies the file from which the host key is read (the default is `/etc/ssh/ssh_host_key`). This option must be given if `sshd` is not run as root (as the normal host file is normally not readable by anyone but root).

`-i`
Specifies that `sshd` is being run from `inetd`. `sshd` is normally not run from `inetd` because it needs to generate the server key before it can respond to the client, and this may take tens of seconds. Clients would have to wait too long if the key was regenerated every time. However, with small key sizes (for example, 512) using `sshd` from `inetd` may be reasonable.

`-k` *key_gen_time*
Specifies how often the server key is regenerated (the default is 3600 seconds, or one hour). The motivation for regenerating the key fairly often is that the key is not stored anywhere, and after about an hour, it becomes impossible to recover the key for decrypting intercepted communications even if the machine is cracked into or physically seized. A value of zero indicates that the key will never be regenerated.

`-p` *port*
Specifies the port on which the server listens for connections (the default is 22).

`-q`
Quiet mode. Nothing is sent to the system log. Normally the beginning, authentication, and termination of each connection is logged.

`-u` *len*
Used to specify the size of the field in the `utmp` structure that holds the remote host name. If the resolved host name is longer than *len*, the dotted decimal value will be used instead. This allows hosts with very long host names that overflow this field to still be uniquely identified. Specifying `-u0` indicates that only dotted decimal addresses should be put into the `utmp` file.

`-4`
Forces `sshd` to use IPv4 addresses only.

`-6`
Forces `sshd` to use IPv6 addresses only.

authorized_keys File Format

The `$HOME/.ssh/authorized_keys` file lists the public keys that are permitted for RSA authentication. Each line of the file contains one key (empty lines and lines starting with a hash mark [#] are ignored as comments). Each line consists of the following fields, separated by spaces: options, bits, exponent, modulus, comment. The options field is optional; its presence is determined by whether the line starts with a number or not (the option field never starts with a number). The bits, exponent, modulus and comment fields give the RSA key; the comment field is not used for anything (but may be convenient for the user to identify the key).

Note that lines in this file are usually several hundred bytes long (because of the size of the RSA key modulus). You will find it very inconvenient to type them in; instead, copy the `identity.pub` file and edit it.

The options (if present) consist of comma-separated option specifications. No spaces are permitted, except within double quotes. The following option specifications are supported:

`from="`*pattern-list*`"`
Specifies that in addition to RSA authentication, the canonical name of the remote host must be present in the comma-separated list of patterns (`*` and `?` serve as wildcards). The list can also contain patterns negated by prefixing them with `!`; if the canonical host name matches a negated pattern, the key is not accepted. The purpose of this option is to optionally increase security: RSA authentication by itself does not trust the network or name servers or anything (but the key); however, if somebody somehow steals the key, the key permits an intruder to log in from anywhere in the world. This additional option makes using a stolen key more difficult (name servers and/or routers would have to be compromised in addition to just the key).

`command="`*command*`"`
Specifies that the *command* is executed whenever this key is used for authentication. The command supplied by the user (if any) is ignored. The command is run on a `pty` if the connection requests a `pty`; otherwise it is run without a `tty`. A quote can be included in the command by quoting it with a backslash. This option might be useful to restrict certain RSA keys to perform only a specific operation. An example might be a key that permits remote backups but nothing else. Note that the client might specify TCP/IP and/or X11 forwarding unless they are explicitly prohibited.

`environment="`*NAME=value*`"`
Specifies that the string *NAME=value* is to be added to the environment when logging in using this key. Environment variables set this way override other default environment values. Multiple options of this type are permitted.

`no-port-forwarding`
Forbids TCP/IP forwarding when this key is used for authentication. Any port forward requests by the client will return an error. This might be used, for example, in connection with the `command` option.

`no-X11-forwarding`
Forbids X11 forwarding when this key is used for authentication. Any X11 forward requests by the client will return an error.

`no-agent-forwarding`
Forbids authentication agent forwarding when this key is used for authentication.

`no-pty`
Prevents `tty` allocation (a request to allocate a `pty` will fail).

ssh_known_hosts File Format

The `/etc/ssh/ssh_known_hosts` and `$HOME/.ssh/known_hosts` files contain host public keys for all known hosts. The global file should be prepared by the administrator (optional), and the per-user file is maintained automatically: whenever the user connects from an unknown host its key is added to the per-user file.

Each line in these files contains the following fields: hostnames, bits, exponent, modulus, comment. The fields are separated by spaces.

Hostnames is a comma-separated list of patterns (`*` and `?` act as wildcards); each pattern in turn is matched against the canonical host name (when authenticating a client) or against the user-supplied name (when authenticating a server). A pattern can also be preceded by `!` to indicate negation: if the host name matches a negated pattern, it is not accepted (by that line) even if it matched another pattern on the line.

Bits, exponent, and modulus are taken directly from the RSA host key; they can be obtained, for example, from `/etc/ssh/ssh_host_rsa_key.pub`. The optional comment field continues to the end of the line, and is not used.

Lines starting with a hash mark (#) and empty lines are ignored as comments.

When performing host authentication, authentication is accepted if any matching line has the proper key. It is thus permissible (but not recommended) to have several lines or different host keys for the same names. This will inevitably happen when short forms of host names from different domains are put in the file. It is possible that the files contain conflicting information; authentication is accepted if valid information can be found from either file.

Note that the lines in these files are typically hundreds of characters long.You should definitely not type in the host keys by hand. Rather, generate them by a script or by taking `/etc/ssh/ssh_host_rsa_key.pub` and adding the host names at the front.

EXAMPLES

EXAMPLE 1 `authorized_key` File Entries

The following are examples of `authorized_key` file entries.

1024 33 12121...312314325 ylo@foo.bar

from="*.niksula.hut.fi,!pc.niksula.hut.fi" 1024 35 23...2334 ylo@niksula

command="dump /home",no-pty,no-port-forwarding 1024 33 23...2323 backup.hut.fi

EXAMPLE 2 `ssh_known_hosts` File Entries

The following are examples of `ssh_known_hosts` file entries.

closenet,closenet.hut.fi,...,130.233.208.41 1024 37 159...93 closenet.hut.fi

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

FILES

`/etc/ssh/sshd_config`
Contains configuration data for `sshd`. This file should be writable by root only, but it is recommended (though not necessary) that it be world-readable.

`/etc/ssh/ssh_host_key`
Contains the private part of the host key. This file should only be owned by root, readable only by root, and not accessible to others. Note that `sshd` does not start if this file is group/world-accessible.

`/etc/ssh/ssh_host_key.pub`
Contains the public part of the host key. This file should be world-readable but writable only by root. Its contents should match the private part. This file is not used for encryption; it is provided only for the convenience of the user so its contents can be copied to known hosts files. These two files are created using `ssh-keygen`(1).

`/var/run/sshd.pid`
Contains the process ID of the `sshd` listening for connections. If there are several daemons running concurrently for different ports, this contains the pid of the one started last. The content of this file is not sensitive; it can be world-readable.

`/etc/ssh/ssh_known_hosts` and `$HOME/.ssh/known_hosts`
These files are consulted when using `rhosts` with RSA host authentication to check the public key of the host. The key must be listed in one of these files to be accepted. The client uses the same files to verify that the remote host is the one it intended to connect. These files should be writable only by root or the owner. `/etc/ssh/ssh_known_hosts` should be world-readable, and `$HOME/.ssh/known_hosts` can but need not be world-readable.

`/etc/nologin`
If this file exists, `sshd` refuses to let anyone except root log in. The contents of the file are displayed to anyone trying to log in, and non-root connections are refused. The file should be world-readable.

`$HOME/.rhosts`
This file contains host-username pairs, separated by a space, one per line. The given user on the corresponding host is permitted to log in without password. The same file is used by `rlogind` and `rshd`. The file must be writable only by the user; it is recommended that it not be accessible by others. It is also possible to use `netgroups` in the file. Either host or user name may be of the form *+@groupname* to specify all hosts or all users in the group.

`$HOME/.shosts`
For `ssh`, this file is exactly the same as for `.rhosts`. However, this file is not used by `rlogin` and `rshd`, so using this permits access using SSH only.

`/etc/hosts.equiv`
This file is used during `.rhosts` authentication. In its simplest form, this file contains host names, one per line. Users on these hosts are permitted to log in

without a password, provided they have the same user name on both machines. The host name can also be followed by a user name; such users are permitted to log in as any user on this machine (except root). Additionally, the syntax *+@group* can be used to specify netgroups. Negated entries start with a hyphen (-).

If the client host/user is successfully matched in this file, login is automatically permitted, provided the client and server user names are the same. Additionally, successful RSA host authentication is normally required. This file must be writable only by root; it is recommended that it be world-readable.

Warning: It is almost never a good idea to use user names in `hosts.equiv`. Beware that it really means that the named user(s) can log in as anybody, which includes `bin`, `daemon`, `adm`, and other accounts that own critical binaries and directories. For practical purposes, using a user name grants the user root access. Probably the only valid use for user names is in negative entries. Note that this warning also applies to `rsh/rlogin`.

`/etc/ssh/shosts.equiv`
This file is processed exactly as `/etc/hosts.equiv`. However, this file might be useful in environments that want to run both `rsh/rlogin` and `ssh`.

`$HOME/.ssh/environment`
This file is read into the environment at login (if it exists). It can contain only empty lines, comment lines (that start with #), and assignment lines of the form *name=value*. The file should be writable only by the user; it need not be readable by anyone else.

`$HOME/.ssh/rc`
If this file exists, it is run with `/bin/sh` after reading the environment files but before starting the user's shell or command. If X11 spoofing is in use, this will receive the "`proto cookie`" pair in standard input (and `DISPLAY` in environment). This must call `xauth`(1) in that case.

The primary purpose of `$HOME/.ssh/rc` is to run any initialization routines that might be needed before the user's home directory becomes accessible; AFS is a particular example of such an environment.

This file will probably contain some initialization code followed by something similar to:

```
if read proto cookie;
then echo add $DISPLAY $proto $cookie | xauth -q -;
fi
```

If this file does not exist, `/etc/ssh/sshrc` is run, and if that does not exist, `xauth` is used to store the cookie. `$HOME/.ssh/rc` should be writable only by the user, and need not be readable by anyone else.

`/etc/ssh/sshrc`
Similar to `$HOME/.ssh/rc`. This can be used to specify machine-specific login-time initializations globally. This file should be writable only by root, and should be world-readable.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsshdu

SEE ALSO

`scp`(1), `ssh`(1), `ssh-add`(1), `ssh-agent`(1), `ssh-keygen`(1), `sftp-server`(1M), `sshd`(1M), `sshd_config`(4), `attributes`(5)

To view license terms, attribution, and copyright for OpenSSH, the default path is `/var/sadm/pkg/SUNWsshdr/install/copyright`. If the Solaris operating environment has been installed anywhere other than the default, modify the given path to access the file at the installed location.

AUTHORS

OpenSSH is a derivative of the original and free `ssh` 1.2.12 release by Tatu Ylonen. Aaron Campbell, Bob Beck, Markus Friedl, Niels Provos, Theo de Raadt and Dug Song removed many bugs, added newer features and created Open SSH. Markus Friedl contributed the support for SSH protocol versions 1.4 and 2.0.

NAME

statd – network status monitor

SYNOPSIS

/usr/lib/nfs/statd

DESCRIPTION

`statd` is an intermediate version of the status monitor. It interacts with `lockd`(1M) to provide the crash and recovery functions for the locking services on NFS. `statd` keeps track of the clients with processes which hold locks on a server. When the server reboots after a crash, `statd` sends a message to the `statd` on each client indicating that the server has rebooted. The client `statd` processes then inform the `lockd` on the client that the server has rebooted. The client `lockd` then attempts to reclaim the lock(s) from the server.

`statd` on the client host also informs the `statd` on the server(s) holding locks for the client when the client has rebooted. In this case, the `statd` on the server informs its `lockd` that all locks held by the rebooting client should be released, allowing other processes to lock those files.

FILES

`/var/statmon/sm`
lists hosts and network addresses to be contacted after a reboot

`/var/statmon/sm.bak`
lists hosts and network addresses that could not be contacted after last reboot

`/var/statmon/state`
includes a number which changes during a reboot

`/usr/include/rpcsvc/sm_inter.x`
contains the rpcgen source code for the interface services provided by the statd daemon.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnfscu

SEE ALSO

`lockd`(1M), `attributes`(5)

System Administration Guide: IP Services

NOTES

The crash of a server is only detected upon its recovery.

NAME

strace – print STREAMS trace messages

SYNOPSIS

`strace` [*mid sid level...*]

DESCRIPTION

`strace` without arguments writes all STREAMS event trace messages from all drivers and modules to its standard output. These messages are obtained from the STREAMS log driver (see `log`(7D)). If arguments are provided, they must be in triplets of the form *mid, sid, level*, where *mid* is a STREAMS module ID number, *sid* is a sub-ID number, and *level* is a tracing priority level. Each triplet indicates that tracing messages are to be received from the given module/driver, sub-ID (usually indicating minor device), and priority level equal to, or less than the given level. The token `all` may be used for any member to indicate no restriction for that attribute.

The format of each trace message output is:

<*seq*> <`time`> <*ticks*> <*level*> <*flags*> <*mid*> <*sid*> <*text*>

<*seq*>	trace sequence number
<`time`>	time of message in *hh:mm:ss*
<*ticks*>	time of message in machine ticks since boot
<*level*>	tracing priority level
<*flags*>	E : message is also in the error log F : indicates a fatal error N : mail was sent to the system administrator (hardcoded as root)
<*mid*>	module ID number of source
<*sid*>	sub-ID number of source
<*text*>	formatted text of the trace message

Once initiated, `strace` will continue to execute until terminated by the user.

EXAMPLES

EXAMPLE 1 A sample output of the `strace` command:

The following example outputs all trace messages from the module or driver whose module ID is `41`:

```
strace  41 all all
```

The following example outputs those trace messages from driver or module ID `41` with sub-IDs `0`, `1`, or `2`:

```
strace  41 0 1  41 1 1  41 2 0
```

Messages from sub-IDs `0` and `1` must have a tracing level less than or equal to `1`. Those from sub-ID `2` must have a tracing level of `0`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO `attributes`(5), `log`(7D)

STREAMS Programming Guide

NOTES

- There is no restriction to the number of `strace` processes opening the STREAMS log driver at a time.
- The log-driver records the list of the triplets specified in the command invocation, and compares each potential trace message against this list to decide if it should be formatted and sent up to the `strace` process. Hence, long lists of triplets will have a greater impact on overall STREAMS performance. Running `strace` will have the most impact on the timing of the modules and drivers generating the trace messages that are sent to the `strace` process. If trace messages are generated faster than the `strace` process can handle them, some of the messages will be lost. This last case can be determined by examining the sequence numbers on the trace messages output.

NAME

strclean – STREAMS error logger cleanup program

SYNOPSIS

strclean [-a *age*] [-d *logdir*]

DESCRIPTION

strclean is used to clean up the STREAMS error logger directory on a regular basis (for example, by using cron. By default, all files with names matching error.* in /var/adm/streams that have not been modified in the last three days are removed.

OPTIONS

The following options are supported:

-a*age* The maximum age in days for a log file can be changed using the -a option.

-d*logdir* A directory other than /var/adm/streams can be specified using the -d option.

EXAMPLES

EXAMPLE 1 A sample of using the strclean command.

This example has the same result as running strclean with no arguments:

```
example%  strclean -d /var/adm/streams -a 3
```

FILES

/var/adm/streams/error.*

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

cron(1M), strerr(1M), attributes(5)

STREAMS Programming Guide

NOTES

strclean is typically run from cron on a daily or weekly basis.

NAME

strerr – STREAMS error logger daemon

SYNOPSIS

```
strerr
```

DESCRIPTION

`strerr` receives error log messages from the STREAMS log driver (see `log`(7D)) and appends them to a log file. The resultant error log files reside in the directory `/var/adm/streams`, and are named `error.`*mm*`-dd`, where *mm* is the month and `dd` is the day of the messages contained in each log file.

The format of an error log message is:

<*seq*> <`time`> <*ticks*> <*flags*> <*mid*> <*sid*> <*text*>

<*seq*>	error sequence number
<`time`>	time of message in hh:mm:ss
<*ticks*>	time of message in machine ticks since boot priority level
<*flags*>	`T` : the message was also sent to a tracing process `F` : indicates a fatal error `N` : send mail to the system administrator (hardcoded as root)
<*mid*>	module ID number of source
<*sid*>	sub-ID number of source
<*text*>	formatted text of the error message

Messages that appear in the error log are intended to report exceptional conditions that require the attention of the system administrator. Those messages which indicate the total failure of a STREAMS driver or module should have the `F` flag set. Those messages requiring the immediate attention of the administrator will have the `N` flag set, which causes the error logger to send the message to the system administrator using `mail`. The priority level usually has no meaning in the error log but will have meaning if the message is also sent to a tracer process.

Once initiated, `strerr` continues to execute until terminated by the user. It is commonly executed asynchronously.

FILES

`/var/adm/streams/error.`*mm*`-dd` error log file.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`attributes`(5), `log`(7D)

STREAMS Programming Guide

NOTES There is no restriction to the number of `strerr` processes opening the STREAMS log driver at a time.

If a module or driver is generating a large number of error messages, running the error logger will cause a degradation in STREAMS performance. If a large burst of messages are generated in a short time, the log driver may not be able to deliver some of the messages. This situation is indicated by gaps in the sequence numbering of the messages in the log files.

NAME

sttydefs – maintain line settings and hunt sequences for TTY ports

SYNOPSIS

/usr/sbin/sttydefs `-a` *ttylabel* `[-b] [-f` *final-flags*`] [-i` *initial-flags*`] [-n` *nextlabel*`]`

/usr/sbin/sttydefs `-l` [*ttylabel*]

/usr/sbin/sttydefs `-r` *ttylabel*

DESCRIPTION

`sttydefs` is an administrative command that maintains the line settings and hunt sequences for the system's TTY ports by making entries in, and deleting entries from the `/etc/ttydefs` file.

`sttydefs` with a `-a` or `-r` option may be invoked only by the super-user. `sttydefs` with `-l` may be invoked by any user on the system.

OPTIONS

The following options are supported:

`-a` *ttylabel*
: Add a record to the `ttydefs` file, using *ttylabel* as its label. The following describes the effect of the `-b`, `-n`, `-i`, or `-f` options when used in conjunction with the `-a` option:

`-b`
: Enable autobaud. Autobaud allows the system to set the line speed of a given TTY port to the line speed of the device connected to the port without the user's intervention.

`-f` *final-flags*
: Specify the value to be used in the *final-flags* field in `/etc/ttydefs`. *final-flags* must be in a format recognized by the `stty` command. *final-flags* are the `termio`(7I) settings used by `ttymon` after receiving a successful connection request and immediately before invoking the service on the port. If this option is not specified, `sttydefs` will set *final-flags* equal to the `termio`(7I) flags `9600` and `sane`.

`-i` *initial-flags*
: Specify the value to be used in the *initial-flags* field in `/etc/ttydefs`. *initial-flags* must be in a format recognized by the `stty` command. These flags are used by `ttymon` when searching for the correct baud rate. They are set prior to writing the prompt. If this option is not specified, `sttydefs` will set *initial-flags* equal to the `termio`(7I) flag `9600`.

`-n` *nextlabel*
: Specify the value to be used in the *nextlabel* field in `/etc/ttydefs`. If this option is not specified, `sttydefs` will set *nextlabel* equal to *ttylabel*.

`-l`[*ttylabel*]
: If a *ttylabel* is specified, `sttydefs` displays the record from `/etc/ttydefs` whose TTY label matches the specified *ttylabel*. If no *ttylabel* is specified, `sttydefs`

displays the entire contents of `/etc/ttydefs`. `sttydefs` verifies that each entry it displays is correct and that the entry's *nextlabel* field references an existing

`-r` *ttylabel* — Remove any record in the `ttydefs` file that has *ttylabel* as its label.

OUTPUT

If successful, `sttydefs` will exit with a status of `0`. `sttydefs -l` will generate the requested information and send it to standard output.

EXAMPLES

EXAMPLE 1 A sample of `sttydefs` command.

The following command lists all the entries in the `ttydefs` file and prints an error message for each invalid entry that is detected.

```
example# sttydefs -l
```

The following shows a command that requests information for a single label and its output:

```
example# sttydefs -l 9600
------------------------------------------------------------------
9600:9600 hupcl erase ^h:9600 sane ixany tab3 hupcl erase ^h::4800
------------------------------------------------------------------
ttylabel:    9600
initial flags:    9600 hupcl erase ^h
final flags:    9600 sane ixany tab3 hupcl erase ^h
autobaud:    no
nextlabel:      4800
```

The following sequence of commands will add the labels `1200`, `2400`, `4800`, and `9600` and put them in a circular list:

```
sttydefs -a 1200 -n 2400 -i 1200 -f "1200 sane"
sttydefs -a 2400 -n 4800 -i 2400 -f "2400 sane"
sttydefs -a 4800 -n 9600 -i 4800 -f "4800 sane"
sttydefs -a 9600 -n 1200 -i 9600 -f "9600 sane"
```

FILES

`/etc/ttydefs`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`attributes`(5), `termio`(7I)

NAME

su – become super user or another user

SYNOPSIS

su [-] [*username* [arg...]]

DESCRIPTION

The su command allows one to become another user without logging off or to assume a role. The default user *name* is root (super user).

To use su, the appropriate password must be supplied (unless the invoker is already root). If the password is correct, su creates a new shell process that has the real and effective user ID, group IDs, and supplementary group list set to those of the specified *username*. Additionally, the new shell's project ID is set to the default project ID of the specified user. See getdefaultproj(3PROJECT), setproject(3PROJECT). The new shell will be the shell specified in the shell field of *username*'s password file entry (see passwd(4)). If no shell is specified, /usr/bin/sh is used (see sh(1)). If superuser privilege is requested and the shell for the superuser cannot be invoked using exec(2), /sbin/sh is used as a fallback. To return to normal user ID privileges, type an EOF character (CTRL-D) to exit the new shell.

Any additional arguments given on the command line are passed to the new shell. When using programs such as sh, an *arg* of the form -c *string* executes *string* using the shell and an *arg* of -r gives the user a restricted shell.

The following statements are true if the login shell is /usr/bin/sh or an empty string (which defaults to /usr/bin/sh) in the specific user's password file entry. If the first argument to su is a dash (), the environment will be changed to what would be expected if the user actually logged in as the specified user. Otherwise, the environment is passed along, with the exception of $PATH, which is controlled by PATH and SUPATH in /etc/default/su.

All attempts to become another user using su are logged in the log file /var/adm/sulog (see sulog(4)).

SECURITY

su uses pam(3PAM) for authentication, account management, and session management.

The PAM configuration policy, listed through /etc/pam.conf, specifies the modules to be used for su. The following example shows a partial pam.conf file with entries for the su command using the authentication, account management, and session management module.

```
su   auth        requisite   pam_authtok_get.so.1
su   auth        required    pam_dhkeys.so.1
su   auth        required    pam_unix_auth.so.1

su   account     required    pam_unix_roles.so.1
su   account     required    pam_unix_projects.so.1
su   account     required    pam_unix_account.so.1

su   session     required    pam_unix_session.so.1
```

If there are no entries for the `su` service, then the entries for the other service will be used. If multiple authentication modules are listed, then the user may be prompted for multiple passwords.

EXAMPLES

EXAMPLE 1 Becoming User `bin` While Retaining Your Previously Exported Environment

To become user `bin` while retaining your previously exported environment, execute:

```
example% su bin
```

EXAMPLE 2 Becoming User `bin` and Changing to `bin`'s Login Environment

To become user `bin` but change the environment to what would be expected if `bin` had originally logged in, execute:

```
example% su - bin
```

EXAMPLE 3 Executing command with user `bin`'s Environment and Permissions

To execute command with the temporary environment and permissions of user `bin`, type:

```
example% su - bin -c "command args"
```

ENVIRONMENT VARIABLES

Variables with `LD_` prefix are removed for security reasons. Thus, `su bin` will not retain previously exported variables with `LD_` prefix while becoming user `bin`.

If any of the `LC_*` variables (`LC_CTYPE`, `LC_MESSAGES`, `LC_TIME`, `LC_COLLATE`, `LC_NUMERIC`, and `LC_MONETARY`) (see `environ`(5)) are not set in the environment, the operational behavior of `su` for each corresponding locale category is determined by the value of the `LANG` environment variable. If `LC_ALL` is set, its contents are used to override both the `LANG` and the other `LC_*` variables. If none of the above variables are set in the environment, the "C" (U.S. style) locale determines how `su` behaves.

`LC_CTYPE` Determines how `su` handles characters. When `LC_CTYPE` is set to a valid value, `su` can display and handle text and filenames containing valid characters for that locale. `su` can display and handle Extended Unix Code (EUC) characters where any individual character can be `1`, `2`, or `3` bytes wide. `su` can also handle EUC characters of `1`, `2`, or more column widths. In the "C" locale, only characters from ISO 8859-1 are valid.

`LC_MESSAGES` Determines how diagnostic and informative messages are presented. This includes the language and style of the messages, and the correct form of affirmative and negative responses. In the "C" locale, the messages are presented in the default form found in the program itself (in most cases, U.S. English).

FILES

`$HOME/.profile`	user's login commands for sh and ksh
`/etc/passwd`	system's password file
`/etc/profile`	system-wide `sh` and `ksh` login commands

/var/adm/sulog	log file
/etc/default/su	the default parameters in this file are:
SULOG	If defined, all attempts to su to another user are logged in the indicated file.
CONSOLE	If defined, all attempts to su to root are logged on the console.
PATH	Default path. (/usr/bin:)
SUPATH	Default path for a user invoking su to root. (/usr/sbin:/usr/bin)
SYSLOG	Determines whether the syslog(3C) LOG_AUTH facility should be used to log all su attempts. LOG_NOTICE messages are generated for su's to root, LOG_INFO messages are generated for su's to other users, and LOG_CRIT messages are generated for failed su attempts.
SLEEPTIME	If present, sets the number of seconds to wait before login failure is printed to the screen and another login attempt is allowed. Default is 4 seconds. Minimum is 0 seconds. Maximum is 5 seconds.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

csh(1), env(1), ksh(1), login(1), roles(1), sh(1), syslogd(1M), exec(2), getdefaultproj(3PROJECT), setproject(3PROJECT), pam(3PAM), syslog(3C), pam.conf(4), passwd(4), profile(4), sulog(4), attributes(5), environ(5), pam_authtok_check(5), pam_authtok_get(5), pam_authtok_store(5), pam_dhkeys(5), pam_passwd_auth(5), pam_unix(5), pam_unix_account(5), pam_unix_auth(5), pam_unix_session(5)

NOTES

The pam_unix(5) module might not be supported in a future release. Similar functionality is provided by pam_authtok_check(5), pam_authtok_get(5), pam_authtok_store(5), pam_dhkeys(5), pam_passwd_auth(5), pam_unix_account(5), pam_unix_auth(5), and pam_unix_session(5).

NAME

sulogin – access single-user mode

SYNOPSIS

sulogin

DESCRIPTION

The sulogin utility is automatically invoked by init when the system is first started. It prompts the user to type the root password to enter system maintenance mode (single-user mode) or to type EOF (typically CTRL-D) for normal startup (multi-user mode). The user should never directly invoke sulogin.

The sulogin utility can prompt the user to enter the root password on a variable number of serial console devices, in addition to the traditional console device. See consadm(1M) and msglog(7D) for a description of how to configure a serial device to display the single-user login prompt.

FILES

/etc/default/sulogin — Default value can be set for the following flag:

PASSREQ — Determines if login requires a password. Default is PASSREQ=YES.

/etc/default/login — Default value can be set for the following flag:

SLEEPTIME — If present, sets the number of seconds to wait before login failure is printed to the screen and another login attempt is allowed. Default is 4 seconds. Minimum is 0 seconds. Maximum is 5 seconds.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsr

SEE ALSO

consadm(1M),init(1M), attributes(5),msglog(7D)

NAME

suninstall – install the Solaris environment

SYNOPSIS

```
suninstall
```

DESCRIPTION

`suninstall` is a forms-based subsystem for installing the operating system.

`suninstall` only exists on the Solaris CD-ROM and should only be invoked from there. Refer to the installation manual for more details.

`suninstall` allows installation of the operating system onto any stand-alone system. `suninstall` loads the software available on the CD-ROM. Refer to the installation manual for disk space requirements.

To abort the installation procedure, use the interrupt character (typically, CTRL-C).

USAGE

Refer to the installation manual for more information on the various menus and selections.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcdrom (Solaris CD)

SEE ALSO

`pkginfo`(1), `install`(1M), `pkgadd`(1M), `attributes`(5)

Solaris 9 Installation Guide

NOTES

It is advisable to exit `suninstall` through the exit options from the `suninstall` menus.

NAME

SUNWgfb_config – fbconfig module for configuring Sun XVR-1000 Graphics Accelerator

SYNOPSIS

```
fbconfig [-dev device-filename] [-active a | b | both | auto]
     [-res video-mode [now | try] [noconfirm]] [-file machine |
     system] [-doublewide  enable | disable | reverse]
     [-multisample enable | disable | auto [static | dynamic]]
     [-samples samples_per-pixel] [-g gamma-correction-value]
     [-gfile gamma-correction-file] [-master a | b | input] [-buffersize
     [width height] | auto]

fbconfig [-dev device-filename] -stream a | b [-port hd15 | dvid |
     svideo | auto] [-res video-mode [now | try] [noconfirm]] [-file
     machine | system] [-tile [lr | ll | rl | rr] [tt | tb | bb |
     bt] [strma | strmb | buff]] [-offset xoff-value yoff-value] [-slave
     [enable | disable] [internal | external]]

fbconfig [-dev device-filename] -defaults

fbconfig [-dev device-filename] [-propt | -prconf | -res \?]

fbconfig [-help | -list]
```

DESCRIPTION

`SUNWgfb_config` is the Sun XVR-1000 Graphics Accelerator device dependent layer for `fbconfig`(1M). It configures the Sun XVR-1000 Graphics Accelerator and some of the X11 window system defaults. The device can drive two monitors, each with a unique video stream (called stream a and stream b).

The first form of `SUNWgfb_config` shown in the synopsis above sets card options, which are options common to both stream a and stream b, and apply to the entire card.

The second form is used to set stream options, which are options specific to either stream a or stream b. The second form usage requires the `-stream` option to define which stream is affected.

Both the first form and the second form store the specified options in the `OWconfig` file. These options will be used to initialize the device the next time the window system is run on that device. Updating options in the `OWconfig` file provides persistence of these options across window system sessions and system reboots. For `-res` `now`, `-try`, `-slave`, `-master`, `-port`, and `-g` , the device will be immediately programmed.

The third form, which invokes the `-defaults` option, sets all card options and all stream options to their default values.

The fourth form, which invokes the `-prconf`, `-propt`, and `-res \?` options, queries the device for status that is card-specific.

The fifth form, which invokes the `-help`, and `-list` options, provides instruction on using `SUNWgfb_config` and a list of available devices. Additionally for the fifth form, all other options are ignored.

You can specify options for only one device at a time. Specifying options for multiple devices requires multiple invocations of `SUNWgfb_config`.

Only Sun XVR-1000 Graphics Accelerator-specific options can be specified through `SUNWgfb_config`. The normal window system options for specifying default depth, default visual class, and so forth are still specified as device modifiers on the command line when the X server is started.

You can also specify the `OWconfig` file that is to be updated. By default, the machine-specific file in the `/etc/openwin` directory tree is updated. You can use the `-file` option to specify an alternate file. For example, the system-global `OWconfig` file in the `/usr/openwin` directory tree can be updated instead.

GENERAL OPTIONS

`-dev` *device-filename*
Specifies the device's special file. The default is `/dev/fb`.

`-file machine | system`
Specifies which `OWconfig` file to update. If `machine`, the machine-specific `OWconfig` file in the `/etc/openwin` directory tree is used. If `system`, the global `OWconfig` file in the `/usr/openwin` directory tree is used. If the file does not exist, it is created.

`-res` *video-mode* [`now | try [ noconfirm ]`]
Specifies the video mode used to drive the monitor connected to the specified device. If `-res` is invoked without specifying `-stream`, then the last setting for `-active` will determine which *video-mode* value(s) are modified. If `-active` is set to both or auto, then both stream *video-mode* values will be modified.

The *video-mode* argument specifies resolution and timing information for the display (for example, `SUNW_STD_1280x1024x76`). The naming convention for the *video-mode* specifier is: < *origin* >_< *type* >_< *width* >x< *height* >x< *rate* >. The elements of the specifier are described as follows:

origin
This can be one of:

- `SUNW`, Sun derived resolution
- `VESA`, Video Electronics Standards Association-derived resolution
- `other`, other source

type
This can be one of:

- `STD`, normal resolution, usable by most display devices
- `DIG`, resolution tuned only for LCD flat panels
- `INT`, interlaced
- `STEREO`, stereo

`width`
screen width in pixels

`height`
screen height in pixels

`rate`
vertical frequency of the screen refresh

Note that some video-modes supported by the device, might not be supported by the monitor. The list of video-modes supported by the device and the monitor can be obtained by running `SUNWgfb_config` with the `-res \?` option (the fourth form shown in the command synopsis above).

The `-res` option also accepts additional, optional arguments, listed below, immediately following the video mode specification. Either `now` or `try` (`try` subsumes `now`) and `noconfirm` can be present.

`now` — If present, not only is the video mode be updated in the `OWconfig` file, but the device is immediately programmed to display this video mode. This is useful for changing the video mode before starting the window system.

Note – It is recommended that you not use this suboption with `SUNWgfb_config` while the configured device is being used (for example, while running the window system). Unpredictable results can occur. To run `SUNWgfb_config` with the `now` suboption, first bring the window system down. If the `now` suboption is used within a window system session, the video mode is changed immediately, but the width and height of the affected screen do not change until the window system is exited and reentered. In addition, the system might not recognize changes in stereo mode.

`noconfirm` — Using the `-res` option, the user can put the system into an unusable state, with no video output. To reduce the chance of this, the default behavior of `SUNWgfb_config` is to display a warning message and to ask the user whether to continue. The `noconfirm` bypasses this confirmation. This option is useful when `SUNWgfb_config` is being run from a shell script.

`try` — If present, the specified video mode will be programmed on a trial basis. The user is asked to confirm the video mode by typing **`y`** within 10 seconds. Alternatively, the user can terminate the trial before 10 seconds elapse by typing any character other than **`y`** or carriage return. Such input is considered a no and the previous video mode is restored. With a negative response, `SUNWgfb_config` does not change the video mode in the `OWconfig` file; other options specified still take effect. If a carriage return is typed, the user is asked (**`y`** or **`n`**) whether to keep the new video mode. The `try` suboption implies the `now` suboption. See the warning note in the description of the `now` suboption, above.

`-defaults`
Resets all option values to their default values.

`-propt`
Displays the current values of all options in the `OWconfig` file specified by the `-file` option for the device specified by the `-dev` option. Displays the values of options as they will be in the `OWconfig` file after the call to `SUNWgfb_config` completes. The following is an example display:

```
--- OpenWindows Configuration for /dev/fbs/gfb0 ---
OWconfig: machine
Active Streams: both
Samples Per Pixel: 2
Multisample Allocation Model: static
Multisample Mode: auto
Doublewide: disable
Gamma Correction Value: 2.22

--- OpenWindows Configuration for Stream a ---
Video Mode: SUNW_STD_1280x1024x76

--- OpenWindows Configuration for Stream b ---
Video Mode: VESA_STD_640x480x60
```

`-prconf`
Displays the XVR-1000 hardware configuration. The following is an example display:

```
--- Hardware Configuration for /dev/fb (SUNWgfb0) ---
Type:   Sun Graphics Accelerator
Part:   501-5865
Memory:
        MAJC:           32MB
        Texture:        256MB total
        3DRAM64:        5.0M pixels

Versions:  FCode 1.14  MCode 0.19  MAJC 2.1  FBC3 3.0  XChip 2.0

Video Streams:
   Stream a
         Current resolution Setting: SUNW_STD_1280x1024x76
         Monitor/EDID data (13W3)
                 Monitor Manufacturer: SUN
                 Monitor Name:  GDM-5410
                 EDID:  Version 1, Revision 2
   Stream b
         Current resolution Setting: VESA_STD_640x480x60
         Port: svideo
```

`-help`
Displays a list of the `SUNWgfb_config` command line options, along with a brief explanation of each.

`-res \?`
Displays list of defined *video-mode* names.

CARD OPTIONS

`-active a | b | both | auto`
Specifies which streams are enabled. `both` select both streams. The default is `auto`, which means whichever stream is chosen by the console.

`-multisample enable | disable | auto`
`[ static | dynamic ]`

The suboptions for `-multisample` are described as follows:

`disable`
No multisample is possible.

`enable`
Multisample is possible but is selected on a per-application basis.

`auto`
All Sun OpenGL applications are rendered using multisampling.

`static`
Multisample allocation occurs at X startup/config load time. The config *samples_per-pixel* or `max` parameter specifies the depth that is pre-allocated.

`dynamic`
OpenGL tasks allocate buffers themselves.

`-samples` *samples_per-pixel*
Specifies the number of samples/pixel to pre-allocate in static mode. Provides a hint to OpenGL in dynamic mode. The allowable choices for *samples_per-pixel* are 2, 3, 4, 5, 6, 8, 10, 16, and `max`. The default is `max`, which means to use the maximum number of samples that can be supported with the amount of memory available.

`-doublewide enable | disable | reverse`
This option makes it easy for you to combine both streams into one virtual display. If you specify `enable`, stream a is to the left of stream b. If `reverse` is specified, stream b is to the left of stream b. Both will be the same resolution defined with the `-res` option. If you specify `disable`, only one stream will be enabled.

`-g` *gamma-correction value*
This option changes the gamma correction value. By default the gamma correction value is 2.22. Any value less than zero is illegal. This option can be used while the window system is running. Changing the gamma correction value will affect all the windows being displayed using gamma-corrected visuals. The gamma correction value is also saved in the `OWconfig` file for the next time the window system starts.

`-gfile` *gamma-correction file*
This option loads the gamma correction table from the specified file. This file should be formatted to provide the gamma correction values for R, G and B channels on each line. Each of these values should be in hexadecimal format and separated from each other by at least one space. This file should provide 1024 such triplets. An example of this file is as follows:

```
0x000 0x000 0x000
0x001 0x001 0x001
0x002 0x002 0x002
...
...
0x3ff 0x3ff 0x3ff
```

Using this option, the gamma correction table can be loaded while the window system is running. The new gamma correction will affect all the windows being displayed using gamma-corrected visuals. Note, when gamma correction is being done using a user-specified table, the gamma correction value is undefined. By default, the window system assumes a gamma correction value of 2.22 and loads the gamma table it creates corresponding to this value.

`-master a | b | input`
This option controls the setting frame pins on the stereo/sync connector on the device. It also controls which stream drives stereo glasses, which attach to the same connector.

If `a` or `b` is selected, the card is setup to be a sync master, and the frame sync signal from the corresponding stream will be sent out this connector.

If you select `input`, the card is setup to take its frame sync from another card through this connector. This sync can then be used to sync either or both streams by setting the stream-specific `-slave` option(s) to `external`.

If stereo glasses are used, the `a` or `b` options select which stream is used for the sync signal to the glasses.

`-buffersize [` *width height* `] | auto`
Explicitly allows you to specify size of managed area, which occupies all or a portion of the device's 3D RAM. The default is `auto`, that is, to automatically size the buffer based on the current stream resolutions. This should be sufficient for most users.

STREAM OPTIONS

`-stream a | b`
Specifies for which stream options will be set. It is a required option for each of the other options in this section. It is optional for `-res`. Only one `-stream` option can be specified.

`-port hd15 | dvid | svideo | auto`
Directs stream b to the appropriate output connector: `hd-15`, `dvid`, or `svideo`. If `auto`, then the output connector for stream b is selected by the console. Stream a is always output through the 13W3 connector.

`-tile [ lr | ll | rl | rr ]`
`[ tt | tb | bb | bt ]`
`[ stra | strb | buff ]`
Stream-specific option aligns display of child stream (specified by `-stream`) relative to parent. Sub-options are as follows:

`ll` Align child left edge with parent left edge.

`lr` Align child left edge with parent right edge.

`rr` Align child right edge with parent right edge.

`rl` Align child right edge with parent left edge.

`tt` Align child top edge with parent top edge.

`tb`	Align child top edge with parent bottom edge.
`bb`	Align child bottom edge with parent bottom edge.
`bt`	Align child bottom edge with parent top edge.
`stra`	Parent is stream a; child must be stream b.
`strb`	Parent is stream b; child must be stream a.
`buff`	Parent is buffer.

An error is returned if parent and child are one in the same (for example, `-stream a -tile ll tt stra`). The default values are the same as the settings for `-doublewide`; stream a is set to `ll tt buff`, and stream b is set to `lr tt stra`.

`-offset` *xoff-value* | *yoff-value*
Offsets display of child stream (specified by `-stream`) relative to parent and child alignment edges specified by `-tile`.

xoff-value	Number of pixels offset in horizontal direction. Positive direction is right; negative is left. Default is 0, which means the two edges touch.
yoff-value	Number of pixels offset in vertical direction. Positive direction is up; negative is down. Default is 0, which means the two edges touch.

`-slave` [`enable` | `disable`] [`internal` | `external`]
This option allows you to specify the sync source for the specified stream. `internal` indicates that the sync source is the other stream of this device. `external` indicates the sync is taken from a source outside the device. If you use `external`, you also need to use the card option `-master input`.

DEFAULTS

For a given invocation of `SUNWgfb_config`, if an option does not appear on the command line, the corresponding `OWconfig` option is not updated. It retains its previous value.

When the window system is run, if an option has never been specified through `SUNWgfb_config`, a default value is used. The option defaults are as follows:

Option	Default
–dev	/dev/fb
–file	machine
–res	none
–samples	max
–multisample	enable/dynamic
–clearpixel	255
–slave	disabled/external

–doublewide	not set
–g	2.22
-active	auto
-port	auto
-tile	ll/tt/buff (for stream a)
	lr/tt/stra (for stream b)
-offset	0/0

The default for the `-res` option of `none` means that, when the window system is run, the screen resolution will be the video mode that is currently programmed in the device. This provides compatibility for users who are used to specifying the device resolution through the PROM.

Note – Some devices (for example, GX) do not support changing the video mode defaults through software. This means that the PROM ultimately determines the default video mode.

EXAMPLES

EXAMPLE 1 Switching Resolution of a Monitor

The following example switches the monitor type to the resolution of 1280 × 1024 at 76 Hz:

```
example% fbconfig -stream a -res -SUNW_STD_1280x1024x76
```

FILES

`/usr/lib/fbconfig/SUNWgfb_config`
device special file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWgfbcf

SEE ALSO

`fbconfig`(1M), `attributes`(5)

See the `dtlogin(1)` man page in the CDE man page collection. Also useful is the `Xsun(1)` man page in the OpenWindows man page collection.

NAME

SUNWifb_config – configure the Sun Expert3D Graphics Accelerator

SYNOPSIS

```
/usr/lib/fbconfig/SUNWifb_config [-dev device-filename] [-res video-mode
    [now | try] [noconfirm | nocheck]] [-file machine | system]
    [-deflinear true | false] [-defoverlay true | false]
    [-linearorder first | last] [-overlayorder first | last]
    [-expvis enable | disable] [-slave enable | disable] [-accum
    enable | disable] [-g gamma-correction-value] [-gfile
    gamma-correction-file] [-propt] [-prconf] [-defaults] [-slave] []
    [-samples 1 | 2 | 4 | 8 | 16] [-multisample enable | disable |
    auto]

/usr/lib/fbconfig/SUNWifb_config [-propt] [prconf]

/usr/lib/fbconfig/SUNWifb_config [-help] [-res \?]
```

DESCRIPTION

`SUNWifb_config` configures the Sun Expert3D Graphics Accelerator and some of the X11 window system defaults for the graphics accelerator.

The first form of `SUNWifb_config` shown in the synopsis above stores the specified options in the `OWconfig` file. These options will be used to initialize the Sun Expert3D device the next time the window system is run on that device. Updating options in the `OWconfig` file provides persistence of these options across window system sessions and system reboots.

The second and third forms of `SUNWifb_config`, which invoke only the `-prconf`, `-propt`, `-help`, and `-res \?` options, do not update the `OWconfig` file. Additionally, for the third form of the command, all other options are ignored.

Options may be specified for only one Sun Expert3D device at a time. Specifying options for multiple Sun Expert3D devices requires multiple invocations of `SUNWifb_config`.

Only options specific to the Sun Expert3D device can be specified through `SUNWifb_config`. The normal window system options for specifying default depth, default visual class and so forth are still specified as device modifiers on the `openwin` command line (see the `Xsun`(1) manual page in the OpenWindows Desktop Reference Manual).

OPTIONS

`-dev` *device-filename*
Specifies the Sun Expert3D special file. The default is `/dev/fbs/ifb0`.

`-file machine | system`
Specifies which `OWconfig` file to update. If `machine`, the machine-specific `OWconfig` file in the `/etc/openwin` directory tree is used. If `system`, the global `OWconfig` file in the `/usr/openwin` directory tree is used. If the file does not exist, it is created.

`-res` *video-mode*
Specifies the video mode used to drive the monitor connected to the specified Sun Expert3D device.

The format of these built-in video modes is:

*width*x*height*x*rate*

where *width* is the screen width in pixels, *height* is the screen height in pixels, and *rate* is the vertical frequency of the screen refresh. The `s` suffix of `960x680x112s` and `960x680x108s` means that these are stereo video modes. The `i` suffix of `640x480x60i` and `768x575x50i` designates interlaced video timing. If absent, non-interlaced timing will be used. As a convenience, `-res` also accepts formats with `@` (at sign) in front of the refresh rate instead of `x`. For example: `1280x1024@76`. Note that some video-modes supported by the Sun Expert3D device might not be supported by the monitor. The list of video-modes supported by the Sun Expert3D device and the monitor can be obtained by running `SUNWifb_config` with the `-res \?` option (shown in the command synopsis above). A list of all possible video-modes supported on the Sun Expert3D device is shown below:

1024x768x60
1024x768x70
1024x768x75
1024x768x75
1024x768x77
1024x800x84
1152x900x66
1152x900x76
1280x800x76
1280x1024x60
1280x1024x67
1280x1024x76
1280x1024x85
1280x1024x112s (Stereo)
960x680x112s (Stereo)
960x680x108s (Stereo)
640x480x60
640x480x60i (Interlaced)
768x575x50i (Interlaced)
1440x900x76
1600x1000x66
1600x1000x76
1600x1280x76
1792x1344x75
1920x1080x72
1920x1200x70

1920x1200x75

Symbolic names

For convenience, some of the above video modes have symbolic names defined for them. Instead of the form *width* x *height* x *rate*, one of these names may be supplied as the argument to `-res`. The meaning of the symbolic name none is that when the window system is run the screen resolution will be the video mode that is currently programmed in the device.

Name	Corresponding Video Mode
svga	1024x768x60
1152	1152x900x76
1280	1280x1024x76
stereo	960x680x112s
ntsc	640x480x60i
pal	768x575x50i
none	(see text above)

The `res` option also accepts additional, optional arguments immediately following the video mode specification. Any or all of the following might be present.

`now`

If present, not only will the video mode be updated in the `OWconfig` file, but the Sun Expert3D device will be immediately programmed to display this video mode. (This is useful for changing the video mode before starting the window system).

Note that it is inadvisable to use this suboption with `SUNWifb_config` while the configured device is being used (for example, while running the window system); unpredictable results might occur. To run `SUNWifb_config` with the `now` suboption, first bring the window system down. If the `now` suboption is used within a window system session, the video mode will be changed immediately, but the width and height of the affected screen will not change until the window system is exited and reentered again. In addition, the system may not recognize changes in stereo mode. Consequently, this usage is strongly discouraged.

`noconfirm`

Using the `-res` option, the user could potentially put the system into an unusable state, a state where there is no video output. This can happen if there is ambiguity in the monitor sense codes for the particular code read. To reduce the chance of this,

the default behavior of `SUNWifb_config` is to print a warning message to this effect and to prompt the user to find out if it is okay to continue. The `noconfirm` option instructs `SUNWifb_config` to bypass this confirmation and to program the requested video mode anyway. This option is useful when `SUNWifb_config` is being run from a shell script.

`nocheck` — If present, the normal error checking based on the monitor sense code (described above) will be suspended. The video mode specified by the user will be accepted regardless of whether it is appropriate for the currently attached monitor. (This option is useful if a different monitor is to be connected to the Sun Expert3D device). Use of this option implies `noconfirm` as well.

`try` — If present, the specified video mode will be programmed on a trial basis. The user will be asked to confirm the video mode by typing **`y`** within 10 seconds. Or the user may terminate the trial before 10 seconds are up by typing any character. Any character other than **`y`** or carriage return is considered a "no" and the previous video mode will be restored and `SUNWifb_config` will not change the video mode in the `OWconfig` file (other options specified will still take effect). If a carriage return is typed, the user is prompted for a yes or no answer on whether to keep the new video mode. This option implies the `now` suboption (see the warning paragraph under the `now` suboption).

`-deflinear true | false`

The Sun Expert3D device possesses two types of visuals: linear and nonlinear. Linear visuals are gamma corrected and nonlinear visuals are not. There are two visuals that have both linear and nonlinear versions: 24-bit TrueColor and 8-bit StaticGray. If `true`, the default visual is set to the linear visual that satisfies other specified default visual selection options (specifically, the `Xsun`(1) `-defdepth` and `-defclass` options described in the OpenWindows Desktop Reference Manual). If `false`, or if there is no linear visual that satisfies the other default visual selection options, the non-linear visual specified by these other options will be chosen to be the default. This option cannot be used when the `-defoverlay` option is present, because the Sun Expert3D does not possess a linear overlay visual.

`-defoverlay true | false`

The Sun Expert3D device provides an 8-bit PseudoColor visual whose pixels are disjoint from the rest of the Sun Expert3D visuals. This is called the overlay visual. Windows created in this visual will not damage windows created in other visuals.

The converse, however, is not true. Windows created in other visuals will damage overlay windows. If the value of this option is `true`, the overlay visual will be made the default visual. If `false`, the nonoverlay visual that satisfies the other default visual selection options, such as `-defdepth` and `-defclass`, will be chosen as the default visual. See the `Xsun`(1) manual page in the OpenWindows Desktop Reference Manual. Whenever `-defoverlay true` is used, the default depth and class chosen on the `openwin` command line must be 8-bit PseudoColor. If not, a warning message will be printed and the `-defoverlay` option will be treated as `false`. This option cannot be used when the `-deflinear` option is present, because the Sun Expert3D device does not possess a linear overlay visual.

`-linearorder first | last`
If `first`, linear visuals will come before their non-linear counterparts on the X11 screen visual list for the Sun Expert3D screen. If `last`, the nonlinear visuals will come before the linear ones.

`-overlayorder first | last`
If `-first`, the depth 8 PseudoColor Overlay visual will come before the non-overlay visual on the X11 screen visual list for the Sun Expert3D screen. If `last`, the non-overlay visual will come before the overlay one.

`-expvis enable | disable`
If enabled, OpenGL Visual Expansion will be activated. Multiple instances of selected visual groups (8-bit PseudoColor, 24-bit TrueColor, and so forth) can be found in the screen visual list.

`-slave enable | disable`
If enabled, the video for this frame buffer will be synced with the video of the display which is connected to it. For applications which support it buffers will also be swapped synchronously.

`-accum enable | disable`
If enabled, frame buffer memory is allocated for accelerated accumulation buffer for windows. If disabled, software accumulation buffering will be done for windows. Accelerated accumulation buffers for pBuffers are always available as memory allows.

`-g` *gamma-correction_value*
This option allows changing the gamma correction value. All linear visuals provide gamma correction. By default the gamma correction value is 2.22. Any value less than zero is illegal. The gamma correction value is applied to the linear visual, which then has an effective gamma value of 1.0, which is the value returned by `XSolarisGetVisualGamma()`. See `XSolarisGetVisualGamma`(3) for a description of that function.This option can be used while the window system is running. Changing the gamma correction value will affect all the windows being displayed using the linear visuals.

`-gfile` *gamma-correction_file*
This option loads gamma correction table from the specified file. This file should be formatted to provide the gamma correction values for R, G and B channels on each

line. Each of these values should be in hexadecimal format and separated from each other by at least one space. Also, this file should provide 1024 such triplets. An example of this file is as follows.

```
0x00 0x00 0x00
0x01 0x01 0x01
0x02 0x02 0x02
...
...
0x3ff 0x3ff 0x3ff
```

Using this option, the gamma correction table can be loaded while the window system is running. The new gamma correction will affect all the windows being displayed using the linear visuals. Note that, when gamma correction is being done using a user-specified table, the gamma correction value is undefined. By default, the window system assumes a gamma correction value of 2.22 and loads the gamma table it creates corresponding to this value.

`-defaults`
Resets all option values to their default values.

`-propt`
Prints the current values of all Sun Expert3D options in the `OWconfig` file specified by the `-file` option for the device specified by the `-dev` option. Prints the values of options as they will be in the `OWconfig` file after the call to `SUNWifb_config` completes. This is a typical display:

```
--- OpenWindows Configuration for /dev/fbs/ifb0 ---
OWconfig: machine
Video Mode: 1280x1024x76
Accum: Disabled (do not allocate an accumulation buffer)
Default Visual: Non-Linear Normal Visual
Visual Ordering: Linear Visuals are last
                 Overlay Visuals are last
OpenGL Visual Expansion: enabled
Gamma Correction Value: 2.22
Gamma Correction Table: Available
```

`-prconf`
Prints the Sun Expert3D hardware configuration. This is a typical display:

```
--- Hardware Configuration for /dev/fbs/ifb0 ---
PROM Information: @(#)ifb.fth 1.25 99/10/12 SMI
EDID Data: Available - EDID version 1 revision 1
Monitor possible resolutions: 1024x768x60, 1024x768x70, 1024x768x75,
1152x900x66, 1152x900x76, 1280x1024x67, 1280x1024x76, 960x680x112s,
640x480x60
Current resolution setting: 1280x1024x76
```

`-help`
Prints a list of the `SUNWifb_config` command-line options, along with a brief explanation of each.

`-samples 1 | 2 | 4 | 8 | 16`
Requested number of samples to compute per display pixel. The requested number of samples per pixel will be used if `-multisample` is not disabled and resouces exist for the request.

Query the number of samples used with `-propt` (see above) or the `xglinfo` utility. The `xglinfo` utility can return the number of multisamples after you specify the option `-multisample enable` (see below).

`-multisample enable | disable | auto`
If set to `disable`, no multisample is possible. If set to `enable`, multisample is possible but is selected on a per-window basis using a library interface. If set to `auto`, all Sun OpenGL windows are rendered using multisampling.

Query the number of samples used with `-propt` (see above) or the `xglinfo` utility. The `xglinfo` utility can return the number of multisamples if `-multisample` is set to `enable`.

The `xglinfo` utility is shipped with the Sun OpenGL package, `SUNWglrt`. The man page for `xglinfo` is part of another Sun OpenGL package, `SUNWgldoc`.

DEFAULTS

For a given invocation of `SUNWifb_config` command line if an option does not appear on the command line, the corresponding `OWconfig` option is not updated; it retains its previous value. When the window system is run, if an Sun Expert3D option has never been specified via `SUNWifb_config`, a default value is used. The option defaults are as follows:

```
Option          Default
_______
-dev            /dev/fbs/ifb0
-file           machine
-res            none
-deflinear      false
-defoverlay     false
-linearorder    last
-overlayorder   last
-expvis         enable
-slave          disable
-accum          enable
-g              2.22
-samples        16
-multisample    disable
```

The default for the `-res` option of `none` means that, when the window system is run, the screen resolution will be the video mode that is currently programmed in the device. This design choice provides compatibility for users who are used to specifying the device resolution through the PROM. On some devices (for example, GX), this is the only way of specifying the video mode. This means that the PROM ultimately determines the default Sun Expert3D video mode.

EXAMPLES

EXAMPLE 1 Changing Monitor Resolution

The following example switches the monitor type to the resolution of 1280 x 1024 at 76 Hz:

```
example% /usr/lib/fbconfig/SUNWifb_config -res 1280x1024x76
```

FILES

`/dev/fbs/ifb0` device special file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWifbcf

SEE ALSO

`attributes`(5), `mmap`(2), `ifb`(7D), `fbio`(7I)

NAME

swap – swap administrative interface

SYNOPSIS

/usr/sbin/swap -a *swapname* [*swaplow*] [*swaplen*]

/usr/sbin/swap -d *swapname* [*swaplow*]

/usr/sbin/swap -l

/usr/sbin/swap -s

DESCRIPTION

The `swap` utility provides a method of adding, deleting, and monitoring the system swap areas used by the memory manager.

OPTIONS

The following options are supported:

`-a` *swapname* Add the specified swap area. This option can only be used by the super-user. *swapname* is the name of the swap file: for example, `/dev/dsk/c0t0d0s1` or a regular file. *swaplow* is the offset in 512-byte blocks into the file where the swap area should begin. *swaplen* is the desired length of the swap area in 512-byte blocks. The value of *swaplen* can not be less than `16`. For example, if *n* blocks are specified, then (*n*-1) blocks would be the actual swap length. *swaplen* must be at least one page in length. The size of a page of memory can be determined by using the `pagesize` command. See `pagesize`(1). Since the first page of a swap file is automatically skipped, and a swap file needs to be at least one page in length, the minimum size should be a multiple of 2 pagesize bytes. The size of a page of memory is machine dependent.

swaplow + *swaplen* must be less than or equal to the size of the swap file. If *swaplen* is not specified, an area will be added starting at *swaplow* and extending to the end of the designated file. If neither *swaplow* nor *swaplen* are specified, the whole file will be used except for the first page. Swap areas are normally added automatically during system startup by the `/sbin/swapadd` script. This script adds all swap areas which have been specified in the `/etc/vfstab` file; for the syntax of these specifications, see `vfstab`(4).

To use an NFS or local file-system *swapname*, you should first create a file using `mkfile`(1M). A local file-system swap file can now be added to the running system by just running the `swap -a` command. For NFS mounted swap files, the server needs to export the file. Do this by performing the following steps:

1. Add the following line to `/etc/dfs/dfstab`:

   `share -F nfs -o rw=`*clientname*`,root=`*clientname* *path-to-swap-file*

2. Run `shareall`(1M).
3. Have the client add the following lines to `/etc/vfstab`:

   server`:`*path-to-swap-file* `-` *local-path-to-swap-file*`nfs - - -`
   local-path-to-swap-file `- - swap - - -`

4. Have the client run `mount`:

   `#` `mount` *local-path-to-swap-file*

5. The client can then run `swap -a` to add the swap space:

   `#` `swap -a` *local-path-to-swap-file*

`-d` *swapname*
: Delete the specified swap area. This option can only be used by the super-user. *swapname* is the name of the swap file: for example, `/dev/dsk/c0t0d0s1` or a regular file. *swaplow* is the offset in 512-byte blocks into the swap area to be deleted. If *swaplow* is not specified, the area will be deleted starting at the second page. When the command completes, swap blocks can no longer be allocated from this area and all swap blocks previously in use in this swap area have been moved to other swap areas.

`-l`
: List the status of all the swap areas. The output has five columns:

  `path` The path name for the swap area.

  `dev` The major/minor device number in decimal if it is a block special device; zeroes otherwise.

  `swaplo` The *swaplow* value for the area in 512-byte blocks.

  `blocks` The *swaplen* value for the area in 512-byte blocks.

  `free` The number of 512-byte blocks in this area that are not currently allocated.

  The list does not include swap space in the form of physical memory because this space is not associated with a particular swap area.

  If `swap -l` is run while *swapname* is in the process of being deleted (by `swap -d`), the string INDEL will appear in a sixth column of the swap stats.

`-s`
: Print summary information about total swap space usage and availability:

  `allocated` The total amount of swap space in bytes currently allocated for use as backing store.

`reserved`	The total amount of swap space in bytes not currently allocated, but claimed by memory mappings for possible future use.
`used`	The total amount of swap space in bytes that is either allocated or reserved.
`available`	The total swap space in bytes that is currently available for future reservation and allocation.

These numbers include swap space from all configured swap areas as listed by the `-l` option, as well swap space in the form of physical memory.

USAGE

Only the first 2 Gbyte of a block device larger than 2 Gbyte in size can be used for swap in `swapfs` on a 32-bit operating system. With a 64-bit operating system, a block device larger than 2 Gbyte can be fully utilized for swap up to 2^{63} –1 bytes.

ENVIRONMENT VARIABLES

See `environ`(5) for descriptions of the following environment variables that affect the execution of `swap`: `LC_CTYPE` and `LC_MESSAGE`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`pagesize`(1), `mkfile`(1M), `shareall`(1M), `getpagesize`(3C), `vfstab`(4), `attributes`(5), `largefile`(5)

WARNINGS

No check is done to determine if a swap area being added overlaps with an existing file system.

NAME

swmtool – install, upgrade, and remove software packages

SYNOPSIS

swmtool [-d directory]

DESCRIPTION

The swmtool command invokes the admintool(1M) application, which is preselected to add or remove software. Using the swmtool command, you can add software from a product CD or hard disk to an installed system, or you can remove software from an installed system.

Once logged in, you may run swmtool to examine the packages on your local system.

Membership in the sysadmin group (gid 14) is used to restrict access to administrative tasks. Members of the sysadmin group can use swmtool to add or remove software packages. Non-members have read-only permissions (where applicable).

Help is available by using the Help button.

OPTIONS

-d *directory*

Specify the directory containing the software to be installed.

EXAMPLES

EXAMPLE 1 A sample display of using swmtool command.

The following example starts the admintool( ) application and tells it to look for software packages in the local directory /cdrom/cdrom0/s0 (the default directory for a CD when running Volume Manager).

```
example% /usr/sbin/swmtool -d /cdrom/cdrom0/s0
```

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWadmap

SEE ALSO

admintool(1M), pkgadd(1M), pkgrm(1M), attributes(5)

Solaris Advanced User's Guide

NAME

sxconfig – configure contiguous memory for the SX video subsystem

SYNOPSIS

/usr/platform/*platform-name***/sbin/sxconfig** `-c`

/usr/platform/*platform-name***/sbin/sxconfig** `-d`

/usr/platform/ *platform-name* `/sbin/sxconfig [-f | -n] [-l` *limit*`] [-s` *size*`]`

DESCRIPTION

The `sxconfig` command configures contiguous memory parameters for exclusive use by the `SX` video system on the Desktop SPARCsystems with graphics option. *platform-name* can be found using the `-i` option of `uname`(1).

After configuring the physically contiguous memory, using the various options described below, the system must be rebooted for the changes to take effect. If this command is being used to configure physically contiguous memory for the first time after the system software has been installed, then the system must be rebooted using the reconfiguration option (`-r`) of `boot`(1M).

The amount of memory to be reserved depends upon the type of application. Applications that benefit from the availability of contiguous memory are those that are written to the XGL and XIL graphics and imaging foundation library APIs.

The *Platform Notes: SPARCstation 10SX System Configuration Guide* provides more detailed information regarding how much memory to reserve for various types of graphics and imaging applications.

`sxconfig` is supported only on Desktop SPARCsystems with SX graphics option.

The interface, output, and command location are uncommitted and subject to change in future releases.

OPTIONS

The following options are supported:

`-c` Display the current configuration parameters in the driver configuration file. If the system was not rebooted after previously changing the configuration parameters, then the displayed values do not reflect the actual system setup.

`-d` Restore all configuration parameters to the default values. By default, 0 megabytes of physically contiguous memory are reserved, fragmentation is not allowed, and 32 megabytes of memory are reserved for system use.

`-f` Allow fragmentation. If no single chunk of memory of at least the requested size is found, allow the request to span multiple chunks. This flag also specifies that less than `size` megabytes of data may be reserved if there are not enough contiguous chunks available. If this flag is not specified, then the memory reserved must be exactly one chunk of the requested size for the request to succeed.

`-n` Specify that fragmentation is not allowed.

`-l` *limit*	Specify that at least `limit` megabytes of total memory must remain for system use after the contiguous memory has been reserved.
`-s` *size*	Reserve `size` megabytes of contiguous memory for exclusive use by the `SX` video subsystem.

EXAMPLES

EXAMPLE 1 Using `sxconfig`

The following example reserves 16 megabytes of contiguous memory without fragmentation and indicates 32 megabytes of memory should remain for system use after reserving the contiguous memory:

```
example# sxconfig -s 16 -l 32
```

EXAMPLE 2 Using sxconfig with Fragmentation

The following example reserves 16 megabytes of contiguous memory with fragmentation allowed and indicates 32 megabytes of memory should remain for system use after reserving the contiguous memory:

```
example# sxconfig -s 16 -f -l 32
```

EXAMPLE 3 Reporting Current Configuration Parameters

The following example reports current configuration parameters in the driver configuration file:

```
example# sxconfig -c
```

EXAMPLE 4 Restoring Configuration Parameters to Default Values

The following example restores all configuration parameters to the default values:

```
example# sxconfig -d
```

EXAMPLE 5 Disabling Fragmentation

The following example disables fragmentation:

```
example# sxconfig -n
```

EXIT STATUS

`sxconfig` returns `0` on success, and a positive integer on failure.

`1`	Permission denied. Only root can run this command.
`2`	Configuration file `sx_cmem.conf` does not exist.
`3`	Illegal option.
`4`	Illegal combination of options.

`5` Illegal argument for `-s` option. Should be an integer.

`6` Illegal argument for `-l` option. Should be an integer.

FILES

`/platform/`*platform-name*`/kernel/drv/sx_cmem`
Contiguous memory device driver

`/platform/`*platform-name*`/kernel/drv/sx_cmem.conf`
Configuration file for contiguous memory driver

`/etc/init.d/sxcmem`
Contiguous memory startup script

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWkvm

SEE ALSO

`uname`(1), `boot`(1M), `init`(1M), `attributes`(5)

Platform Notes: SPARCstation 10SX System Configuration Guide

NAME

sync – update the super block

SYNOPSIS

```
sync
```

DESCRIPTION

`sync` executes the `sync` system primitive. If the system is to be stopped, `sync` must be called to insure file system integrity. It will flush all previously unwritten system buffers out to disk, thus assuring that all file modifications up to that point will be saved. See `sync`(2) for details.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`sync`(2), `attributes`(5)

NAME

syncinit – set serial line interface operating parameters

SYNOPSIS

/usr/sbin/syncinit *device* [[*baud_rate*] | [*keyword=value*,...] | [*single-word option*]]

DESCRIPTION

The `syncinit` utility allows the user to modify some of the hardware operating modes common to synchronous serial lines. This can be useful in troubleshooting a link, or necessary to the operation of a communications package.

If run without options, `syncinit` reports the options as presently set on the port. If options are specified, the new settings are reported after they have been made.

OPTIONS

Options to `syncinit` normally take the form of a keyword, followed by an equal sign and a value. The exception is that a baud rate may be specified as a decimal integer by itself. Keywords must begin with the value shown in the options table, but may contain additional letters up to the equal sign. For example, `loop=` and `loopback=` are equivalent.

The following options are supported:

Keyword	Value	Effect
`loop`	yes	Set the port to operate in `internal loopback` mode. The receiver is electrically disconnected from the DCE receive data input and tied to the outgoing transmit data line. Transmit data is available to the DCE. The Digital Phase-Locked Loop (DPLL) may not be used as a clock source in this mode. If no other clocking options have been specified, perform the equivalent of `txc=baud` and `rxc=baud`.
	no	Disable internal loopback mode. If no other clocking options have been specified, perform the equivalent of `txc=txc` and `rxc=rxc`.
`echo`	yes	Set the port to operate in `auto-echo` mode. The transmit data output is electrically disconnected from the transmitter and tied to the receive data input. Incoming receive data is still visible. Use of this mode in combination with local loopback mode has no value, and should be rejected by the device driver. The `auto-echo` mode is useful to make a system become the endpoint of a remote loopback test.
	no	Disable `auto-echo` mode.
`nrzi`	yes	Set the port to operate with `NRZI` data encoding.
	no	Set the port to operate with `NRZ` data encoding.
`txc`	`txc`	Transmit clock source will be the `TxC` signal (pin 15).
	`rxc`	Transmit clock source will be the `RxC` signal (pin 17).

	baud	Transmit clock source will be the internal baud rate generator.
	pll	Transmit clock source will be the output of the DPLL circuit.
rxc	rxc	Receive clock source will be the RxC signal (pin 17).
	txc	Receive clock source will be the TxC signal (pin 15).
	baud	Receive clock source will be the internal baud rate generator.
	pll	Receive clock source will be the output of the DPLL circuit.
speed	*integer*	Set the baud rate to *integer* bits per second.

There are also several single-word options that set one or more paramaters at a time:

Keyword	Equivalent to Options:
external	txc=txc rxc=rxc loop=no
sender	txc=baud rxc=rxc loop=no
internal	txc=pll rxc=pll loop=no
stop	speed=0

EXAMPLES

EXAMPLE 1 Using syncinit

The following command sets the first CPU port to loop internally, using internal clocking and operating at 38400 baud:

```
example# syncinit zsh0 38400 loop=yes
device: /dev/zsh  ppa: 0
speed=38400, loopback=yes, echo=no, nrzi=no, txc=baud, rxc=baud
```

The following command sets the same port's clocking, local loopback and baud rate settings to their default values:

```
example# syncinit zsh0 stop loop=no
device: /dev/zsh  ppa: 0
speed=0, loopback=no, echo=no, nrzi=no, txc=txc, rxc=rxc
```

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

syncloop(1M), syncstat(1M), intro(2), ioctl(2), attributes(5), zsh(7D)

DIAGNOSTICS

device `missing minor device number`
The name *device* does not end in a decimal number that can be used as a minor device number.

`bad speed:` *arg*
The string *arg* that accompanied the `speed=` option could not be interpreted as a decimal integer.

`Bad arg:` *arg*
The string *arg* did not make sense as an option.

`ioctl failure code =` *errno*
An `ioctl`(2) system called failed. The meaning of the value of *errno* may be found in `intro`(2).

WARNINGS

Do not use `syncinit` on an active serial link, unless needed to resolve an error condition. Do not use this command casually or without being aware of the consequences.

NAME

syncloop – synchronous serial loopback test program

SYNOPSIS

/usr/sbin/syncloop [-cdlstv] *device*

DESCRIPTION

The `syncloop` command performs several loopback tests that are useful in exercising the various components of a serial communications link.

Before running a test, `syncloop` opens the designated port and configures it according to command line options and the specified test type. It announces the names of the devices being used to control the hardware channel, the channel number (ppa) corresponding to the *device* argument, and the parameters it has set for that channel. It then runs the loopback test in three phases.

The first phase is to listen on the port for any activity. If no activity is seen for at least four seconds, `syncloop` proceeds to the next phase. Otherwise, the user is informed that the line is active and that the test cannot proceed, and the program exits.

In the second phase, called the "first-packet" phase, `syncloop` attempts to send and receive one packet. The program will wait for up to four seconds for the returned packet. If no packets are seen after five attempts, the test fails with an excoriating message. If a packet is returned, the result is compared with the original. If the length and content do not match exactly, the test fails.

The final phase, known as the "multiple-packet" phase, attempts to send many packets through the loop. Because the program has verified the integrity of the link in the first-packet phase, the test will not fail after a particular number of timeouts. If a packet is not seen after four seconds, a message is displayed. Otherwise, a count of the number of packets received is updated on the display once per second. If it becomes obvious that the test is not receiving packets during this phase, the user may wish to stop the program manually. The number and size of the packets sent during this phase is determined by default values, or by command line options. Each returned packet is compared with its original for length and content. If a mismatch is detected, the test fails. The test completes when the required number of packets have been sent, regardless of errors.

After the multiple-packet phase has completed, the program displays a summary of the hardware event statistics for the channel that was tested. The display takes the following form:

```
CRC errors   Aborts   Overruns   Underruns   In<-Drops-> Out
        0         0          0           0   0             0
```

This is followed by an estimated line speed, which is an approximation of the bit rate of the line, based on the number of bytes sent and the actual time that it took to send them.

OPTIONS

The options for `syncloop` are described in the following table:

Option	Parameter	Default	Description
`-c`	*packet_count*	100	Specifies the number of packets to be sent in the multiple-packet phase.
`-d`	*hex_data_byte*	*random*	Specifies that each packet will be filled with bytes with the value of *hex_data_byte*.
`-l`	*packet_length*	100	Specifies the length of each packet in bytes.
`-s`	*line_speed*	9600	Bit rate in bits per second.
`-v`			Sets verbose mode. If data errors occur, the expected and received data is displayed.
`-t`	*test_type*	*none*	A number, from 1 to 4, that specifies which test to perform. The values for *test_type* are as follows: `1`: Internal loopback test. Port loopback is on. Transmit and receive clock sources are internal (baud rate generator). `2`: External loopback test. Port loopback is off. Transmit and receive clock sources are internal. Requires a loopback plug suitable to the port under test. `3`: External loopback test. Port loopback is off. Transmit and receive clock sources are external (modem). Requires that one of the local modem, the remote modem, or the remote system be set in a loopback configuration. `4`: Test using predefined parameters. User defines hardware configuration and may select port parameters using the `syncinit`(1M) command.

All numeric options except `-d` are entered as decimal numbers (for example, `-s 19200`). If you do not provide the `-t` *test_type* option, `syncloop` prompts for it.

EXAMPLES

EXAMPLE 1 A sample display of using the `syncloop` command.

In the following command `syncloop` uses a packet length of 512 bytes over the first CPU port:

```
example# syncloop -l 512 zsh0
```

In response to the above command, `syncloop` prompts you for the test option you want.

The following command performs an internal loopback test on the first CPU port, using 5000 packets and a bit rate of 56Kbps:

```
example# syncloop -t 1 -s 56000 -c 5000 zsh0
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`syncinit`(1M), `syncstat`(1M), `attributes`(5), `zsh`(7D)

DIAGNOSTICS

device `missing minor device number`
The name *device* does not end in a decimal number that can be used as a minor device number.

`invalid packet length:` *nnn*
The packet length was specified to be less than zero or greater than 4096.

`poll: nothing to read`

`poll: nothing to read or write.`
The `poll`(2) system call indicates that there is no input pending and/or that output would be blocked if attempted.

`len` *xxx* `should be` *yyy*
The packet that was sent had a length of *yyy*, but was received with a length of *xxx*.

nnn `packets lost in outbound queueing`

nnn `packets lost in inbound queueing`
A discrepancy has been found between the number of packets sent by `syncloop` and the number of packets the driver counted as transmitted, or between the number counted as received and the number read by the program.

WARNINGS

To allow its tests to run properly, as well as prevent disturbance of normal operations, `syncloop` should only be run on a port that is not being used for any other purpose at that time.

NAME

syncstat – report driver statistics from a synchronous serial link

SYNOPSIS

/usr/sbin/syncstat [-c] *device* [*interval*]

DESCRIPTION

The `syncstat` command reports the event statistics maintained by a synchronous serial device driver. The report may be a single snapshot of the accumulated totals, or a series of samples showing incremental changes. Prior to these it prints the device name being used to query a particular device driver, along with a number indicating the channel number (ppa) under control of that driver.

Event statistics are maintained by a driver for each physical channel that it supports. They are initialized to zero at the time the driver module is loaded into the system, which may be either at boot time or when one of the driver's entry points is first called.

The *device* argument is the name of the serial device as it appears in the `/dev` directory. For example, `zsh0` specifies the first on-board serial device.

The following is a breakdown of `syncstat` output:

`speed`	The line speed the device has been set to operate at. It is the user's responsibility to make this value correspond to the modem clocking speed when clocking is provided by the modem.
`ipkts`	The total number of input packets.
`opkts`	The total number of output packets.
`undrun`	The number of transmitter underrun errors.
`ovrrun`	The number of receiver overrun errors.
`abort`	The number of aborted received frames.
`crc`	The number of received frames with CRC errors.
`isize`	The average size (in bytes) of input packets.
`osize`	The average size (in bytes) of output packets.

OPTIONS

`-c` Clear the accumulated statistics for the device specified. This may be useful when it is not desirable to unload a particular driver, or when the driver is not capable of being unloaded.

interval `syncstat` samples the statistics every *interval* seconds and reports incremental changes. The output reports line utilization for input and output in place of average packet sizes. These are the relationships between bytes transferred and the baud rate, expressed as percentages. The loop repeats indefinitely, with a column heading printed every twenty lines for convenience.

EXAMPLES

EXAMPLE 1 Sample output from the `syncstat` command:

```
example# syncstat zsh0
speed ipkts opkts undrun ovrrun abort crc isize osize
9600  15716 17121   0      0      1    3   98    89

example# syncstat -c zsh0speed ipkts opkts undrun ovrrun abort crc isize osize
9600   0     0     0      0      0     0    0     0
```

In the following sample output a new line of output is generated every five seconds:

```
example# syncstat zsh0 5ipkts opkts undrun ovrrun abort crc iutil outil
12    10      0     0      0     0   5%    4%
22    60      0     0      0     0   3%    90%
36    14      0     0      0     1   51%   2%
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`syncinit`(1M), `syncloop`(1M), `attributes`(5), `zsh`(7D)

DIAGNOSTICS

`bad interval:` *arg*
The argument *arg* is expected to be an interval and could not be understood.

device `missing minor device number`
The name *device* does not end in a decimal number that can be used as a minor device number.

`baud rate not set`
The *interval* option is being used and the baud rate on the device is zero. This would cause a divide-by-zero error when computing the line utilization statistics.

WARNINGS

Underrun, overrun, frame-abort, and CRC errors have a variety of causes. Communication protocols are typically able to handle such errors and initiate recovery of the transmission in which the error occurred. Small numbers of such errors are not a significant problem for most protocols. However, because the overhead involved in recovering from a link error can be much greater than that of normal operation, high error rates can greatly degrade overall link throughput. High error rates are often caused by problems in the link hardware, such as cables, connectors, interface electronics or telephone lines. They may also be related to excessive load on the link or the supporting system.

The percentages for input and output line utilization reported when using the *interval* option may occasionally be reported as slightly greater than 100% because of inexact sampling times and differences in the accuracy between the system clock and the modem clock. If the percentage of use greatly exceeds 100%, or never exceeds 50%, then the baud rate set for the device probably does not reflect the speed of the modem.

NAME

sysdef – output system definition

SYNOPSIS

/usr/sbin/sysdef [-n *namelist*]

/usr/sbin/sysdef [-h] [-d] [-D]

DESCRIPTION

The `sysdef` utility outputs the current system definition in tabular form. It lists all hardware devices, as well as pseudo devices, system devices, loadable modules, and the values of selected kernel tunable parameters.

It generates the output by analyzing the named bootable operating system file (*namelist*) and extracting the configuration information from it.

The default system *namelist* is `/dev/kmem`.

OPTIONS

`-n` *namelist*	Specifies a *namelist* other than the default (`/dev/kmem`). The *namelist* specified must be a valid bootable operating system.
`-h`	Prints the identifier of the current host in hexadecimal. This numeric value is unique across all Sun hosts.
`-d`	The output includes the configuration of system peripherals formatted as a device tree.
`-D`	For each system peripheral in the device tree, display the name of the device driver used to manage the peripheral.

EXAMPLES

EXAMPLE 1 Sample output format

The following example displays the format of the `sysdef-d` output:

```
example%  sysdef -d
Node 'Sun 4/60', unit #0 (no driver)
      Node 'options', unit #0 (no driver)
      Node 'zs', unit #0
      Node 'zs', unit #1
      Node 'fd', unit #0
      Node 'audio', unit #0
      Node 'sbus', unit #0
             Node 'dma', unit #0
             Node 'esp', unit #0
                  Node 'st', unit #1 (no driver)
                  Node 'st', unit #0
                  Node 'sd', unit #2
                  Node 'sd', unit #1
                  Node 'sd', unit #0
             Node 'le', unit #0
             Node 'bwtwo', unit #0
      Node 'auxiliary-io', unit #0
      Node 'interrupt-enable', unit #0
      Node 'memory-error', unit #0
      Node 'counter-timer', unit #0
      Node 'eeprom', unit #0
```

FILES

`/dev/kmem`	default operating system image

ATTRIBUTES See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu (32-bit) SUNWcsxu (64-bit)

SEE ALSO `hostid`(1), `prtconf`(1M), `nlist`(3ELF), `attributes`(5)

NAME

syseventadm – sysevent event specification administration

SYNOPSIS

```
syseventadm add [-R rootdir] [-v vendor] [-p publisher] [-c class] [-s
     subclass] [-u username] path [args]

syseventadm remove [-R rootdir] [-v vendor] [-p publisher] [-c class] [-s
     subclass] [-u username] [path [args]]

syseventadm list [-R rootdir] [-v vendor] [-p publisher] [-c class] [-s
     subclass] [-u username] [path [args]]

syseventadm restart
```

DESCRIPTION

The `syseventadm` command is an administrative front-end to add, remove and list `sysevent` event handlers. `syseventadm` can only be run by root.

The `syseventadm add` command adds a handler for a `sysevent` event specified by at least one of vendor, publisher or class. If *class* is specified, it may be qualified with a *sub-class*. Only the values specified for *vendor*, *publisher*, *class* and *sub-class* when adding the handler are matched against `sysevent` events to determine if the specification matches the event and the handler should be run. *path* is the full pathname of the command to be run in response to matching events, with optional arguments (*args*). If username is specified, the command is invoked as user *username*, otherwise as root.

The `syseventadm remove` command removes handlers for matching sysevent event specifications. Event specifications may be matched by specifying at least one of *vendor*, *publisher*, *class*, *username* or *path*. If *class* is specified, it may be qualified with a *sub-class*. Any of *vendor*, *publisher*, *class*, *sub-class*, *username*, *path* or *args* not specified match the corresponding fields of all events. Handlers for all matching specifications are removed.

The `syseventadm list` command lists the handlers for matching `sysevent` event specifications using the same match criteria as the `remove` command but without the requirement that at least one of *vendor*, *publisher*, *class*, *username* or *path* be specified. With no match criteria, all specifications are listed. The list command output format is: [vendor=*vendor*] [publisher=*publisher*] [class=*class*] [subclass=*subclass*] [username=*username*] *path* [*args*] where each of *class*, *sub-class*, *vendor*, *publisher* and *username* is listed only if part of the match criteria for the listed specification.

The `syseventadm restart` command informs the `syseventd` daemon to reread the `sysevent` registry after a change has been made by adding or removing one or more `sysevent` handler specifications.

Argument Macro Substitution

The `sysevent` handling facility provides extensive macro capability for constructing the command line arguments to be executed in response to an event. Macro expansion applies only to the command line *args* specified for an event handler, with macros expanded with data from the event itself. Pre-defined macros are provided for the

event *class*, *subclass*, *publisher* and *vendor* information. Macros not matching one of the pre-defined macro names cause the attribute list attached to the event to be searched for an attribute of that name, with the value of the matching attribute substituted on the command line.

Macros are introduced by the `$` character, with the macro name being the following token separated by a SPACE or TAB character. If the macro name is embedded in text, it may be delineated by `${` and `}`. A `\` before the `$` causes macro expansion not to occur.

$*class*	The class string defining the event
$*publisher*	The publisher string defining the event
$*sequence*	The sequence number of the event.
$*subclass*	The subclass string defining the event
$*timestamp*	The timestamp of the event.
$*vendor*	The vendor string defining the event

Macro names other than those pre-defined are compared against the attribute list provided with the event. An attribute with name matching the macro name causes the value of the attribute to be substituted as ASCII text on the generated command line.

Use of a macro for which no attribute with that name is defined, or for which multiple attributes with that name are provided, cause an error and the command is not invoked.

Attributes with signed data types (`DATA_TYPE_INT16`, `DATA_TYPE_INT32` and `DATA_TYPE_INT64`) are expanded as decimal digits.

Attributes with unsigned data types (`DATA_TYPE_BYTE`, `DATA_TYPE_UINT16`, `DATA_TYPE_UINT32`, `DATA_TYPE_UINT64` and `DATA_TYPE_HTTIME`) are expanded as hexadecimal digits with a `0x` prefix.

Attributes with string data type (`DATA_TYPE_STRING`) are expanded with the string data. The data is not quoted. If if it desired that the quoted strings be generated on the command line, put quotes around the macro call in the arguments.

Array types are expanded with each element expanded as defined for that scalar type, with a space separating each element substitution.

OPTIONS

The following options are supported:

`-c` *class*	The string specifying the event class.
`-p` *publisher*	The string specifying the event publisher.
`-R` *rootdir*	Cause `syseventadm` to use an alternate root path *rootdir*.
`-s` *subclass*	The string specifying the event subclass.

`-u` *username*	The user under which the command is to be invoked.
`-v` *vendor*	The string specifying the vendor defining the event. Events defined by third-party software should use the company's stock symbol. Sun-defined events use `SUNW`.

OPERANDS

The following operands are supported:

args	Command arguments
path	Full path of command to be run in response to event

EXAMPLES

EXAMPLE 1 Adding an Event Handler

The following example adds an event handler for an event defined by vendor `SUNW`, class `EC_ENV` and sub-class `ESC_ENV_TEMP`. The command to be run is `/opt/SUNWenv/bin/ec_env_temp`, with arguments being the class name, sub-class name and pathname derived from the event attributes. The `$` characters are preceded by a backslash to circumvent shell interpretation. There is no need to restart the service after the change since the registry is maintained on `$ALTROOT`.

```
# syseventadm add -R \$ALTROOT -v SUNW -c EC_ENV -s ESC_ENV_TEMP
     /opt/SUNWenv/bin/ec_env_temp \$class \$subclass \$pathname
```

EXAMPLE 2 Removing an Event Handler

The following example removes the event handler added in Example 1.

```
# syseventadm remove -R \$ALTROOT -v SUNW -c EC_ENV -s ESC_ENV_TEMP
     /opt/SUNWenv/bin/ec_env_temp \${class} \${subclass} \${pathname}
```

EXAMPLE 3 Listing Event Handlers

The following example lists all event handlers for events of class `EC_ENV`, subclass `ESC_ENV_TEMP`, as defined by vendor `SUNW`:

```
# syseventadm list -v SUNW -c EC_ENV -s ESC_ENV_TEMP
     vendor=SUNW class=EC_ENV subclass=ESC_ENV_TEMP
     /opt/SUNWenv/bin/ec_env_temp \${class} \${subclass} \${pathname}
```

EXAMPLE 4 Listing Event Handlers

The following example lists all event handlers defined by vendor `VRTS`.

```
# syseventadm list -v VRTS
```

EXAMPLE 5 Removing Event Handlers

The following example removes all event handlers defined by vendor `VRTS`, and restarts service.

EXAMPLE 5 Removing Event Handlers *(Continued)*

```
# syseventadm remove -v VRTS
# syseventadm restart
```

EXAMPLE 6 Listing All Event Handlers Specified to Run a Command

The following example lists all event handlers specified to run the command `/opt/SUNWenv/bin/ec_env_temp`:

```
# syseventadm list /opt/SUNWenv/bin/ec_env_temp
```

EXAMPLE 7 Removing Event Handlers and Restarting Service

The following example removes all event handlers specified to run the command `/opt/SUNWenv/bin/ec_env_temp`, and restarts service:

```
# syseventadm remove /opt/SUNWenv/bin/ec_env_temp
# syseventadm restart
```

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` No matching event specification found (remove or list commands only)

`2` Incorrect command usage.

`3` Permission denied.

`4` Command failed.

`5` Out of memory

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`syseventd`(1M), `attributes`(5)

NOTES

To avoid upgrade problems, packages delivering a sysevent event handler should install the event handler by running `syseventadm` from the package's postinstall script. The event handler can then be removed by running `syseventadm` from the package's preremove script using the same arguments as when added.

NAME

syseventconfd – kernel system event command invocation daemon

SYNOPSIS

/usr/lib/sysevent/syseventconfd [-r *rootdir*]

DESCRIPTION

`syseventconfd` is the user-level daemon that invokes user-level commands in response to kernel system events received from `sysevent`d(1M).

OPTIONS

The following options are supported:

`-r` *rootdir* Cause `syseventconfd` to use an alternate root path when creating its door.The root path must match the root path used to invoke `sysevent`d.

FILES

`/etc/init.d/devfsadm`
daemon start and stop script

`/etc/rcS.d/S50devfsadm`
link to `init.d` script

`/etc/rc0.d/K83devfsadm`
link to `init.d` script

`/etc/sysevent/syseventconfd_event_service`
`syseventconfd` event service door file

`/usr/lib/sysevent/modules/sysevent_conf_mod.so`
`sysevent`d loadable module (SLM) managing `sysevent.conf` files

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`sysevent`d(1M), `attributes`(5)

NAME

syseventd – kernel system event notification daemon

SYNOPSIS

/usr/lib/sysevent/syseventd [-d *debug_level*] [-r *rootdir*]

DESCRIPTION

`syseventd` is a user-level daemon that accepts delivery of system event buffers from the kernel. Once an event buffer has been delivered to `sysevntd`, it, in turn, attempts to propagate the event to all interested end event subscribers.

Event subscribers take the form of a `sysevntd` loadable module (SLM). syseventd passes the event buffer to each of its subscribers and in return expects a notification as to the successful or unsuccessful delivery attempt.

Upon successful delivery of the event buffer to all interested event subscribers, `sysevntd` frees the event buffer from the kernel event queue.

OPTIONS

The following options are supported:

-d *debug_level* — Enable debug mode. Messages are printed to the invoking user's terminal.

-r *rootdir* — Cause `sysevntd` to use an alternate root path when creating its door and lock files. Modules continue to be loaded from the standard module directories.

EXIT STATUS

The following exit values are returned:

`0` — Successful completion.

`non-zero` — An error occurred.

FILES

`/etc/init.d/devfsadm`
daemon start and stop script

`/etc/rcS.d/S50devfsadm`
link to `init.d` script

`/etc/rc0.d/K83devfsadm`
link to `init.d` script

`/etc/sysevent/syseventd_daemon.lock`
daemon lock file

`/etc/sysevent/sysevent_door`
kernel to `sysevntd` door file

`/usr/lib/sysevent/modules`
SLM directory repository

`/usr/platform/‘uname –i‘/lib/sysevent/modules`
SLM directory repository

`/usr/platform/‘uname –m‘/lib/sysevent/modules`
SLM directory repository

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO `syseventconfd`(1M), `attributes`(5)

NAME

sysidconfig – execute system configuration applications, or define set of system configuration applications

SYNOPSIS

`sysidconfig` `[-lv] [-a` *application*`] [-b` *basedir*`] [-r` *application*`]`

DESCRIPTION

Invoked without any options, the `sysidconfig` program executes a list of applications. An application on this list is referred to as a "system configuration application." Every application on this list will be passed one command-line argument, `-c`. This flag will cause the system configuration application to perform its configuration function. Without options, `sysidconfig` should only be invoked by startup scripts, which occurs during the initial installation and during a reconfigure reboot.

All applications on the list will be executed, if possible. All activity taken by the `sysidconfig` program is logged in the `sysidconfig` log file, `/var/log/sysidconfig.log`. If one or more of the applications on the list are either not present at execution time, are not executable, or execute but return a failure code upon completion, then that information will be logged as well. Successful completion of the program can be assumed if no error message is present in the log file. Programs are executed sequentially, with only one configuration application active at a time.

Executed with the `-l`, `-a`, or `-r` options, the `sysidconfig` program allows the super-user to list the defined configuration applications, and to add items to or remove items from that list. Running `sysidconfig` with options is the only way to view or manipulate the list. Only the super-user can execute the `sysidconfig` program with options.

The `-b` and `-v` options change the behavior of `sysidconfig`, and can be used with or without the list manipulation options discussed above. The `-b` *basedir* option is used to specify a reference `root` directory other than the default, `/`. The `-v` option duplicates the log file output on `stdout`.

By default, no SPARC based applications exist on this list. However, the IA based systems are delivered with one application, `kdmconfig`(1M), on the list. `kdmconfig` is not delivered on SPARC based systems.

This application is an extension of the `sysidtool`(1M) suite of programs. It is executed during initial installation and during a reconfigure reboot, before the window system has been started. Graphical User Interface (GUI) applications will not execute successfully if they are added to the list of configuration applications via `sysidconfig -a`.

This program is referenced, but not fully described, in the `sysidtool`(1M) manual page.

OPTIONS

The valid options are:

`-a` *application* — Add the named application to the list of defined applications. When next invoked without arguments, `sysidconfig` will run

this newly added application after all previously defined applications. *application* must be a fully qualified path name that is not currently on the list of applications to execute.

`-b` *basedir* — Specify an alternate base directory (`/` is defined as the default base directory if no other is specified). The specified directory is used as the `root` directory when adding, listing, removing, or executing configuration applications. The log file where information is recorded is in `/var/log`, relative to the specified *basedir*. In the log file, the *basedir* is not noted. This means, for example, that if the super-user on a diskless client's server executes:

```
sysidconfig -b /export/root/client -a /sbin/someapp
```

then the diskless client *client* would have `/sbin/`*someapp* executed upon reconfigure reboot. The diskless client's log file would note that `/sbin/`*someapp* was added, not `/export/root/client/sbin/`*someapp*.

`-l` — List defined configuration applications. Applications will be executed one at a time, in the order shown in the list.

`-r` *application* — Remove the named application from the list of defined applications. *application* must be a fully qualified path name and it must be on the existing list of applications to execute.

`-v` — Verbose mode. This option echoes all information sent to the log file to `stdout`. Such information includes timestamp information about when the program was executed, the names of applications being executed, and results of those executions.

RETURN VALUES

The `sysidconfig` program will return `0` if it completes successfully.

When executed with the `-r` or `-a` options, error conditions or warnings will be reported on `stderr`. If the requested action completes successfully, an exit code of `0` will be returned.

ERRORS

`EPERM` The program was executed by a user other than the super-user.

`EINVAL` Option `-l`, `-a`, or `-r` was passed and the action could not be completed successfully.

FILES

`/var/log/sysidconfig.log` `sysidconfig` log file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWadmap

SEE ALSO

`sys-unconfig`(1M), `sysidtool`(1M), `attributes`(5)

IA Only

`kdmconfig`(1M)

DIAGNOSTICS

When run without options, a log of the `sysidconfig` program's activity can be found in `/var/log/sysidconfig.log`. This file contains a timestamp log of each program executed, its resulting `stderr` output, and its exit code. If an application in the list was not found or is not executable, that will also be noted.

NAME

sysidtool, sysidnet, sysidns, sysidsys, sysidroot, sysidpm – system configuration

SYNOPSIS

/usr/sbin/sysidnet

/usr/sbin/sysidns

/usr/sbin/sysidsys

/usr/sbin/sysidroot

/usr/sbin/sysidpm

DESCRIPTION

`sysidtool` is a suite of five programs that configure a new system, or one that has been unconfigured with `sys-unconfig`(1M). The `sysidtool` programs run automatically at system installation, or during the first boot after a machine has been successfully unconfigured.

These programs have no effect except at such times, and should never be run manually.

The `sysidtool` programs set up the appropriate information in the machine's configuration files, in the kernel, and on the machine's network interface. The following list shows the available commands and the information for which each of the commands lists.

`sysidnet`: network configuration	Machine's default locale. Machine's console type. Machine's host name. Machine's IP address.
`sysidns`: Name Service configuration	Name service choice: NIS, NIS+, DNS, LDAP, or none. Machine's IP subnet mask (if no NIS/NIS+ server can automatically be located on the machine's sub-network. Domain name for chosen name service. Hostname and IP address of name server(s). DNS search list (DNS name service only)
`sysidsys`: miscellaneous system configuration	Machine's IP subnet mask (if an NIS/NIS+ server was automatically located on the machine's sub-network). Machine's time zone. Date and time.
`sysidroot`: control superuser information	Machine's root password.
`sysidpm`: Power Management configuration	Auto-shutdown confirmation if the system is Energystar-V2 compliant, that is, a new system model shipped after October 1, 1995.
`sysidconfig`: host or platform-specific configuration	This command controls specification and execution of custom configuration applications that may be specified for a particular host or a particular platform. See `sysidconfig`(1M).

The `sysidtool` programs attempt to obtain system configuration information from various name service databases, for example, NIS, or from the `sysidcfg`(4) file, and you are prompted to provide the information if it cannot be found. However, you can avoid one or more of the prompts by preconfiguring the appropriate configuration information in the name service databases or in the `sysidcfg`(4) file.

To preconfigure the information in the name service databases, you must use the name service commands or the Solstice AdminSuite tools. See *Solaris 9 Installation Guide* for more details about how to preconfigure the system configuration information.

The machine's configuration information is set up in its `/etc` and `/var` files.

If a system has more than one network interface, you can use `sysidtool` to configure only the primary interface on the system. All other interfaces on the system must be configured manually.

You cannot use the name service databases or the `sysidcfg`(4) file to suppress the Power Management configuration prompt. However, you can suppress it by creating either the `/autoshutdown` or `/noautoshutdown` file before installation reboot. Accordingly, the auto-shutdown feature is silently configured. The `/autoshutdown` or `/noautoshutdown` files are removed by `sysidpm` before it exits.

FILES

`/etc/.UNCONFIGURED`	
`/etc/nodename`	
`/etc/hostname.??[0-9]`	
`/etc/default/init`	
`/etc/defaultdomain`	
`/etc/passwd`	password file. See `passwd`(4).
`/etc/shadow`	shadow password file. See `shadow`(4).
`/etc/inet/hosts`	
`/etc/inet/netmasks`	
`/etc/net/*/hosts`	
`/var/nis/NIS_COLD_START`	
`/var/yp/aliases`	
`/var/yp/binding/*/ypservers`	
`/etc/.sysIDtool.state`	
`/etc/power.conf`	Power Management configuration file. See `power.conf`(4).
`/etc/.PM_RECONFIGURE`	If this file is present during system reboot, the `sysidpm` program will be run. This file

will be removed by `sysidpm`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWadmap
	SUNWpmu

SEE ALSO

`powerd`(1M), `sys-unconfig`(1M), `sysidconfig`(1M), `passwd`(4), `power.conf`(4), `shadow` (4), `sysidcfg`(4), `attributes`(5)

Solaris 9 Installation Guide

NOTES

NIS+ might not be supported in future releases of the Solaris™ Operating Environment. Tools to aid the migration from NIS+ to LDAP are available in the Solaris 9 operating environment. For more information, visit http://www.sun.com/directory/nisplus/transition.html.

NAME

syslogd – log system messages

SYNOPSIS

/usr/sbin/syslogd [-d] [-f *configfile*] [-m *markinterval*] [-p *path*] [-t | -T]

DESCRIPTION

`syslogd` reads and forwards system messages to the appropriate log files or users, depending upon the priority of a message and the system facility from which it originates. The configuration file `/etc/syslog.conf` (see `syslog.conf`(4)) controls where messages are forwarded. `syslogd` logs a mark (timestamp) message every *markinterval* minutes (default `20`) at priority `LOG_INFO` to the facility whose name is given as `mark` in the `syslog.conf` file.

A system message consists of a single line of text, which may be prefixed with a priority code number enclosed in angle-brackets (`< >`); priorities are defined in `<sys/syslog.h>`.

`syslogd` reads from the STREAMS log driver, `/dev/log`, and from any transport provider specified in `/etc/netconfig`, `/etc/net/transport/hosts`, and `/etc/net/transport/services`.

`syslogd` reads the configuration file when it starts up, and again whenever it receives a `HUP` signal (see `signal`(3HEAD), at which time it also closes all files it has open, re-reads its configuration file, and then opens only the log files that are listed in that file. `syslogd` exits when it receives a `TERM` signal.

As it starts up, `syslogd` creates the file `/etc/syslog.pid`, if possible, containing its process identifier (PID).

If message ID generation is enabled (see `log`(7D)), each message will be preceded by an identifier in the following format: `[ID` *msgid facility*`.`*priority*`]`. *msgid* is the message's numeric identifier described in `msgid`(1M). *facility* and *priority* are described in `syslog.conf`(4). `[ID 123456 kern.notice]` is an example of an identifier when message ID generation is enabled.

If the message originated in a loadable kernel module or driver, the kernel module's name (for example, `ufs`) will be displayed instead of `unix`. See `EXAMPLES` for sample output from `syslogd` with and without message ID generation enabled.

In an effort to reduce visual clutter, message IDs are not displayed when writing to the console; message IDs are only written to the log file. See EXAMPLES.

The `/etc/default/syslogd` file contains the following default parameter settings. See `FILES`.

`LOG_FROM_REMOTE` — Specifies whether remote messages are logged. `LOG_FROM_REMOTE=NO` is equivalent to the `-t` command-line option. The default value for `LOG_FROM_REMOTE` is `YES`.

OPTIONS

The following options are supported:

`-d`	Turn on debugging. This option should only be used interactively in a root shell once the system is in multi-user mode. It should *not* be used in the system start-up scripts, as this will cause the system to hang at the point where `syslogd` is started.
`-f` *configfile*	Specify an alternate configuration file.
`-m` *markinterval*	Specify an interval, in minutes, between mark messages.
`-p` *path*	Specify an alternative log device name. The default is `/dev/log`.
`-T`	Enable the `syslogd` UDP port to turn on logging of remote messages. This is the default behavior. See FILES.
`-t`	Disable the `syslogd` UDP port to turn off logging of remote messages. See FILES.

EXAMPLES

EXAMPLE 1 `syslogd` Output Without Message ID Generation Enabled

The following example shows the output from `syslogd` when message ID generation *is not* enabled:

```
Sep 29 21:41:18 cathy unix: alloc /: file system full
```

EXAMPLE 2 `syslogd` Output with ID generation Enabled

The following example shows the output from `syslogd` when message ID generation *is* enabled. The message ID is displayed when writing to log file`/var/adm/messages`.

```
Sep 29 21:41:18 cathy ufs: [ID 845546 kern.notice]
                                    alloc /: file system full
```

EXAMPLE 3 `syslogd` Output with ID Generation Enabled

The following example shows the output from `syslogd` when message ID generation *is* enabled when writing to the console. Even though message ID is enabled, the message ID is not displayed at the console.

```
Sep 29 21:41:18 cathy ufs: alloc /: file system full
```

FILES

`/etc/syslog.conf`	Configuration file
`/etc/syslog.pid`	Process ID

`/etc/default/syslogd`	Contains default settings. You can override some of the settings by command-line options.
`/dev/log`	STREAMS log driver
`/etc/netconfig`	Transport providers available on the system
`/etc/net/transport/hosts`	Network hosts for each transport
`/etc/net/transport/services`	Network services for each transport

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`logger`(1), `msgid`(1M),`syslog`(3C), `syslog.conf`(4), `attributes`(5), `signal`(3HEAD), `log`(7D)

NOTES

The `mark` message is a system time stamp, and so it is only defined for the system on which `syslogd` is running. It can not be forwarded to other systems.

When `syslogd` receives a HUP signal, it attempts to close all log files to which it is currently logging messages. If, for some reason, one of these files does not close within a generous grace period, `syslogd` shuts down, logging an error message to the console. If the console is one of the files that did not successfully close, `syslogd` instead sends a mail message to the superuser on the current system stating that it shut down, and that the console was not responding.

Care should be taken to ensure that each window displaying messages forwarded by `syslogd` (especially console windows) is run in the system default locale (which is `syslogd`'s locale). If this advice is not followed, it is possible for a `syslog` message to alter the terminal settings for that window, possibly even allowing remote execution of arbitrary commands from that window.

NAME

sys-unconfig – undo a system's configuration

SYNOPSIS

/usr/sbin/sys-unconfig

DESCRIPTION

The `sys-unconfig` command is used to restore a system's configuration to an "as-manufactured" state, ready to be reconfigured again. The system's configuration consists of hostname, Network Information Service (NIS) domain name, timezone, IP address, IP subnet mask, and root password. This operation is the inverse of those performed by the `sysidnet`(1M), `sysidns`(1M), and `sysidsys`(1M) programs run at boot. See `sysidtool`(1M).

`sys-unconfig` does the following:

- Saves current `/etc/inet/hosts` file information in `/etc/inet/hosts.saved`.
- If the current `/etc/vfstab` file contains NFS mount entries, saves the `/etc/vfstab` file to `/etc/vfstab.orig`.
- Restores the default `/etc/inet/hosts` file.
- Removes the default hostname in `/etc/hostname.`*interface* files for all interfaces configured when this command is run. To determine which interfaces are configured, run the command '`ifconfig-a`'. The `/etc/hostname.`*interface* files corresponding to all of the interfaces listed in the resulting output, with the exception of the loopback interface (lo0), will be removed.
- Removes the default domainname in `/etc/defaultdomain`.
- Restores the timezone to PST8PDT in `/etc/TIMEZONE`.
- Disables the Network Information Service (NIS) and Network Information Service Plus (NIS+) if either NIS or NIS+ was configured.
- Removes the entries for this host in `/etc/net/*/hosts`.
- Removes the file `/etc/inet/netmasks`.
- Removes the file `/etc/defaultrouter`.
- Removes the password set for root in `/etc/shadow`.
- Removes the file `/etc/.rootkey`.
- Executes all system configuration applications. These applications are defined by prior executions of a `sysidconfig -a` *application*. (See `sysidconfig`(1M)). When `sys-unconfig` is run, all system configuration applications are passed one argument, `-u`.
- Removes the file `/etc/resolv.conf`.
- Disables LDAP by removing `/var/ldap/ldap_client_cache`, `/var/ldap/ldap_client_file`, `/var/ldap/ldap_client_cred`, and `/var/ldap/cachemgr.log`.

When `sys-unconfig` is finished, it performs a system shutdown. `sys-unconfig` is a potentially dangerous utility and can only be run by the super user.

FILES

`/etc/default/init`	process control initialization

`/etc/defaultdomain`	
`/etc/defaultrouter`	
`/etc/hostname.`*interface*	
`/etc/inet/hosts`	host name database
`/etc/inet/netmasks`	network mask database
`/etc/net/*/hosts`	
`/etc/nodename`	
`/etc/.rootkey`	super-user's secret key
`/etc/shadow`	shadow password file
`/etc/vfstab`	virtual file system table
`/var/nis/NIS_COLD_START`	
`/var/yp/binding/*/ypservers`	

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWadmap

SEE ALSO

`init`(1M), `kdmconfig`(1M), `sysidconfig`(1M), `sysidtool`(1M), `hosts`(4), `netmasks`(4), `shadow`(4), `attributes`(5)

NOTES

`sys-unconfig` is not available on diskless clients.

NAME

tapes – creates /dev entries for tape drives attached to the system

SYNOPSIS

/usr/sbin/tapes [-r *rootdir*]

DESCRIPTION

devfsadm(1M) is now the preferred command for /dev and /devices and should be used instead of tapes.

tapes creates symbolic links in the /dev/rmt directory to the actual tape device special files under the /devices directory tree. tapes searches the kernel device tree to see what tape devices are attached to the system. For each equipped tape drive, the following steps are performed:

1. The /dev/rmt directory is searched for a /dev/rmt/*n* entry that is a symbolic link to the /devices special node of the current tape drive. If one is found, this determines the logical controller number of the tape drive.
2. The rest of the special devices associated with the drive are checked, and incorrect symbolic links are removed and necessary ones added.
3. If none are found, a new logical controller number is assigned (the lowest-unused number), and new symbolic links are created for all the special devices associated with the drive.

tapes does not remove links to non-existent devices; these must be removed by hand.

tapes is run each time a reconfiguration-boot is performed, or when add_drv(1M) is executed. When invoking tapes(1M) manually, first run drvconfig(1M) to ensure /devices is consistent with the current device configuration.

Notice to Driver Writers

tapes(1M) considers all devices with the node type DDI_NT_TAPE to be tape devices; these devices must have their minor name created with a specific format. The minor name encodes operational modes for the tape device and consists of an ASCII string of the form [l,m,h,c,u][b][n].

The first character set is used to specify the tape density of the device, and are named low (l), medium (m), high (h), compressed (c), and ultra (u). These specifiers only express a relative density; it is up to the driver to assign specific meanings as needed. For example, 9 track tape devices interpret these as actual bits-per-inch densities, where l means 800 BPI, m means 1600 BPI , and h means 6250 BPI, whereas 4mm DAT tapes defines l as standard format, and m, h, c and u as compressed format. Drivers may choose to implement any or all of these format types.

During normal tape operation (non-BSD behavior), once an EOF mark has been reached, subsequent reads from the tape device return an error. An explicit IOCTL must be issued to space over the EOF mark before the next file can be read. b instructs the device to observe BSD behavior, where reading at EOF will cause the tape device to automatically space over the EOF mark and begin reading from the next file.

n or no-rewind-on-close instructs the driver to not rewind to the beginning of tape when the device is closed. Normal behavior for tape devices is to reposition to BOT when closing. See mtio(7I).

The minor number for tape devices should be created by encoding the device's instance number using the tape macro `MTMINOR` and ORing in the proper combination of density, BSD behavior, and no-rewind flags. See `mtio`(7I).

To prevent `tapes` from attempting to automatically generate links for a device, drivers must specify a private node type and refrain from using the node type string `DDI_NT_TAPE` when calling`ddi_create_minor_node`(9F).

OPTIONS

`-r` *rootdir* — Causes `tapes` to presume that the `/dev/rmt` directory tree is found under *rootdir*, not directly under `/`.

ERRORS

If `tapes` finds entries of a particular logical controller linked to different physical controllers, it prints an error message and exits without making any changes to the `/dev` directory, since it cannot determine which of the two alternative logical to physical mappings is correct. The links should be manually corrected or removed before another reconfiguration boot is performed.

EXAMPLES

EXAMPLE 1 Creating Tape Device Nodes From Within the Driver's `attach()` Function

This example demonstrates creating tape device nodes from within the `xktape` driver's `attach`(9E) function.

```
#include <sys/mtio.h>
struct tape_minor_info {
    char *minor_name;
    int   minor_mode;
};
/*
 * create all combinations of logical tapes
*/
static struct tape_minor_info example_tape[] = {
   {"",    0},                    /* default tape */
   {"l",   MT_DENSITY1},
   {"lb",  MT_DENSITY1 | MT_BSD},
   {"lbn", MT_DENSITY1 | MT_BSD | MT_NOREWIND},
   {"m",   MT_DENSITY2},
   {"mb",  MT_DENSITY2 | MT_BSD},
   {"mbn", MT_DENSITY2 | MT_BSD | MT_NOREWIND},
   {"h",   MT_DENSITY3},
   {"hb",  MT_DENSITY3 | MT_BSD},
   {"hbn", MT_DENSITY3 | MT_BSD | MT_NOREWIND},
   {"c",   MT_DENSITY4},
   {"cb",  MT_DENSITY4 | MT_BSD},
   {"cbn", MT_DENSITY4| MT_BSD | MT_NOREWIND},
   {NULL,  0},
};

int
xktapeattach(dev_info_t *dip, ddi_attach_cmd_t cmd)
{
   int instance;
   struct tape_minor_info *mdp;
      /* other stuff in attach... */
   instance = ddi_get_instance(dip);
```

EXAMPLE 1 Creating Tape Device Nodes From Within the Driver's `attach()` Function *(Continued)*

```
   for (mdp = example_tape; mdp->minor_name != NULL; mdp++) {
            ddi_create_minor_node(dip, mdp->minor_name, S_IFCHR,
                 (MTMINOR(instance) | mdp->minor_mode), DDI_NT_TAPE, 0);
 }
```

Installing the `xktape` driver on a SPARCstation 20, with the driver controlling a SCSI tape (target 4 attached to an `esp`(7D) SCSI HBA) and performing a reconfiguration-boot creates the following special files in `/devices`.

```
# ls -l /devices/iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/
crw-rw-rw-   1 root sys   33,136 Aug 29 00:02  xktape@4,0:
crw-rw-rw-   1 root sys   33,200 Aug 29 00:02  xktape@4,0:b
crw-rw-rw-   1 root sys   33,204 Aug 29 00:02  xktape@4,0:bn
crw-rw-rw-   1 root sys   33,152 Aug 29 00:02  xktape@4,0:c
crw-rw-rw-   1 root sys   33,216 Aug 29 00:02  xktape@4,0:cb
crw-rw-rw-   1 root sys   33,220 Aug 29 00:02  xktape@4,0:cbn
crw-rw-rw-   1 root sys   33,156 Aug 29 00:02  xktape@4,0:cn
crw-rw-rw-   1 root sys   33,144 Aug 29 00:02  xktape@4,0:h
crw-rw-rw-   1 root sys   33,208 Aug 29 00:02  xktape@4,0:hb
crw-rw-rw-   1 root sys   33,212 Aug 29 00:02  xktape@4,0:hbn
crw-rw-rw-   1 root sys   33,148 Aug 29 00:02  xktape@4,0:hn
crw-rw-rw-   1 root sys   33,128 Aug 29 00:02  xktape@4,0:l
crw-rw-rw-   1 root sys   33,192 Aug 29 00:02  xktape@4,0:lb
crw-rw-rw-   1 root sys   33,196 Aug 29 00:02  xktape@4,0:lbn
crw-rw-rw-   1 root sys   33,132 Aug 29 00:02  xktape@4,0:ln
crw-rw-rw-   1 root sys   33,136 Aug 29 00:02  xktape@4,0:m
crw-rw-rw-   1 root sys   33,200 Aug 29 00:02  xktape@4,0:mb
crw-rw-rw-   1 root sys   33,204 Aug 29 00:02  xktape@4,0:mbn
crw-rw-rw-   1 root sys   33,140 Aug 29 00:02  xktape@4,0:mn
crw-rw-rw-   1 root sys   33,140 Aug 29 00:02  xktape@4,0:n
```

`/dev/rmt` will contain the logical tape devices (symbolic links to tape devices in `/devices`).

```
# ls -l /dev/dsk
/dev/rmt/0    -> ../../devices/[....]/xktape@4,0:
/dev/rmt/0b   -> ../../devices/[....]/xktape@4,0:b
/dev/rmt/0bn  -> ../../devices/[....]/xktape@4,0:bn
/dev/rmt/0c   -> ../../devices/[....]/xktape@4,0:c
/dev/rmt/0cb  -> ../../devices/[....]/xktape@4,0:cb
/dev/rmt/0cbn -> ../../devices/[....]/xktape@4,0:cbn
/dev/rmt/0cn  -> ../../devices/[....]/xktape@4,0:cn
/dev/rmt/0h   -> ../../devices/[....]/xktape@4,0:h
/dev/rmt/0hb  -> ../../devices/[....]/xktape@4,0:hb
/dev/rmt/0hbn -> ../../devices/[....]/xktape@4,0:hbn
/dev/rmt/0hn  -> ../../devices/[....]/xktape@4,0:hn
/dev/rmt/0l   -> ../../devices/[....]/xktape@4,0:l
/dev/rmt/0lb  -> ../../devices/[....]/xktape@4,0:lb
/dev/rmt/0lbn -> ../../devices/[....]/xktape@4,0:lbn
/dev/rmt/0ln  -> ../../devices/[....]/xktape@4,0:ln
/dev/rmt/0m   -> ../../devices/[....]/xktape@4,0:m
/dev/rmt/0mb  -> ../../devices/[....]/xktape@4,0:mb
```

EXAMPLE 1 Creating Tape Device Nodes From Within the Driver's `attach()` Function *(Continued)*

```
/dev/rmt/0mbn -> ../../devices/[....]/xktape@4,0:mbn
/dev/rmt/0mn  -> ../../devices/[....]/xktape@4,0:mn
/dev/rmt/0n   -> ../../devices/[....]/xktape@4,0:n
```

FILES

`/dev/rmt/*` logical tape devices

`/devices/*` tape device nodes

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`add_drv`(1M), `devfsadm`(1M), `devlinks`(1M), `disks`(1M), `drvconfig`(1M), `ports`(1M), `attributes`(5), `esp`(7D), `mtio`(7I), `attach`(9E), `ddi_create_minor_node`(9F)

Writing Device Drivers

BUGS

`tapes` silently ignores malformed minor device names.

NAME

taskstat – prints ASET tasks status

SYNOPSIS

/usr/aset/util/taskstat [-d *aset_dir*]

DESCRIPTION

`taskstat` is located in the `/usr/aset/util` directory. `/usr/aset` is the default operating directory of the Automated Security Enhancement Tool (ASET). An alternative working directory can be specified by the administrators through the `aset -d` command or the `ASETDIR` environment variable. See `aset`(1M). Because `aset` dispatches its tasks to run in the background, when it returns, these tasks may or may not have completed. `taskstat` prints the status of the tasks, listing those that are completed and those that are still executing.

The ASET reports, which are located in the `/usr/aset/reports` directory (see the `-d` option), are not complete until all the tasks finish executing.

OPTIONS

`-d` *aset_dir* — Specify the working directory for ASET. By default, this directory is `/usr/aset`. With this option, the `reports` directory will be located under *aset_dir*.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWast

SEE ALSO

`aset`(1M), `attributes`(5)

System Administration Guide: Basic Administration

NAME

tcxconfig – configure the default linearity of the 24-bit TrueColor Visual for OpenWindows on a system with an S24 frame buffer (TCX)

SYNOPSIS

/usr/sbin/tcxconfig [linear | nonlinear]

DESCRIPTION

The `tcxconfig` script changes the default linearity of a 24-bit TrueColor Visual for OpenWindows on a system with an S24 frame buffer. When the S24 graphics driver for OpenWindows is installed, the default 24-bit TrueColor Visual is nonlinear. You can run `tcxconfig` with an argument that specifies the setting you want.

OpenWindows should not be running when you execute the `tcxconfig` script with an option. Start OpenWindows after `tcxconfig` has set the linearity you desire.

OPTIONS

If you specify no option, `tcxconfig` displays the current default setting.

You must become superuser before you can execute `tcxconfig` with one of the following options.

`linear` Set linear visual to be the default 24-bit TrueColor Visual. This means colors will be gamma-corrected.

`nonlinear` Set nonlinear visual to be the default 24-bit TrueColor Visual.

EXIT STATUS

The following exit values are returned:

`0` success

`1` an error has occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWtcxow

SEE ALSO

`attributes`(5)

NAME

th_define – create fault injection test harness error specifications

SYNOPSIS

```
th_define [-n name -i instance| -P path] [-a acc_types] [-r reg_number]
     [-l offset [length]] [-c count [failcount]] [-o operator [operand]]
     [-f acc_chk] [-w max_wait_period [report_interval]]

or

th_define [-n name -i instance| -P path] [-a log [acc_types]
     [-r reg_number] [-l offset [length]]] [-c count [failcount]]
     [-s collect_time] [-p policy] [-x flags] [-C comment_string]
     [-e fixup_script [args]]

or

th_define [-h]
```

DESCRIPTION

The `th_define` utility provides an interface to the `bus_ops` fault injection `bofi` device driver for defining error injection specifications (referred to as errdefs). An errdef corresponds to a specification of how to corrupt a device driver's accesses to its hardware. The command line arguments determine the precise nature of the fault to be injected. If the supplied arguments define a consistent errdef, the `th_define` process will store the errdef with the `bofi` driver and suspend itself until the criteria given by the errdef become satisfied (in practice, this will occur when the access counts go to zero).

You use the `th_manage`(1M) command with the `start` option to activate the resulting errdef. The effect of `th_manage` with the `start` option is that the `bofi` driver acts upon the errdef by matching the number of hardware accesses—specified in *count*, that are of the type specified in *acc_types*, made by instance number *instance*—of the driver whose name is *name*, (or by the driver instance specified by *path*) to the register set (or DMA handle) specified by *reg_number*, that lie within the range *offset* to *offset* + *length* from the beginning of the register set or DMA handle. It then applies *operator* and *operand* to the next *failcount* matching accesses.

If *acc_types* includes `log`, `th_define` runs in automatic test script generation mode, and a set of test scripts (written in the Korn shell) is created and placed in a sub-directory of the current directory with the name <*driver*>`.test.`<*id*> (for example, `glm.test.978177106`). A separate, executable script is generated for each access handle that matches the logging criteria. The log of accesses is placed at the top of each script as a record of the session. If the current directory is not writable, file output is written to standard output. The base name of each test file is the driver name, and the extension is a number that discriminates between different access handles. A control script (with the same name as the created test directory) is generated that will run all the test scripts sequentially.

Executing the scripts will install, and then activate, the resulting error definitions. Error definitions are activated sequentially and the driver instance under test is taken offline and brought back online before each test (refer to the `-e` option for more information). By default, logging applies to all `PIO` accesses, all interrupts, and all

DMA accesses to and from areas mapped for both reading and writing. You can constrain logging by specifying additional *acc_types*, *reg_number*, *offset* and *length*. Logging will continue for *count* matching accesses, with an optional time limit of *collect_time* seconds.

Either the `-n` or `-P` option must be provided. The other options are optional. If an option (other than `-a`) is specified multiple times, only the final value for the option is used. If an option is not specified, its associated value is set to an appropriate default, which will provide maximal error coverage as described below.

OPTIONS

The following options are available:

`-n` *name*
Specify the name of the driver to test. (String)

`-i` *instance*
Test only the specified driver instance (-1 matches all instances of driver). (Numeric)

`-P` *path*
Specify the full device path of the driver to test. (String)

`-r` *reg_number*
Test only the given register set or DMA handle (-1 matches all register sets and DMA handles). (Numeric)

`-a` *acc_types*
Only the specified access types will be matched. Valid values for the *acc_types* argument are `log`, `pio`, `pio_r`, `pio_w`, `dma`, `dma_r`, `dma_w` and `intr`. Multiple access types, separated by spaces, can be specified. The default is to match all hardware accesses.

If *acc_types* is set to `log`, logging will match all `PIO` accesses, interrupts and DMA accesses to and from areas mapped for both reading and writing. `log` can be combined with other *acc_types*, in which case the matching condition for logging will be restricted to the specified addional *acc_types*. Note that `dma_r` will match only DMA handles mapped for reading only; `dma_w` will match only DMA handles mapped for writing only; `dma` will match only DMA handles mapped for both reading and writing.

`-l` *offset* [*length*]
Constrain the range of qualifying accesses. The *offset* and *length* arguments indicate that any access of the type specified with the `-a` option, to the register set or DMA handle specified with the `-r` option, lie at least *offset* bytes into the register set or DMA handle and at most *offset* + *length* bytes into it. The default for *offset* is 0. The default for *length* is the maximum value that can be placed in an offset_t C data type (see `types.h`). Negative values are converted into unsigned quantities. Thus, `th_define -l 0 -1` is maximal.

`-c` *count* [*failcount*]
Wait for *count* number of matching accesses, then apply an operator and operand (see the `-o` option) to the next *failcount* number of matching accesses. If the access

type (see the `-a` option) includes logging, the number of logged accesses is given by *count* + *failcount* - 1. The -1 is required because the last access coincides with the first faulting access.

Note that access logging may be combined with error injection if *failcount* and *operator* are nonzero and if the access type includes logging and any of the other access types (`pio`, `dma` and `intr`) See the description of access types in the definition of the `-a` option, above.

When the *count* and *failcount* fields reach zero, the status of the errdef is reported to standard output. When all active errdefs created by the `th_define` process complete, the process exits. If *acc_types* includes `log`, *count* determines how many accesses to log. If *count* is not specified, a default value is used. If *failcount* is set in this mode, it will simply increase the number of accesses logged by a further *failcount* - 1.

`-o` *operator* `[`*operand*`]`
For qualifying PIO read and write accesses, the value read from or written to the hardware is corrupted according to the value of *operator*:

`EQ` *operand* is returned to the driver.

`OR` *operand* is bitwise ORed with the real value.

`AND` *operand* is bitwise ANDed with the real value.

`XOR` *operand* is bitwise XORed with the real value.

For PIO write accesses, the following operator is allowed:

`NO` Simply ignore the driver's attempt to write to the hardware.

Note that a driver performs PIO via the `ddi_get`*X*`()`, `ddi_put`*X*`()`, `ddi_rep_get`*X*`()` and `ddi_rep_put`*X*`()` routines (where *X* is 8, 16, 32 or 64). Accesses made using `ddi_get`*X*`()` and `ddi_put`*X*`()` are treated as a single access, whereas an access made using the `ddi_rep_*`(9F) routines are broken down into their respective number of accesses, as given by the *repcount* parameter to these DDI calls. If the access is performed via a DMA handle, *operator* and *value* are applied to every access that comprises the DMA request. If interference with interrupts has been requested then the operator may take any of the following values:

`DELAY` After *count* accesses (see the `-c` option), delay delivery of the next *failcount* number of interrupts for *operand* number of microseconds.

`LOSE` After *count* number of interrupts, fail to deliver the next *failcount* number of real interrupts to the driver.

`EXTRA` After *count* number of interrupts, start delivering *operand* number of extra interrupts for the next *failcount* number of real interrupts.

The default value for *operand* and *operator* is to corrupt the data access by flipping each bit (XOR with -1).

`-f` *acc_chk*
If the *acc_chk* parameter is set to 1 or `pio`, then the driver's calls to `ddi_check_acc_handle`(9F) return `DDI_FAILURE` when the access count goes to 1. If the *acc_chk* parameter is set to 2 or `dma`, then the driver's calls to `ddi_check_dma_handle`(9F) return `DDI_FAILURE` when the access count goes to 1.

`-w` *max_wait_period* [*report_interval*]
Constrain the period for which an error definition will remain active. The option applies only to non-logging errdefs. If an error definition remains active for *max_wait_period* seconds, the test will be aborted. If *report_interval* is set to a nonzero value, the current status of the error definition is reported to standard output every *report_interval* seconds. The default value is zero. The status of the errdef is reported in parsable format (eight fields, each separated by a colon (`:`) character, the last of which is a string enclosed by double quotes and the remaining seven fields are integers):

ft:*mt*:*ac*:*fc*:*chk*:*ec*:*s*:*"message"* which are defined as follows:

ft	The UTC time when the fault was injected.
mt	The UTC time when the driver reported the fault.
ac	The number of remaining non-faulting accesses.
fc	The number of remaining faulting accesses.
chk	The value of the *acc_chk* field of the errdef.
ec	The number of fault reports issued by the driver against this errdef (*mt* holds the time of the initial report).
s	The severity level reported by the driver.
"message"	Textual reason why the driver has reported a fault.

`-h`
Display the command usage string.

`-s` *collect_time*
If *acc_types* is given with the `-a` option and includes `log`, the errdef will log accesses for *collect_time* seconds (the default is to log until the log becomes full). Note that, if the errdef specification matches multiple driver handles, multiple logging errdefs are registered with the `bofi` driver and logging terminates when all logs become full or when *collect_time* expires or when the associated errdefs are cleared. The current state of the log can be checked with the `th_manage`(1M) command, using the `broadcast` parameter. A log can be terminated by running `th_manage`(1M) with the `clear_errdefs` option or by sending a `SIGALRM` signal to the `th_define` process. See `alarm`(2) for the semantics of `SIGALRM`.

`-p` *policy*
Applicable when the *acc_types* option includes `log`. The parameter modifies the policy used for converting from logged accesses to errdefs. All policies are inclusive:

- Use `rare` to bias error definitions toward rare accesses (default).
- Use `operator` to produce a separate error definition for each operator type (default).
- Use `common` to bias error definitions toward common accesses.
- Use `median` to bias error definitions toward median accesses.
- Use `maximal` to produce multiple error definitions for duplicate accesses.
- Use `unbiased` to create unbiased error definitions.
- Use `onebyte`, `twobyte`, `fourbyte`, or `eightbyte` to select errdefs corresponding to 1, 2, 4 or 8 byte accesses (if chosen, the `-xr` option is enforced in order to ensure that `ddi_rep_*()` calls are decomposed into *multiple single accesses*).
- Use `multibyte` to create error definitions for multibyte accesses performed using `ddi_rep_get*()` and `ddi_rep_put*()`.

Policies can be combined by adding together these options. See the NOTES section for further information.

`-x` *flags*
Applicable when the *acc_types* option includes `log`. The *flags* parameter modifies the way in which the `bofi` driver logs accesses. It is specified as a string containing any combination of the following letters:

`w`	Continuous logging (that is, the log will wrap when full).
`t`	Timestamp each log entry (access times are in seconds).
`r`	Log repeated I/O as individual accesses (for example, a `ddi_rep_get16`(9F) call which has a repcount of *N* is logged *N* times with each transaction logged as size 2 bytes. Without this option, the default logging behavior is to log this access once only, with a transaction size of twice the *repcount*).

`-C` *comment_string*
Applicable when the *acc_types* option includes `log`. It provides a comment string to be placed in any generated test scripts. The string must be enclosed in double quotes.

`-e` *fixup_script* [*args*]
Applicable when the *acc_types* option includes `log`. The output of a logging errdefs is to generate a test script for each driver access handle. Use this option to embed a command in the resulting script before the errors are injected. The generated test scripts will take an instance offline and bring it back online before injecting errors in order to bring the instance into a known fault–free state. The executable *fixup_script* will be called twice with the set of optional *args*— once just before the instance is taken offline and again after the instance has been brought online. The following variables are passed into the environment of the called executable:

`DRIVER_PATH`	Identifies the device path of the instance.
`DRIVER_INSTANCE`	Identifies the instance number of the device.

DRIVER_UNCONFIGURE	Has the value 1 when the instance is about to be taken offline.
DRIVER_CONFIGURE	Has the value 1 when the instance has just been brought online.

Typically, the executable ensures that the device under test is in a suitable state to be taken offline (unconfigured) or in a suitable state for error injection (for example configured, error free and servicing a workload). A minimal script for a network driver could be:

```
#!/bin/ksh

driver=xyznetdriver
ifnum=$driver$DRIVER_INSTANCE

if [[ $DRIVER_CONFIGURE = 1 ]]; then
    ifconfig $ifnum plumb
    ifconfig $ifnum ...
    ifworkload start $ifnum
elif [[ $DRIVER_UNCONFIGURE = 1 ]]; then
    ifworkload stop $ifnum
    ifconfig $ifnum down
    ifconfig $ifnum unplumb
fi
exit $?
```

The `-e` option must be the last option on the command line.

If the `-a log` option is selected but the `-e` option is not given, a default script is used. This script repeatedly attempts to detach and then re-attach the device instance under test.

Examples of Error Definitions

```
th_define -n foo -i 1 -a log
```

Logs all accesses to all handles used by instance 1 of the `foo` driver while running the default workload (attaching and detaching the instance). Then generates a set of test scripts to inject appropriate errdefs while running that default workload.

```
th_define -n foo -i 1 -a log pio
```

Logs PIO accesses to each PIO handle used by instance 1 of the `foo` driver while running the default workload (attaching and detaching the instance). Then generates a set of test scripts to inject appropriate errdefs while running that default workload.

```
th_define -n foo -i 1 -p onebyte median -e fixup arg -now
```

Logs all accesses to all handles used by instance 1 of the `foo` driver while running the workload defined in the fixup script `fixup` with arguments `arg` and `-now`. Then generates a set of test scripts to inject appropriate errdefs while running that workload. The resulting error definitions are requested to focus upon single byte accesses to locations that are accessed a *median* number of times with respect to frequency of access to I/O addresses.

```
th_define -n se -l 0x20 1 -a pio_r -o OR 0x4 -c 10 1000
```

Simulates a stuck serial chip command by forcing 1000 consecutive read accesses made by any instance of the `se` driver to its command status register, thereby returning status busy.

```
th_define -n foo -i 3 -r 1 -a pio_r -c 0 1 -f 1 -o OR 0x100
```

Causes 0x100 to be ORed into the next physical I/O read access from any register in register set 1 of instance 3 of the `foo` driver. Subsequent calls in the driver to `ddi_check_acc_handle()` return `DDI_FAILURE`.

```
th_define -n foo -i 3 -r 1 -a pio_r -c 0 1 -o OR 0x0
```

Causes 0x0 to be ORed into the next physical I/O read access from any register in register set 1 of instance 3 of the `foo` driver. This is of course a no-op.

```
th_define -n foo -i 3 -r 1 -l 0x8100 1 -a pio_r -c 0 10 -o EQ
0x70003
```

Causes the next ten next physical I/O reads from the register at offset 0x8100 in register set 1 of instance 3 of the `foo` driver to return 0x70003.

```
th_define -n foo -i 3 -r 1 -l 0x8100 1 -a pio_w -c 100 3 -o AND
0xffffffffffffefff
```

The next 100 physical I/O writes to the register at offset 0x8100 in register set 1 of instance 3 of the `foo` driver take place as normal. However, on each of the three subsequent accesses, the 0x1000 bit will be cleared.

```
th_define -n foo -i 3 -r 1 -l 0x8100 0x10 -a pio_r -c 0 1 -f 1 -o
XOR 7
```

Causes the bottom three bits to have their values toggled for the next physical I/O read access to registers with offsets in the range 0x8100 to 0x8110 in register set 1 of instance 3 of the `foo` driver. Subsequent calls in the driver to `ddi_check_acc_handle()` return `DDI_FAILURE`.

```
th_define -n foo -i 3 -a pio_w -c 0 1 -o NO 0
```

Prevents the next physical I/O write access to any register in any register set of instance 3 of the `foo` driver from going out on the bus.

```
th_define -n foo -i 3 -l 0 8192 -a dma_r -c 0 1 -o OR 7
```

Causes 0x7 to be ORed into each long long in the first 8192 bytes of the next DMA read, using any DMA handle for instance 3 of the `foo` driver.

```
th_define -n foo -i 3 -r 2 -l 0 8 -a dma_r -c 0 1 -o OR
0x7070707070707070
```

Causes 0x70 to be ORed into each byte of the first long long of the next DMA read, using the DMA handle with sequential allocation number 2 for instance 3 of the `foo` driver.

```
th_define -n foo -i 3 -l 256 256 -a dma_w -c 0 1 -f 2 -o OR 7
```

Causes 0x7 to be ORed into each long long in the range from offset 256 to offset 512 of the next DMA write, using any DMA handle for instance 3 of the `foo` driver. Subsequent calls in the driver to `ddi_check_dma_handle()` return `DDI_FAILURE`.

```
th_define -n foo -i 3 -r 0 -l 0 8 -a dma_w -c 100 3 -o AND
0xfffffffffffefff
```

The next 100 DMA writes using the DMA handle with sequential allocation number 0 for instance 3 of the `foo` driver take place as normal. However, on each of the three subsequent accesses, the 0x1000 bit will be cleared in the first long long of the transfer.

```
th_define -n foo -i 3 -a intr -c 0 6 -o LOSE 0
```

Causes the next six interrupts for instance 3 of the `foo` driver to be lost.

```
th_define -n foo -i 3 -a intr -c 30 1 -o EXTRA 10
```

When the thirty-first subsequent interrupt for instance 3 of the `foo` driver occurs, a further ten interrupts are also generated.

```
th_define -n foo -i 3 -a intr -c 0 1 -o DELAY 1024
```

Causes the next interrupt for instance 3 of the `foo` driver to be delayed by 1024 microseconds.

NOTES

The policy option in the `th_define -p` syntax determines how a set of logged accesses will be converted into the set of error definitions. Each logged access will be matched against the chosen policies to determine whether an error definition should be created based on the access.

Any number of policy options can be combined to modify the generated error definitions.

Bytewise Policies

These select particular I/O transfer sizes. Specifing a byte policy will exclude other byte policies that have not been chosen. If none of the byte type policies is selected, all transfer sizes are treated equally. Otherwise, only those specified transfer sizes will be selected.

`onebyte`	Create errdefs for one byte accesses (`ddi_get8()`)
`twobyte`	Create errdefs for two byte accesses (`ddi_get16()`)
`fourbyte`	Create errdefs for four byte accesses (`ddi_get32()`)
`eightbyte`	Create errdefs for eight byte accesses (`ddi_get64()`)
`multibyte`	Create errdefs for repeated byte accesses (`ddi_rep_get*()`)

Frequency of Access Policies

The frequency of access to a location is determined according to the access type, location and transfer size (for example, a two-byte read access to address A is considered distinct from a four-byte read access to address A). The algorithm is to count the number of accesses (of a given type and size) to a given location, and find the locations that were most and least accessed (let *maxa* and *mina* be the number of times these locations were accessed, and *mean* the total number of accesses divided by total number of locations that were accessed). Then a rare access is a location that was accessed less than

(mean - mina) / 3 + mina

times. Similarly for the definition of common accesses:

maxa - (maxa - mean) / 3

A location whose access patterns lies within these cutoffs is regarded as a location that is accessed with median frequency.

`rare`	Create errdefs for locations that are rarely accessed.
`common`	Create errdefs for locations that are commonly accessed.
`median`	Create errdefs for locations that are accessed a median frequency.

Policies for Minimizing errdefs

If a transaction is duplicated, either a single or multiple errdefs will be written to the test scripts, depending upon the following two policies:

`maximal`	Create multiple errdefs for locations that are repeatedly accessed.
`unbiased`	Create a single errdef for locations that are repeatedly accessed.
`operators`	For each location, a default operator and operand is typically applied. For maximal test coverage, this default may be modified using the `operators` policy so that a separate errdef is created for each of the possible corruption operators.

SEE ALSO

`kill`(1), `th_manage`(1M), `alarm`(2), `ddi_check_acc_handle`(9F), `ddi_check_dma_handle`(9F)

NAME

th_manage – manage the fault injection test harness

SYNOPSIS

th_manage *name instance command*

th_manage *path command*

DESCRIPTION

th_manage applies the action specified by *command* to the instance specified by *instance* of the driver specified by *name* (or the driver instance specified by *path*). The driver instance must be running fault injection specifications (errdefs) defined by th_define(1M).

th_manage supports several commands that operate on the driver instance specified by *name* and *instance* (or *path*). The commands are:

broadcast	Awaken all th_define processes, causing them to display their current status and exit if the errdef is now defunct (that is, if *count*, *failcount* , and *acc_chk* are all zero).
clear_acc_chk	Awaken all th_define processes. If *count* and *failcount* are already zero, then set *acc_chk* to zero, so that th_define exits once it has displayed its status.
clear_errdefs	Awaken all th_define processes. *count*, *failcount* and *acc_chk* are all set to zero so that all th_define commands exit once they have displayed their status.
clear_errors	Awaken all th_define processes. If *count* is already zero, set *failcount* and *acc_chk* to zero, so that th_define exits once it has displayed its status.
get_handles	List all the access handles.
start	Begin or resume execution of all errdefs.
stop	Suspend all errdefs for this *name* and *instance* (or *path).*

EXAMPLES

EXAMPLE 1 Useful Commands

To begin the tests, enter:

```
# th_manage foo 0 start
```

To check the status of the errdefs, enter:

```
# th_manage foo 0 broadcast
```

This causes each th_define process to print out its current status.

If the driver has reported a fatal error, you can take the driver offline using libdevice, clear the error condition by entering:

```
# th_manage foo 0 clear_acc_chk
```

or

EXAMPLE 1 Useful Commands *(Continued)*

```
# th_manage foo 0 clear_errors
```

and bring the driver online again using `libdevice`.

To terminate testing, enter:

```
# th_manage foo 0 clear_errdefs
```

SEE ALSO `th_define`(1M)

NAME

tic – terminfo compiler

SYNOPSIS

tic [-v [*n*]] [-c] *file*

DESCRIPTION

The command tic translates a terminfo file from the source format into the compiled format. The results are placed in the directory /usr/share/lib/terminfo. The compiled format is necessary for use with the library routines in curses(3CURSES).

If the environment variable TERMINFO is set, the compiled results are placed there instead of /usr/share/lib/terminfo.

Total compiled entries cannot exceed 4096 bytes. The name field cannot exceed 128 bytes. Terminal names exceeding 14 characters will be truncated to 14 characters and a warning message will be printed.

OPTIONS

The following options are supported:

-c Specifies to check only file for errors. Errors in use= links are not detected.

-v[*n*] Specify that (verbose) output be written to standard error trace information showing tic's progress. The optional integer *n* is a number from 1 to 10, indicating the desired level of detail of information. If *n* is omitted, the default level is 1. If *n* is specified and greater than 1, the level of detail is increased.

OPERANDS

file Contains one or more terminfo terminal descriptions in source format [see terminfo(4)]. Each description in the file describes the capabilities of a particular terminal. When a use=*entry-name* field is discovered in a terminal entry currently being compiled, tic reads in the binary from /usr/share/lib/terminfo to complete the entry. (Entries created from file will be used first. If the environment variable TERMINFO is set, that directory is searched instead of /usr/share/lib/terminfo.) tic duplicates the capabilities in *entry-name* for the current entry, with the exception of those capabilities that are explicitly defined in the current entry.

FILES

/usr/share/lib/terminfo/?/* Compiled terminal description database

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

captoinfo(1M), infocmp(1M), curses(3CURSES), terminfo(4), attributes(5)

NOTES

When an entry, for example, entry_name_1, contains a use=*entry_name_2* field, any canceled capabilities in *entry_name_2* must also appear in entry_name_1 before use= for these capabilities to be canceled in entry_name_1.

NAME

traceroute – print the route packets take to network host

SYNOPSIS

traceroute [-adFIlnSvx] [-A *addr_family*] [-c *traffic_class*] [-f *first_hop*] [-g *gateway* [-g *gateway...*] | -r] [-i *iface*] [-L *flow_label*] [-m *max_hop*] [-P *pause_sec*] [-p *port*] [-Q *max_timeout*] [-q *nqueries*] [-s *src_addr*] [-t *tos*] [-w *wait_time*] host [*packetlen*]

DESCRIPTION

The Internet is a large and complex aggregation of network hardware, connected by gateways. Tracking the route a packet follows can be difficult. The utility `traceroute` traces the route that an IP packet follows to another internet host.

The `traceroute` utility utilizes the both the IPv4 and IPv6 protocols. Use the `-A` option to override the default behavior. `traceroute` uses the IPv4 protocol *ttl* (time to live) field or the IPv6 field *hop limit*. It attempts to elicit an ICMP or ICMP6 `TIME_EXCEEDED` response from each *gateway* along the path, and a `PORT_UNREACHABLE`(or `ECHO_REPLY` if `-I` is used) response from the destination host. It starts by sending probes with a *ttl* or *hop limit* of `1` and increases by `1` until it either gets to the host, or it hits the maximum *max_hop*. The default maximum *max_hop* is 30 hops, but this can be set by the `-m` option.

Three probes are sent at each *ttl* (*hop limit*) setting, and a line is printed showing the *ttl* (*hop limit*), the hostname and the address of the gateway, and the *rtt* (round trip time) of each probe. The number of probes may be specifically set using the `-q` option. If the probe answers come from different gateways, the hostname and the address of each responding system will be printed. If there is no response within a 5 second timeout interval, a "`*`" is printed for that probe. The `-w` option may be used to set the timeout interval. Other possible annotations that may appear after the time are:

`!` the *ttl* (*hop limit*) value in the received packet is `<= 1`.

`!H` host unreachable.

`!X` communication administratively prohibited.

`<!N>` ICMP (ICMP6) unreachable code N.

The following annotations appear only for IPv4:

`!F` fragmentation needed. This should never occur. If this is seen, the associated gateway is broken.

`!N` network unreachable.

`!P` protocol unreachable.

`!S` source route failed. This should never occur. If this is seen, the associated gateway is broken.

`!T` unreachable for the specified tos (type-of-service).

`!U` source host isolated or precedence problem.

The following annotations appear only for IPv6:

`!A` host unreachable for a reason other than lack of an entry in the routing table.

`!B` packet too big.

`!E` destination is not a neighbor.

`!R` unrecognized next header.

If almost all the probes result in some kind of unreachable code, then `traceroute` gives up and exits.

The destination *host* is not supposed to process the UDP probe packets, so the destination *port* default is set to an unlikely value. However, if some application on the destination is using that value, the value of *port* can be changed with the `-p` option.

The only mandatory parameter is the destination *host* name or IP number. The default probe datagram length is 40 bytes (60 bytes for IPv6), but this may be increased by specifying a packet length (in bytes) after the destination *host* name.

All integer arguments to `traceroute` can be specified in either decimal or hexadecimal notation. For example, *packetlen* can be specified either as `256` or `0x100`.

OPTIONS

`-A` *addr_family*

Specify the address family of the target host. *addr_family* can be either `inet` or `inet6`. Address family determines which protocol to use. For an argument of `inet`, IPv4 is used. For `inet6`, IPv6 is used.

By default, if the name of a host is provided, not the literal IP address, and a valid IPv6 address exists in the name service database, `traceroute` will use this address. Otherwise, if the name service database contains an IPv4 address, it will try the IPv4 address.

Specify the address family `inet` or `inet6` to override the default behavior. If the argument specified is `inet`, `traceroute` will use the IPv4 address associated with the hostname. If none exists, `traceroute` will state that the host is unknown and exit. It will not try to determine if an IPv6 address exists in the name service database.

If the specified argument is `inet6`, `traceroute` will use the IPv6 address that is associated with the hostname. If none exists, `traceroute` will state that the host is unknown and exit.

`-a`

Probe all of the addresses of a multi-homed destination. The output looks like `traceroute` has been run once for each IP address of the destination. If

	this option is used together with `-A`, `traceroute` probes only the addresses that are of the specified address family. While probing one of the addresses of the destination, user can skip to the next address by sending a `SIGINT`, or exit `traceroute` by sending a `SIGQUIT` signal. See `signal`(3C)
`-c` *traffic_class*	Specify the traffic class of probe packets. The value must be an integer in the range from 0 to 255. Gateways along the path may route the probe packet differently depending upon the value of *traffic_class* set in the probe packet. This option is valid only on IPv6.
`-d`	Set the `SO_DEBUG` socket option.
`-F`	Set the "don't fragment" bit. This option is valid only on IPv4.
`-f` *first_hop*	Set the starting *ttl* (*hop limit*) value to *first_hop*, to override the default value `1`. `traceroute` skips processing for those intermediate gateways which are less than *first_hop* hops away.
`-g` *gateway*	Specify a loose source route *gateway*. The user can specify more than one *gateway* by using `-g` for each gateway. The maximum number of gateways is 8 for IPv4 and 127 for IPv6. Note that some factors such as the link MTU can further limit the number of gateways for IPv6. This option cannot be used with the `-r` option.
`-I`	Use ICMP (ICMP6) `ECHO` instead of UDP datagrams.
`-i` *iface*	For IPv4, this option specifies a network interface to obtain the source IP address. This is normally only useful on a multi-homed host. The `-s` option is also another way to do this. For IPv6, it specifies the network interface on which probe packets are transmitted. The argument can be either an interface index, for example, `1`, `2`, or an interface name, for example, `le0`, `hme0`.
`-L` *flow_label*	Specify the flow label of probe packets. The value must be an integer in the range from 0 to 1048575. This option is valid only on IPv6.
`-l`	Print the value of the *ttl* (*hop limit*) field in each packet received.
`-m` *max_hop*	Set the maximum *ttl* (*hop limit*) used in outgoing probe packets. The default is 30 hops, which is the same default used for TCP connections.

`-n` — Print hop addresses numerically rather than symbolically and numerically. This saves a nameserver address-to-name lookup for each gateway found on the path.

`-P` *pause_sec* — Specify a delay, in seconds, to pause between probe packets. This may be necessary if the final destination does not accept undeliverable packets in bursts. By default, `traceroute` sends the next probe as soon as it has received a reply. Note that *pause_sec* is a real number.

`-p` *port* — Set the base UDP *port* number used in probes.The default is 33434. `traceroute` hopes that nothing is listening on UDP *ports* `(base+(nhops-1)*nqueries)` to `(base+(nhops*nqueries)-1)` at the destination host, so that an ICMP (ICMP6) `PORT_UNREACHABLE` message will be returned to terminate the route tracing. If something is listening on a *port* in the default range, this option can be used to select an unused *port* range. `nhops` is defined as the number of hops between the source and the destination.

`-Q` *max_timeout* — Stop probing this hop after *max_timeout* consecutive timeouts are detected. The default value is 5. Useful in combination with the `-q` option if you have specified a large *nqueries* probe count.

`-q` *nqueries* — Set the desired number of probe queries. The default is 3.

`-r` — Bypass the normal routing tables and send directly to a host on an attached network. If the host is not on a directly-attached network, an error is returned. This option can be used to send probes to a local host through an interface that has been dropped by the router deamon. See `in.routed`(1M). You cannot use this option if the `-g` option is used.

`-s` *src_addr* — Use the following address, which usually is given as a literal IP address, not a hostname, as the source address in outgoing probe packets. On multi-homed hosts, those with more than one IP address, this option can be used to force the source address to be something other than the IP address `traceroute` picks by default. If the IP address is not one of this machine's interface addresses, an error is returned and nothing is sent. For IPv4, when used together with the `-i` option, the given IP address should be configured on the specified

interface. Otherwise, an error will be returned. In the case of IPv6, the interface name and the source address do not have to match.

`-t` *tos* — Set the *tos*(type-of-service) in probe packets to the specified value. The default is zero. The value must be an integer in the range from 0 to 255. Gateways along the path may route the probe packet differently depending upon the *tos* value set in the probe packet. This option is valid only on IPv4.

`-v` — Verbose output. For each hop, the size and the destination of the response packets is displayed. Also ICMP (ICMP6) packets received other than `TIME_EXCEEDED` and `UNREACHABLE` are listed as well.

`-w` *waittime* — Set the time, in seconds, to wait for a response to a probe. The default is 5 seconds.

`-x` — Prevent `traceroute` from calculating checksums. Note that checksums are usually required for the last hop when using ICMP `ECHO` probes. This option is valid only on IPv4. See the `-I` option.

OPERANDS

The following operands are supported:

host — The network host.

EXAMPLES

EXAMPLE 1 Sample Output From the `traceroute` Utility

Some sample output from the `traceroute` utility might be:

```
istanbul% traceroute london
traceroute: Warning: london has multiple addresses;  \
  using 4::114:a00:20ff:ab3d:83ed
traceroute: Warning: Multiple interfaces found; \
  using 4::56:a00:20ff:fe93:8dde @ le0:2
traceroute to london (4::114:a00:20ff:ab3d:83ed), 30 hops max, \
  60 byte packets
1  frbldg7c-86 (4::56:a00:20ff:fe1f:65a1)  1.786 ms  1.544 ms  1.719 ms
2  frbldg7b-77 (4::255:0:0:c0a8:517)  2.587 ms 3.001 ms  2.988 ms
3  london (4::114:a00:20ff:ab3d:83ed)  3.122 ms  2.744 ms  3.356 ms
```

The target host, london, has both IPv4 and IPv6 addresses in the name service database. According to the default behavior, `traceroute` uses IPv6 address of the destination host.

EXAMPLE 2 Using the `traceroute` Utility For a Host Which has Only IPv4 Addresses

In the following examples, `traceroute` is tracking the route to host sanfrancisco, which has only IPv4 addresses in the name service database. Therefore traceroute uses only IPv4 addresses. The following shows the 7-hop path that a packet would follow from the host istanbul to the host sanfrancisco.

EXAMPLE 2 Using the `traceroute` Utility For a Host Which has Only IPv4 Addresses *(Continued)*

```
istanbul% traceroute sanfrancisco
traceroute: Warning: Multiple interfaces found; using 172.31.86.247 @ le0
traceroute to sanfrancisco (172.29.64.39), 30 hops max, 40 byte packets
1  frbldg7c-86 (172.31.86.1)  1.516 ms  1.283 ms  1.362 ms
2  bldg1a-001 (172.31.1.211)  2.277 ms  1.773 ms  2.186 ms
3  bldg4-bldg1 (172.30.4.42)  1.978 ms  1.986 ms  13.996 ms
4  bldg6-bldg4 (172.30.4.49)  2.655 ms  3.042 ms  2.344 ms
5  ferbldg11a-001 (172.29.1.236)  2.636 ms  3.432 ms  3.830 ms
6  frbldg12b-153 (172.29.153.72)  3.452 ms  3.146 ms  2.962 ms
7  sanfrancisco (172.29.64.39)  3.430 ms  3.312 ms  3.451 ms
```

EXAMPLE 3 Using the `traceroute` Utility With Source Routing

The following example shows the path of a packet that goes from istanbul to sanfrancisco through the hosts cairo and paris, as specified by the `-g` option. The `-I` option makes `traceroute` send ICMP `ECHO` probes to the host sanfrancisco. The `-i` options sets the source address to the IP address configured on the interface `qe0`.

```
istanbul% traceroute -g cairo -g paris -i qe0 -q 1 -I sanfrancisco
traceroute to sanfrancisco (172.29.64.39), 30 hops max, 56 byte packets
1  frbldg7c-86 (172.31.86.1)  2.012 ms
2  flrbldg7u (172.31.17.131)  4.960 ms
3  cairo (192.168.163.175) 4.894 ms
4  flrbldg7u (172.31.17.131)  3.475 ms
5  frbldg7c-017 (172.31.17.83)  4.126 ms
6  paris (172.31.86.31) 4.086 ms
7  frbldg7b-82 (172.31.82.1)  6.454 ms
8  bldg1a-001 (172.31.1.211)  6.541 ms
9  bldg6-bldg4 (172.30.4.49)  6.518 ms
10  ferbldg11a-001 (172.29.1.236)  9.108 ms
11  frbldg12b-153 (172.29.153.72)  9.634 ms
12  sanfrancisco (172.29.64.39)  14.631 ms
```

EXIT STATUS

The following exit values are returned:

`0` Successful operation.

`>0` An error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`netstat`(1M), `signal`(3C), `ping`(1M), `attributes`(5)

WARNINGS This utility is intended for use in network testing, measurement and management. It should be used primarily for manual fault isolation. Because of the load it could impose on the network, it is unwise to use `traceroute`(1M) during normal operations or from automated scripts.

NAME

trapstat – report trap statistics

SYNOPSIS

/usr/platform/ *platform-name* /sbin/trapstat [-t | -T | -e *entry*] [-C *processor_set_id* | -c *cpulist*] [-P] [-a] [-r *rate*] [[*interval* [*count*]] | *command* | [*args*]]

/usr/platform/ *platform-name* /sbin/trapstat -l

DESCRIPTION

The `trapstat` utility gathers and displays run-time trap statistics on UltraSPARC-based systems. The default output is a table of trap types and CPU IDs, with each row of the table denoting a trap type and each column of the table denoting a CPU. If standard output is a terminal, the table contains as many columns of data as can fit within the terminal width; if standard output is not a terminal, the table contains at most six columns of data. By default, data is gathered and and displayed for all CPUs; if the data cannot fit in a single table, it is printed across multiple tables. The set of CPUs for which data is gathered and displayed can be optionally specified with the `-c` or `-C` option.

Unless the `-r` option or the `-a` option is specified, the value displayed in each entry of the table corresponds to the number of traps per second. If the `-r` option is specified, the value corresponds to the number of traps over the interval implied by the specified sampling rate; if the `-a` option is specified, the value corresponds to the accumulated number of traps since the invocation of `trapstat`.

By default, `trapstat` displays data once per second, and runs indefinitely; both of these behaviors can be optionally controlled with the *interval* and *count* parameters, respectively. The *interval* is specified in seconds; the *count* indicates the number of intervals to be executed before exiting. Alternatively, `command` can be specified, in which case `trapstat` executes the provided command and continues to run until the command exits. A positive integer is assumed to be an *interval*; if the desired *command* cannot be distinguished from an integer, the full path of *command* must be specified.

UltraSPARC I, II and III handle translation lookaside buffer (TLB) misses by trapping to the operating system. TLB miss traps can be a significant component of overall system performance for some workloads; the `-t` option provides in-depth information on these traps. When run with this option, `trapstat` displays both the rate of TLB miss traps and the percentage of time spent processing those traps. Additionally, TLB misses that hit in the translation storage buffer (TSB) are differentiated from TLB misses that further miss in the TSB. (The TSB is a software structure used as a translation entry cache to allow the TLB to be quickly filled; it is discussed in detail in the *UltraSPARC I&II User's Manual*.) The TLB and TSB miss information is further broken down into user- and kernel-mode misses.

Workloads with working sets that exceed the TLB reach may spend a significant amount of time missing in the TLB. To accommodate such workloads, the operating system supports multiple page sizes: larger page sizes increase the effective TLB reach and thereby reduce the number of TLB misses. To provide insight into the relationship

between page size and TLB miss rate, `trapstat` optionally provides in-depth TLB miss information broken down by page size using the `-T` option. The information provided by the `-T` option is a superset of that provided by the `-t` option; only one of `-t` and `-T` can be specified.

OPTIONS

The following options are supported:

`-a`
: Displays the number of traps as accumulating, monotonically increasing values instead of per-second or per-interval rates.

`-c` *cpulist*
: Enables `trapstat` only on the CPUs specified by *cpulist*.

  cpulist can be a single processor ID (for example, `4`), a range of processor IDs (for example, `4-6`), or a comma separated list of processor IDs or processor ID ranges (for example, `4,5,6` or `4,6-8`).

`-C` *processor_set_id*
: Enables `trapstat` only on the CPUs in the processor set specified by *processor_set_id*.

  `trapstat` modifies its output to always reflect the CPUs in the specified processor set. If a CPU is added to the set, `trapstat` modifies its output to include the added CPU; if a CPU is removed from the set, `trapstat` modifies its output to exclude the removed CPU. At most one processor set can be specified.

`-e` *entrylist*
: Enables `trapstat` only for the trap table entry or entries specified by *entrylist*. A trap table entry can be specified by trap number or by trap name (for example, the level–10 trap can be specified as `74`, `0x4A`, `0x4a`, or `level-10`).

  entrylist can be a single trap table entry or a comma separated list of trap table entries. If the specified trap table entry is not valid, `trapstat` prints a table of all valid trap table entries and values. A list of valid trap table entries is also found in *The SPARC Architecture Manual, Version 9* and the *Sun Microelectronics UltraSPARC I&II User's Manual.* If the parsable option (`-P`) is specified in addition to the `-e` option, the format of the data is as follows:

Field	Contents
1	Timestamp (nanoseconds since start)

Field	Contents
2	CPU ID
3	Trap number (in hexadecimal)
4	Trap name
5	Trap rate per interval

Each field is separated with whitespace. If the format is modified, it will be modified by adding potentially new fields beginning with field 6; exant fields will remain unchanged.

`-l` Lists trap table entries. By default, a table is displayed containing all valid trap numbers, their names and a brief description. The trap name is used in both the default output and in the *entrylist* parameter for the `-e` argument. If the parsable option (`-P`) is specified in addition to the `-l` option, the format of the data is as follows:

Field	Contents
1	Trap number in hexadecimal
2	Trap number in decimal
3	Trap name
Remaining	Trap description

`-P` Generates parsable output. When run without other data gathering modifying options (that is, `-e`, `-t` or `-T`), `trapstat`'s the parsable output has the following format:

Field	Contents
1	Timestamp (nanoseconds since start)
2	CPU ID
3	Trap number (in hexadecimal)

Field	Contents
4	Trap name
5	Trap rate per interval

Each field is separated with whitespace. If the format is modified, it will be modified by adding potentially new fields beginning with field 6; extant fields will remain unchanged.

`-r` *rate*

Explicitly sets the sampling rate to be *rate* samples per second. If this option is specified, `trapstat`'s output changes from a traps-per-second to traps-per-sampling-interval.

`-t`

Enables TLB statistics.

A table is displayed with four principal columns of data: *itlb-miss*, *itsb-miss*, *dtlb-miss*, and *dtsb-miss*. The columns contain both the rate of the corresponding event and the percentage of CPU time spent processing the event. The percentage of CPU time is given only in terms of a single CPU. The rows of the table correspond to CPUs, with each CPU consuming two rows: one row for user-mode events (denoted with `u`) and one row for kernel-mode events (denoted with `k`). For each row, the percentage of CPU time is totalled and displayed in the rightmost column. The CPUs are delineated with a solid line. If the parsable option (`-P`) is specified in addition to the `-t` option, the format of the data is as follows:

Field	Contents
1	Timestamp (nanoseconds since start)
2	CPU ID
3	Mode (`k` denotes kernel, `u` denotes user)
4	I-TLB misses
5	Percentage of time in I-TLB miss handler
6	I-TSB misses

Field	Contents
7	Percentage of time in I-TSB miss handler
8	D-TLB misses
9	Percentage of time in D-TLB miss handler
10	D-TSB misses
11	Percentage of time in D-TSB miss handler

Each field is separated with whitespace. If the format is modified, it will be modified by adding potentially new fields beginning with field 12; extant fields will remain unchanged.

`-T` Enables TLB statistics, with page size information. As with the `-t` option, a table is displayed with four principal columns of data: *itlb-miss*, *itsb-miss*, *dtlb-miss*, and *dtsb-miss*. The columns contain both the absolute number of the corresponding event, and the percentage of CPU time spent processing the event. The percentage of CPU time is given only in terms of a single CPU. The rows of the table correspond to CPUs, with each CPU consuming two sets of rows: one set for user-level events (denoted with `u`) and one set for kernel-level events (denoted with `k`). Each set, in turn, contains as many rows as there are page sizes supported (see `getpagesizes`(3C)). For each row, the percentage of CPU time is totalled and displayed in the right-most column. The two sets are delineated with a dashed line; CPUs are delineated with a solid line. If the parsable option (`-P`) is specified in addition to the `-T` option, the format of the data is as follows:

Field	Contents
1	Timestamp (nanoseconds since start)
2	CPU ID
3	Mode `k` denotes kernel, `u` denotes user)
4	Page size, in decimal

Field	Contents
5	I-TLB misses
6	Percentage of time in I-TLB miss handler
7	I-TSB misses
8	Percentage of time in I-TSB miss handler
9	D-TLB misses
10	Percentage of time in D-TLB miss handler
11	D-TSB misses
12	Percentage of time in D-TSB miss handler

Each field is separated with whitespace. If the format is modified, it will be modified by adding potentially new fields beginning with field 13; extant fields will remain unchanged.

EXAMPLES

EXAMPLE 1 Using `trapstat` Without Options

When run without options, `trapstat` displays a table of trap types and CPUs. At most six columns can fit in the default terminal width; if (as in this example) there are more than six CPUs, multiple tables are displayed:

```
example# trapstat
vct  name                |     cpu0     cpu1     cpu4     cpu5     cpu8     cpu9
-------------------------+-----------------------------------------------------
 24 cleanwin             |     6446     4837     6368     2153     2623     1321
 41 level-1              |      100        0        0        0        1        0
 44 level-4              |        0        1        1        1        0        0
 45 level-5              |        0        0        0        0        0        0
 47 level-7              |        0        0        0        0        9        0
 49 level-9              |      100      100      100      100      100      100
 4a level-10             |      100        0        0        0        0        0
 4d level-13             |        6       10        7       16       13       11
 4e level-14             |      100        0        0        0        1        0
 60 int-vec              |     2607     2740     2642     2922     2920     3033
 64 itlb-miss            |     3129     2475     3167     1037     1200      569
 68 dtlb-miss            |   121061    86162   109838    37386    45639    20269
 6c dtlb-prot            |      997      847     1061      379      406      184
 84 spill-user-32        |     2809     2133     2739   200806   332776   454504
 88 spill-user-64        |    45819   207856    93487   228529    68373    77590
 8c spill-user-32-cln    |      784      561      767      274      353      215
 90 spill-user-64-cln    |        9       37       17       39       12       13
 98 spill-kern-64        |    62913    50145    63869    21916    28431    11738
 a4 spill-asuser-32      |     1327      947     1288      460      572      335
 a8 spill-asuser-64      |       26       48       18       54       10       14
```

EXAMPLE 1 Using `trapstat` Without Options *(Continued)*

```
 ac spill-asuser-32-cln |     4580     3599     4555     1538     1978      857
 b0 spill-asuser-64-cln |       26        0        0        2        0        0
 c4 fill-user-32        |     2862     2161     2798   191746   318115   435850
 c8 fill-user-64        |    45813   197781    89179   217668    63905    74281
 cc fill-user-32-cln    |     3802     2833     3733    10153    16419    19475
 d0 fill-user-64-cln    |      329    10105     4873    10603     4235     3649
 d8 fill-kern-64        |    62519    49943    63611    21824    28328    11693
108 syscall-32          |     2285     1634     2278      737      957      383
126 self-xcall          |      100        0        0        0        0        0

vct  name               |    cpu12    cpu13    cpu14    cpu15
------------------------+-----------------------------------
 24 cleanwin            |     5435     4232     6302     6104
 41 level-1             |        0        0        0        0
 44 level-4             |        2        0        0        1
 45 level-5             |        0        0        0        0
 47 level-7             |        0        0        0        0
 49 level-9             |      100      100      100      100
 4a level-10            |        0        0        0        0
 4d level-13            |       15       11       22       11
 4e level-14            |        0        0        0        0
 60 int-vec             |     2813     2833     2738     2714
 64 itlb-miss           |     2636     1925     3133     3029
 68 dtlb-miss           |    90528    70639   107786   103425
 6c dtlb-prot           |      819      675      988      954
 84 spill-user-32       |   175768    39933     2811     2742
 88 spill-user-64       |        0   241348    96907   118298
 8c spill-user-32-cln   |      681      513      753      730
 90 spill-user-64-cln   |        0       42       16       20
 98 spill-kern-64       |    52158    40914    62305    60141
 a4 spill-asuser-32     |     1113      856     1251     1208
 a8 spill-asuser-64     |        0       64       16       24
 ac spill-asuser-32-cln |     3816     2942     4515     4381
 b0 spill-asuser-64-cln |        0        0        0        0
 c4 fill-user-32        |   170744    38444     2876     2784
 c8 fill-user-64        |        0   230381    92941   111694
 cc fill-user-32-cln    |     8550     3790     3612     3553
 d0 fill-user-64-cln    |        0    10726     4495     5845
 d8 fill-kern-64        |    51968    40760    62053    59922
108 syscall-32          |     1839     1495     2144     2083
126 self-xcall          |        0        0        0        0
```

EXAMPLE 2 Using `trapset` with CPU Filtering

The `-c` option can be used to limit the CPUs on which `trapstat` is enabled. This example limits `CPU 1` and CPUs `12` through `15`.

```
example# trapstat -c 1,12-15
vct  name               |     cpu1    cpu12    cpu13    cpu14    cpu15
------------------------+-----------------------------------------------
 24 cleanwin            |     6923     3072     2500     3518     2261
 44 level-4             |        3        0        0        1        1
 49 level-9             |      100      100      100      100      100
 4d level-13            |       23        8       14       19       14
```

EXAMPLE 2 Using `trapset` with CPU Filtering *(Continued)*

```
 60 int-vec             |     2559     2699     2752     2688     2792
 64 itlb-miss           |     3296     1548     1174     1698     1087
 68 dtlb-miss           |   114788    54313    43040    58336    38057
 6c dtlb-prot           |     1046      549      417      545      370
 84 spill-user-32       |    66551    29480   301588    26522   213032
 88 spill-user-64       |        0   318652   111239   299829   221716
 8c spill-user-32-cln   |      856      347      331      416      293
 90 spill-user-64-cln   |        0       55       21       59       39
 98 spill-kern-64       |    66464    31803    24758    34004    22277
 a4 spill-asuser-32     |     1423      569      560      698      483
 a8 spill-asuser-64     |        0       74       32       98       46
 ac spill-asuser-32-cln |     4875     2250     1728     2384     1584
 b0 spill-asuser-64-cln |        0        2        0        1        0
 c4 fill-user-32        |    64193    28418   287516    27055   202093
 c8 fill-user-64        |        0   305016   106692   288542   210654
 cc fill-user-32-cln    |     6733     3520    15185     2396    12035
 d0 fill-user-64-cln    |        0    13226     3506    12933    11032
 d8 fill-kern-64        |    66220    31680    24674    33892    22196
108 syscall-32          |     2446      967      817     1196      755
```

EXAMPLE 3 Using `trapstat` with TLB Statistics

The `-t` option displays in-depth TLB statistics, including the amount of time spent performing TLB miss processing. The following example shows that the machine is spending 14.1 percent of its time just handling D-TLB misses:

```
example# trapstat -t
cpu m| itlb-miss %tim itsb-miss %tim | dtlb-miss %tim dtsb-miss %tim |%tim
-----+-------------------------------+-------------------------------+----
  0 u|      2571  0.3         0  0.0 |     10802  1.3         0  0.0 | 1.6
  0 k|         0  0.0         0  0.0 |    106420 13.4       184  0.1 |13.6
-----+-------------------------------+-------------------------------+----
  1 u|      3069  0.3         0  0.0 |     10983  1.2       100  0.0 | 1.6
  1 k|        27  0.0         0  0.0 |    106974 12.6        19  0.0 |12.7
-----+-------------------------------+-------------------------------+----
  2 u|      3033  0.3         0  0.0 |     11045  1.2       105  0.0 | 1.6
  2 k|        43  0.0         0  0.0 |    107842 12.7       108  0.0 |12.8
-----+-------------------------------+-------------------------------+----
  3 u|      2924  0.3         0  0.0 |     10380  1.2       121  0.0 | 1.6
  3 k|        54  0.0         0  0.0 |    102682 12.2        16  0.0 |12.2
-----+-------------------------------+-------------------------------+----
  4 u|      3064  0.3         0  0.0 |     10832  1.2       120  0.0 | 1.6
  4 k|        31  0.0         0  0.0 |    107977 13.0       236  0.1 |13.1
=====+===============================+===============================+====
 ttl |     14816  0.3         0  0.0 |    585937 14.1      1009  0.0 |14.5
```

EXAMPLE 4 Using `trapstat` with TLB Statistics and Page Size Information

By specifying the `-T` option, `trapstat` shows TLB misses broken down by page size. In this example, CPU 0 is spending 7.9 percent of its time handling user-mode TLB misses on 8K pages, and another 2.3 percent of its time handling user-mode TLB misses on 64K pages.

EXAMPLE 4 Using `trapstat` with TLB Statistics and Page Size Information *(Continued)*

```
example# trapstat -T -c 0
cpu m size| itlb-miss %tim itsb-miss %tim | dtlb-miss %tim dtsb-miss %tim |%tim
----------+-------------------------------+-------------------------------+----
  0 u   8k|      1300  0.1        15  0.0 |    104897  7.9        90  0.0 | 8.0
  0 u  64k|         0  0.0         0  0.0 |     29935  2.3         7  0.0 | 2.3
  0 u 512k|         0  0.0         0  0.0 |      3569  0.2         2  0.0 | 0.2
  0 u   4m|         0  0.0         0  0.0 |       233  0.0         2  0.0 | 0.0
- - - - - + - - - - - - - - - - - - - - - + - - - - - - - - - - - - - - - + - -
  0 k   8k|        13  0.0         0  0.0 |     71733  6.5       110  0.0 | 6.5
  0 k  64k|         0  0.0         0  0.0 |         0  0.0         0  0.0 | 0.0
  0 k 512k|         0  0.0         0  0.0 |         0  0.0       206  0.1 | 0.1
  0 k   4m|         0  0.0         0  0.0 |         0  0.0         0  0.0 | 0.0
==========+===============================+===============================+====
      ttl |      1313  0.1        15  0.0 |    210367 17.1       417  0.2 |17.5
```

EXAMPLE 5 Using `trapstat` with Entry Filtering

By specifying the `-e` option, `trapstat` displays statistics for only specific trap types. Using this option minimizes the probe effect when seeking specific data. This example yields statistics for only the *dtlb-prot* and *syscall-32* traps on CPUs 12 through 15:

```
example# trapstat -e dtlb-prot,syscall-32 -c 12-15
vct  name                 |    cpu12    cpu13    cpu14    cpu15
------------------------+-----------------------------------
 6c dtlb-prot             |      817      754     1018      560
108 syscall-32            |     1426     1647     2186     1142

vct  name                 |    cpu12    cpu13    cpu14    cpu15
------------------------+-----------------------------------
 6c dtlb-prot             |     1085      996      800      707
108 syscall-32            |     2578     2167     1638     1452
```

EXAMPLE 6 Using `trapstat` with a Higher Sampling Rate

The following example uses the `-r` option to specify a sampling rate of 1000 samples per second, and filter only for the level-10 trap. Additionally, specifying the `-P` option yields parsable output.

Notice the timestamp difference between the level-10 events: 9,998,000 nanoseconds and 10,007,000 nanoseconds. These level-10 events correspond to the system clock, which by default ticks at 100 hertz (that is, every 10,000,000 nanoseconds).

```
example# trapstat -e level-10 -P -r 1000
1070400 0 4a level-10 0
2048600 0 4a level-10 0
3030400 0 4a level-10 1
4035800 0 4a level-10 0
5027200 0 4a level-10 0
6027200 0 4a level-10 0
7027400 0 4a level-10 0
8028200 0 4a level-10 0
9026400 0 4a level-10 0
```

EXAMPLE 6 Using `trapstat` with a Higher Sampling Rate *(Continued)*

```
10029600 0 4a level-10 0
11028600 0 4a level-10 0
12024000 0 4a level-10 0
13028400 0 4a level-10 1
14031200 0 4a level-10 0
15027200 0 4a level-10 0
16027600 0 4a level-10 0
17025000 0 4a level-10 0
18026000 0 4a level-10 0
19027800 0 4a level-10 0
20025600 0 4a level-10 0
21025200 0 4a level-10 0
22025000 0 4a level-10 0
23035400 0 4a level-10 1
24027400 0 4a level-10 0
25026000 0 4a level-10 0
26027000 0 4a level-10 0
```

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu
Interface Stability	
Human Readable Output	Unstable
Parsable Output	Evolving

SEE ALSO

`lockstat`(1M), `pmap`(1), `psrset`(1M), `psrinfo`(1M), `pbind`(1M), `ppgsz`(1), `getpagesizes`(3C)

Sun Microelectronics UltraSPARC I&II User's Manual, January 1997, STP1031,

The SPARC Architecture Manual, Version 9, 1994, Prentice-Hall.

NOTES

When enabled, `trapstat` induces a varying probe effect, depending on the type of information collected. While the precise probe effect depends upon the specifics of the hardware, the following table can be used as a rough guide:

Option	Approximate probe effect
default	3-5% per trap
`-e`	3-5% per specified trap
`-t`, `-T`	40-45% per TLB miss trap hitting in the TSB, 25-30% per TLB miss trap missing in the TSB

These probe effects are *per trap* not for the system as a whole. For example, running `trapstat` with the default options on a system that spends 7% of total time handling traps induces a performance degradation of less than one half of one percent; running `trapstat` with the `-t` or `-T` option on a system spending 5% of total time processing TLB misses induce a performance degradation of no more than 2.5%.

When run with the `-t` or `-T` option, `trapstat` accounts for its probe effect when calculating the *%tim* fields. This assures that the *%tim* fields are a reasonably accurate indicator of the time a given workload is spending handling TLB misses — regardless of the perturbing presence of `trapstat`.

While the *%tim* fields include the explicit cost of executing the TLB miss handler, they do *not* include the implicit costs of TLB miss traps (for example, pipeline effects, cache pollution, etc). These implicit costs become more significant as the trap rate grows; if high *%tim* values are reported (greater than 50%), you can accurately infer that much of the balance of time is being spent on the implicit costs of the TLB miss traps.

Due to the potential system wide degradation induced, only the super-user can run `trapstat`.

Due to the limitation of the underlying statistics gathering methodology, only one instance of `trapstat` can run at a time.

NAME

ttyadm – format and output port monitor-specific information

SYNOPSIS

/usr/sbin/ttyadm [-b] [-c] [-h] [-I] [-r *count*] [-i *msg*] [-m *modules*] [-p *prompt*] [-t *timeout*] [-S y | n] [-T *termtype*] -d *device* -l *ttylabel* -s *service*

/usr/sbin/ttyadm -V

DESCRIPTION

The `ttyadm` command is an administrative command that formats `ttymon`(1M)-specific information and writes it to standard output. The Service Access Facility (SAF) requires each port monitor to provide such a command. Note that the port monitor administrative file is updated by the Service Access Controller's administrative commands, `sacadm`(1M) and `pmadm`(1M). `ttyadm` provides a means of presenting formatted port monitor-specific (`ttymon`-specific) data to these commands.

OPTIONS

The following options are supported:

`-b` Set the "bi-directional port" flag. When this flag is set, the line can be used in both directions. `ttymon` will allow users to connect to the service associated with the port, but if the port is free, `uucico`(1M), `cu`(1C), or `ct`(1C) can use it for dialing out.

`-c` Set the connect-on-carrier flag for the port. If the `-c` flag is set, `ttymon` will invoke the port's associated service immediately when a connect indication is received (that is, no prompt is printed and no baud-rate searching is done).

`-d` *device* *device* is the full pathname of the device file for the TTY port.

`-h` Set the hangup flag for the port. If the `-h` flag is not set, `ttymon` will force a hangup on the line by setting the speed to `0` before setting the speed to the default or specified value.

`-i` *message* Specify the inactive (disabled) response message. This message will be sent to the TTY port if the port is disabled or the `ttymon` monitoring the port is disabled.

`-I` Initialize only. If the `-I` option is used, `ttymon` will invoke the service only once. This can be used to configure a particular device without actually monitoring it, as with software carrier.

`-l` *ttylabel* Specify which *ttylabel* in the `/etc/ttydefs` file to use as the starting point when searching for the proper baud rate.

`-m` *modules* Specify a list of pushable STREAMS modules. The modules will be pushed in the order in which they are specified before the service is invoked. *modules* must be a comma-separated list of modules, with no white space included. Any modules currently on the stream will be popped before these modules are pushed.

`-p` *prompt* Specify the prompt message, for example, "`login:`".

`-r` *count*	When the `-r` option is invoked, `ttymon` will wait until it receives data from the port before it displays a prompt. If *count* is `0`, `ttymon` will wait until it receives any character. If *count* is greater than `0`, `ttymon` will wait until *count* newlines have been received.
`-s` *service*	*service* is the full pathname of the service to be invoked when a connection request is received. If arguments are required, the command and its arguments must be enclosed in double quotes (" ").
`-S y|n`	Set the software carrier value. `y` will turn software carrier on. `n` will turn software carrier off.
`-t` *timeout*	Specify that `ttymon` should close a port if the open on the port succeeds, and no input data is received in *timeout* seconds.
`-T` *termtype*	Set the terminal type. The `TERM` environment variable will be set to *termtype*.
`-V`	Display the version number of the current `/usr/lib/saf/ttymon` command.

OUTPUT

If successful, `ttyadm` will generate the requested information, write it to standard output, and exit with a status of `0`. If `ttyadm` is invoked with an invalid number of arguments or invalid arguments, or if an incomplete option is specified, an error message will be written to standard error and `ttymon` will exit with a non-zero status.

FILES

`/etc/ttydefs`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`ct`(1C), `cu`(1C), `pmadm`(1M), `sacadm`(1M), `ttymon`(1M), `uucico`(1M), `attributes`(5)

System Administration Guide: Basic Administration

NAME

ttymon – port monitor for terminal ports

SYNOPSIS

/usr/lib/saf/ttymon

/usr/lib/saf/ttymon `-g [-d` *device*`] [-h] [-t` *timeout*`] [-l` *ttylabel*`] [-p` *prompt*`] [-m` *modules*`] [-T` *termtype*`]`

DESCRIPTION

`ttymon` is a STREAMS-based TTY port monitor. Its function is to monitor ports, to set terminal modes, baud rates, and line disciplines for the ports, and to connect users or applications to services associated with the ports. Normally, `ttymon` is configured to run under the Service Access Controller, `sac`(1M), as part of the Service Access Facility (SAF). It is configured using the `sacadm`(1M) command. Each instance of `ttymon` can monitor multiple ports. The ports monitored by an instance of `ttymon` are specified in the port monitor's administrative file. The administrative file is configured using the `pmadm`(1M) and `ttyadm`(1M) commands. When an instance of `ttymon` is invoked by the `sac` command, it starts to monitor its ports. For each port, `ttymon` first initializes the line disciplines, if they are specified, and the speed and terminal settings. For ports with entries in `/etc/logindevperm`, device owner, group and permissions are set. (See `logindevperm`(4).) The values used for initialization are taken from the appropriate entry in the TTY settings file. This file is maintained by the `sttydefs`(1M) command. Default line disciplines on ports are usually set up by the `autopush`(1M) command of the Autopush Facility.

`ttymon` then writes the prompt and waits for user input. If the user indicates that the speed is inappropriate by pressing the BREAK key, `ttymon` tries the next speed and writes the prompt again. When valid input is received, `ttymon` interprets the per-service configuration file for the port, if one exists, creates a `utmpx` entry if required (see `utmpx`(4)), establishes the service environment, and then invokes the service associated with the port. Valid input consists of a string of at least one non-newline character, terminated by a carriage return. After the service terminates, `ttymon` cleans up the `utmpx` entry, if one exists, and returns the port to its initial state.

If *autobaud* is enabled for a port, `ttymon` will try to determine the baud rate on the port automatically. Users must enter a carriage return before `ttymon` can recognize the baud rate and print the prompt. Currently, the baud rates that can be determined by *autobaud* are `110`, `1200`, `2400`, `4800`, and `9600`.

If a port is configured as a bidirectional port, `ttymon` will allow users to connect to a service, and, if the port is free, will allow `uucico`(1M), `cu`(1C), or `ct`(1C) to use it for dialing out. If a port is bidirectional, `ttymon` will wait to read a character before it prints a prompt.

If the *connect-on-carrier* flag is set for a port, `ttymon` will immediately invoke the port's associated service when a connection request is received. The prompt message will not be sent.

If a port is disabled, `ttymon` will not start any service on that port. If a disabled message is specified, `ttymon` will send out the disabled message when a connection request is received. If `ttymon` is disabled, all ports under that instance of `ttymon` will also be disabled.

SERVICE INVOCATION

The service `ttymon` invokes for a port is specified in the `ttymon` administrative file. `ttymon` will scan the character string giving the service to be invoked for this port, looking for a `%d` or a `%%` two-character sequence. If `%d` is found, `ttymon` will modify the service command to be executed by replacing those two characters by the full path name of this port (the device name). If `%%` is found, they will be replaced by a single `%`. When the service is invoked, file descriptor `0`, `1`, and `2` are opened to the port device for reading and writing. The service is invoked with the user ID, group ID and current home directory set to that of the user name under which the service was registered with `ttymon`. Two environment variables, `HOME` and `TTYPROMPT`, are added to the service's environment by `ttymon`. `HOME` is set to the home directory of the user name under which the service is invoked. `TTYPROMPT` is set to the prompt string configured for the service on the port. This is provided so that a service invoked by `ttymon` has a means of determining if a prompt was actually issued by `ttymon` and, if so, what that prompt actually was.

See `ttyadm`(1M) for options that can be set for ports monitored by `ttymon` under the Service Access Controller.

SECURITY

`ttymon` uses `pam`(3PAM) for session management. The PAM configuration policy, listed through `/etc/pam.conf`, specifies the modules to be used for `ttymon`. Here is a partial `pam.conf` file with entries for `ttymon` using the UNIX session management module.

```
ttymon   session   required   /usr/lib/security/pam_unix.so.1
```

If there are no entries for the `ttymon` service, then the entries for the "other" service will be used.

OPTIONS

`-g` A special invocation of `ttymon` is provided with the `-g` option. This form of the command should only be called by applications that need to set the correct baud rate and terminal settings on a port and then connect to `login` service, but that cannot be pre-configured under the SAC. The following combinations of options can be used with `-g`:

`-d` *device* *device* is the full path name of the port to which `ttymon` is to attach. If this option is not specified, file descriptor `0` must be set up by the invoking process to a TTY port.

`-h` If the -h flag is not set, `ttymon` will force a hangup on the line by setting the speed to zero before setting the speed to the default or specified speed.

`-l` *ttylabel* *ttylabel* is a link to a speed and TTY definition in the `ttydefs` file. This definition tells `ttymon` at what speed to run initially, what the initial TTY settings are, and what speed to try next if the user indicates that the speed is inappropriate by pressing the BREAK key. The default speed is 9600 baud.

`-m` *modules* When initializing the port, `ttymon` will pop all modules on the port, and then push *modules* in the order specified. *modules* is a

comma-separated list of pushable modules. Default modules on the ports are usually set up by the Autopush Facility.

`-p` *prompt* — Allows the user to specify a prompt string. The default prompt is `Login: .`

`-t` *timeout* — Specifies that `ttymon` should exit if no one types anything in *timeout* seconds after the prompt is sent.

`-T` *termtype* — Sets the `TERM` environment variable to *termtype*.

ENVIRONMENT VARIABLES

If any of the `LC_*` variables (`LC_CTYPE`, `LC_MESSAGES`, `LC_TIME`, `LC_COLLATE`, `LC_NUMERIC`, and `LC_MONETARY`) (see `environ`(5)) are not set in the environment, the operational behavior of `ttymon` for each corresponding locale category is determined by the value of the `LANG` environment variable. If `LC_ALL` is set, its contents are used to override both the `LANG` and the other `LC_*` variables. If none of the above variables is set in the environment, the "C" (U.S. style) locale determines how `ttymon` behaves.

`LC_CTYPE` — Determines how `ttymon` handles characters. When `LC_CTYPE` is set to a valid value, `ttymon` can display and handle text and filenames containing valid characters for that locale. `ttymon` can display and handle Extended Unix Code (EUC) characters where any individual character can be 1, 2, or 3 bytes wide. `ttymon` can also handle EUC characters of 1, 2, or more column widths. In the "C" locale, only characters from ISO 8859-1 are valid.

FILES

`/etc/logindevperm`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`ct`(1C), `cu`(1C), `autopush`(1M), `pmadm`(1M), `sac`(1M), `sacadm`(1M), `sttydefs`(1M), `ttyadm`(1M), `uucico`(1M), `pam`(3PAM), `logindevperm`(4), `pam.conf`(4), `utmpx`(4), `attributes`(5), `environ`(5), `pam_authtok_check`(5), `pam_authtok_get`(5), `pam_authtok_store`(5), `pam_dhkeys`(5), `pam_passwd_auth`(5), `pam_unix`(5), `pam_unix_account`(5), `pam_unix_auth`(5), `pam_unix_session`(5)

System Administration Guide: Basic Administration

NOTES

If a port is monitored by more than one `ttymon`, it is possible for the `ttymon`s to send out prompt messages in such a way that they compete for input.

The `pam_unix`(5) module might not be supported in a future release. Similar functionality is provided by `pam_authtok_check`(5), `pam_authtok_get`(5), `pam_authtok_store`(5), `pam_dhkeys`(5), `pam_passwd_auth`(5), `pam_unix_account`(5), `pam_unix_auth`(5), and `pam_unix_session`(5).

NAME

tunefs – tune up an existing file system

SYNOPSIS

tunefs [-a *maxcontig*] [-d *rotdelay*] [-e *maxbpg*] [-m *minfree*] [-o [space | time]] *special* | *filesystem*

DESCRIPTION

`tunefs` is designed to change the dynamic parameters of a file system which affect the layout policies. When using `tunefs` with *filesystem*, *filesystem* must be in `/etc/vfstab`. The parameters which are to be changed are indicated by the options given below.

Generally, one should optimize for time unless the file system is over 90% full.

OPTIONS

The following options are supported:

`-a` *maxcontig*
Specifies the maximum number of contiguous logical blocks that will be laid out before forcing a rotational delay (see `-d`). The default value is determined from the disk drive's maximum transfer rate.

UFS supports no more than `1048576` byte (`1MB`) runs. With a logical block size of `4096`, this gives a *maxcontig* limit of `256`. With `8192`, the limit is `128`.

`-d` *rotdelay*
Specifies the expected time (in milliseconds) to service a transfer completion interrupt and initiate a new transfer on the same disk. It is used to decide how much rotational spacing to place between successive blocks in a file.

`-e` *maxbpg*
Indicates the maximum number of contiguous logical blocks any single file can allocate out of a cylinder group before it is forced to begin allocating blocks from another cylinder group. Typically this value is set to approximately one quarter of the total contiguous logical blocks in a cylinder group. The intent is to prevent any single file from using up all the blocks in a single cylinder group, thus degrading access times for all files subsequently allocated in that cylinder group. The effect of this limit is to cause big files to do long seeks more frequently than if they were allowed to allocate all the blocks in a cylinder group before seeking elsewhere. For file systems with exclusively large files, this parameter should be set higher.

`-m` *minfree*
Specifies the percentage of space held back from normal users; the minimum free space threshold. This value can be set to `0`; however, up to a factor of three in throughput will be lost over the performance obtained at a 10% threshold. *Note:* If the value is raised above the current usage level, users will be unable to allocate files until enough files have been deleted to get under the higher threshold.

`-o` [`space` | `time`]
Changes optimization strategy for the file system.

`space`: conserves space

`time`: attempt to organize file layout to minimize access time

USAGE

See `largefile`(5) for the description of the behavior of `tunefs` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`mkfs`(1M), `attributes`(5), `largefile`(5)

NAME

tzselect – select a time zone

SYNOPSIS

/usr/bin/tzselect

DESCRIPTION

The `tzselect` program asks you a series of questions about the current location and outputs the resulting time zone description to standard output. The output is suitable as a value for the `TZ` environment variable.

All user interaction is through standard input and standard error.

OPTIONS

The `tzselect` command has no options.

EXIT STATUS

The following exit values are returned:

`0` Timezone information was successfully obtained.

`>0` An error occurred.

FILES

`/usr/share/lib/zoneinfo`
directory containing timezone data files

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu
Stability	Evolving

SEE ALSO

`zdump`(1M), `zic`(1M), `ctime`(3C), `attributes`(5)

NAME

uadmin – administrative control

SYNOPSIS

/usr/sbin/uadmin *cmd fcn* [*mdep*]

/sbin/uadmin *cmd fcn* [*mdep*]

DESCRIPTION

The `uadmin` command provides control for basic administrative functions. This command is tightly coupled to the system administration procedures and is not intended for general use. It may be invoked only by the super-user.

Both the *cmd* (command) and *fcn* (function) arguments are converted to integers and passed to the `uadmin` system call. The optional *mdep* (machine dependent) argument is only available for the *cmd* values of `1` (`A_REBOOT`) or `2` (`A_SHUTDOWN`), to pass a single string of boot arguments to the `uadmin` system call. For any other *cmd* value, no *mdep* command-line argument is allowed.

When passing an *mdep* value that contains whitespaces, the string must be grouped together as a single argument enclosed within quotes (for example, **`uadmin 1 1 "-s kernel/unix"`**).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`uadmin`(2), `attributes`(5)

NAME

ufsdump – incremental file system dump

SYNOPSIS

/usr/sbin/ufsdump [*options*] [*arguments*] *files_to_dump*

DESCRIPTION

`ufsdump` backs up all files specified by *files_to_dump* (normally either a whole file system or files within a file system changed after a certain date) to magnetic tape, diskette, or disk file. When running `ufsdump`, the file system must be inactive; otherwise, the output of `ufsdump` may be inconsistent and restoring files correctly may be impossible. A file system is inactive when it is unmounted or the system is in single user mode. A file system is not considered inactive if one tree of the file system is quiescent while another tree has files or directories being modified.

options is a single string of one-letter `ufsdump` options.

arguments may be multiple strings whose association with the options is determined by order. That is, the first argument goes with the first option that takes an argument; the second argument goes with the second option that takes an argument, and so on.

files_to_dump is required and must be the last argument on the command line. See `OPERANDS` for more information.

With most devices `ufsdump` can automatically detect the end-of-media. Consequently, the `d`, `s`, and `t` options are not necessary for multi-volume dumps, unless `ufsdump` does not understand the way the device detects the end-of-media, or the files are to be restored on a system with an older version of the `restore` command.

OPTIONS

The following options are supported:

`0–9`
: The "dump level." All files specified by *files_to_dump* that have been modified since the last `ufsdump` at a lower dump level are copied to the *dump_file* destination (normally a magnetic tape device). For instance, if a "level `2`" dump was done on Monday, followed by a "level `4`" dump on Tuesday, a subsequent "level `3`" dump on Wednesday would contain all files modified or added since the "level `2`" (Monday) backup. A "level `0`" dump copies the entire file system to the *dump_file*.

`a` *archive_file*
: Archive file. Archive a dump table-of-contents in the specified *archive_file* to be used by `ufsrestore`(1M) to determine whether a file is in the dump file that is being restored.

`b` *factor*
: Blocking factor. Specify the blocking factor for tape writes. The default is 20 blocks per write for tapes of density less than 6250BPI (bytes-per-inch). The default blocking factor for tapes of density 6250BPI and greater is 64. The default blocking factor for cartridge tapes (`c` option) is 126. The highest blocking factor available

with most tape drives is 126. Note: the blocking factor is specified in terms of 512-byte blocks, for compatibility with `tar`(1).

`c` — Cartridge. Set the defaults for cartridge instead of the standard half-inch reel. This sets the density to 1000BPI and the blocking factor to 126. Since `ufsdump` can automatically detect the end-of-media, only the blocking parameter normally has an effect. When cartridge tapes are used, and this option is *not* specified, `ufsdump` will slightly miscompute the size of the tape. If the `b`, `d`, `s` or `t` options are specified with this option, their values will override the defaults set by this option.

`d` *bpi* — Tape density. Not normally required, as `ufsdump` can detect end-of-media. This parameter can be used to keep a running tab on the amount of tape used per reel. The default density is 6250BPI except when the `c` option is used for cartridge tape, in which case it is assumed to be 1000BPI per track. Typical values for tape devices are:

1/2 inch tape
6250 BPI

1/4 inch cartridge
1000 BPI The tape densities and other options are documented in the `st`(7D) man page.

`D` — Diskette. Dump to diskette.

`f` *dump_file* — Dump file. Use *dump_file* as the file to dump to, instead of `/dev/rmt/0`. If *dump_file* is specified as –, dump to standard output.

If the name of the file is of the form *machine*`:`*device*, the dump is done from the specified machine over the network using `rmt`(1M). Since `ufsdump` is normally run by root, the name of the local machine must appear in the `/.rhosts` file of the remote machine. If the file is specified as *user*`@`*machine*`:`*device*, `ufsdump` will attempt to execute as the specified user on the remote machine. The specified user must have a `.rhosts` file on the remote machine that allows the user invoking the command from the local machine to access the remote machine.

`l` — Autoload. When the end-of-tape is reached before the dump is complete, take the drive offline and wait up to two minutes for the tape drive to be ready again. This

gives autoloading (stackloader) tape drives a chance to load a new tape. If the drive is ready within two minutes, continue. If it is not, prompt for another tape and wait.

`L` *string*
: Sets the tape label to *string*, instead of the default `none`. *string* may be no more than sixteen characters long. If it is longer, it is truncated and a warning printed; the dump will still be done. The tape label is specific to the `ufsdump` tape format, and bears no resemblance to IBM or ANSI-standard tape labels.

`n`
: Notify all operators in the `sys` group that `ufsdump` requires attention by sending messages to their terminals, in a manner similar to that used by the `wall`(1M) command. Otherwise, such messages are sent only to the terminals (such as the console) on which the user running `ufsdump` is logged in.

`N` *device_name*
: Use *device_name* when recording information in `/etc/dumpdates` (see the `u` option) and when comparing against information in `/etc/dumpdates` for incremental dumps. The *device_name* provided can contain no white space as defined in `scanf`(3C) and is case-sensitive.

`o`
: Offline. Take the drive offline when the dump is complete or the end-of-media is reached and rewind the tape, or eject the diskette. In the case of some autoloading 8mm drives, the tape is removed from the drive automatically. This prevents another process which rushes in to use the drive, from inadvertently overwriting the media.

`s` *size*
: Specify the *size* of the volume being dumped to. Not normally required, as `ufsdump` can detect end-of-media. When the specified size is reached, `ufsdump` waits for you to change the volume. `ufsdump` interprets the specified size as the length in feet for tapes and cartridges, and as the number of 1024-byte blocks for diskettes. The values should be a little smaller than the actual physical size of the media (for example, 425 for a 450-foot cartridge). Typical values for tape devices depend on the `c` option, for cartridge devices, and the `D` option for diskettes:

  1/2 inch tape
  : 2300 feet

60-Mbyte 1/4 inch cartridge
425 feet

150-Mbyte 1/4 inch cartridge
700 feet

diskette
1422 blocks (Corresponds to a 1.44-Mbyte diskette, with one cylinder reserved for bad block information.)

`S` — Size estimate. Determine the amount of space that is needed to perform the dump without actually doing it, and display the estimated number of bytes it will take. This is useful with incremental dumps to determine how many volumes of media will be needed.

`t` *tracks* — Specify the number of tracks for a cartridge tape. Not normally required, as `ufsdump` can detect end-of-media. The default is 9 tracks. The `t` option is not compatible with the `D` option. Values for Sun-supported tape devices are:

60-Mbyte 1/4 inch cartridge	9 tracks
150-Mbyte 1/4 inch cartridge	18 tracks

`T` *time_wait* `[hms]` — Sets the amount of time to wait for an `autoload` command to complete. This option is ignored unless the `l` option has also been specified. The default time period to wait is two minutes. Specify time units with a trailing `h` (for hours), `m` (for minutes), or `s` (for seconds). The default unit is minutes.

`u` — Update the dump record. Add an entry to the file `/etc/dumpdates`, for each file system successfully dumped that includes the file system name (or *device_name* as specified with the `N` option), date, and dump level.

`v` — Verify. After each tape or diskette is written, verify the contents of the media against the source file system. If any discrepancies occur, prompt for new media, then repeat the dump/verification process. The file system *must* be unmounted. This option cannot be used to verify a dump to standard output.

`w` — Warning. List the file systems that have not been backed up within a day. This information is gleaned from the files `/etc/dumpdates` and `/etc/vfstab`. When the `w` option is used, all other options are ignored. After reporting, `ufsdump` exits immediately.

`W` Warning with highlight. Similar to the `w` option, except that the `W` option includes all file systems that appear in `/etc/dumpdates`, along with information about their most recent dump dates and levels. File systems that have not been backed up within a day are highlighted.

OPERANDS

The following operand is supported:

files_to_dump Specifies the files to dump. Usually it identifies a whole file system by its raw device name (for example, `/dev/rdsk/c0t3d0s6`). Incremental dumps (levels `1` to `9`) of files changed after a certain date only apply to a whole file system. Alternatively, *files_to_dump* can identify individual files or directories. All named directories that may be examined by the user running `ufsdump`, as well as any explicitly-named files, are dumped. This dump is equivalent to a level `0` dump of the indicated portions of the filesystem, except that `/etc/dumpdates` is not updated even if the `-u` option has been specified. In all cases, the files must be contained in the same file system, and the file system must be local to the system where `ufsdump` is being run.

files_to_dump is required and must be the last argument on the command line.

If no *options* are given, the default is `9uf /dev/rmt/0` *files_to_dump*.

USAGE

See `largefile`(5) for the description of the behavior of `ufsdump` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXAMPLES

EXAMPLE 1 A sample display of the `ufsdump` command.

To make a full dump of a root file system on `c0t3d0`, on a 150-MByte cartridge tape unit `0`, use:

```
example# ufsdump 0cfu /dev/rmt/0 /dev/rdsk/c0t3d0s0
```

To make and verify an incremental dump at level `5` of the `usr` partition of `c0t3d0`, on a 1/2 inch reel tape unit `1`, use:

```
example# ufsdump 5fuv /dev/rmt/1 /dev/rdsk/c0t3d0s6
```

EXIT STATUS

While running, `ufsdump` emits many verbose messages. `ufsdump` returns the following exit values:

`0` Normal exit.

`1` Startup errors encountered.

`3` Abort – no checkpoint attempted.

FILES

`/dev/rmt/0` default unit to dump to

`/etc/dumpdates`	dump date record
`/etc/group`	to find group `sys`
`/etc/hosts`	to gain access to remote system with drive
`/etc/vfstab`	list of file systems

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`cpio`(1), `tar`(1), `dd`(1M), `devnm`(1M), `prtvtoc`(1M), `rmt`(1M), `shutdown`(1M), `ufsrestore`(1M), `volcopy`(1M), `wall`(1M), `scanf`(3C), `attributes`(5), `largefile`(5), `st`(7D)

Read Errors

Fewer than 32 read errors on the file system are ignored.

Process Per Reel

Because each reel requires a new process, parent processes for reels that are already written hang around until the entire tape is written.

Operator Intervention

`ufsdump` requires operator intervention on these conditions: end of volume, end of dump, volume write error, volume open error or disk read error (if there are more than a threshold of 32). In addition to alerting all operators implied by the `n` option, `ufsdump` interacts with the operator on `ufsdump`'s control terminal at times when `ufsdump` can no longer proceed, or if something is grossly wrong. All questions `ufsdump` poses *must* be answered by typing `yes` or `no`, as appropriate.

Since backing up a disk can involve a lot of time and effort, `ufsdump` checkpoints at the start of each volume. If writing that volume fails for some reason, `ufsdump` will, with operator permission, restart itself from the checkpoint after a defective volume has been replaced.

Suggested Dump Schedule

It is vital to perform full, "level `0`", dumps at regular intervals. When performing a full dump, bring the machine down to single-user mode using `shutdown`(1M). While preparing for a full dump, it is a good idea to clean the tape drive and heads. Incremental dumps should be performed with the system running in single-user mode.

Incremental dumps allow for convenient backup and recovery of active files on a more frequent basis, with a minimum of media and time. However, there are some tradeoffs. First, the interval between backups should be kept to a minimum (once a day at least). To guard against data loss as a result of a media failure (a rare, but possible occurrence), capture active files on (at least) two sets of dump volumes. Another consideration is the desire to keep unnecessary duplication of files to a minimum to

save both operator time and media storage. A third consideration is the ease with which a particular backed-up version of a file can be located and restored. The following four-week schedule offers a reasonable tradeoff between these goals.

```
            Sun     Mon     Tue     Wed     Thu     Fri
Week 1:     Full     5       5       5       5       3
Week 2:              5       5       5       5       3
Week 3:              5       5       5       5       3
Week 4:              5       5       5       5       3
```

Although the Tuesday through Friday incrementals contain "extra copies" of files from Monday, this scheme assures that any file modified during the week can be recovered from the previous day's incremental dump.

Process Priority of ufsdump

`ufsdump` uses multiple processes to allow it to read from the disk and write to the media concurrently. Due to the way it synchronizes between these processes, any attempt to run dump with a `nice` (process priority) of '–5' or better will likely make `ufsdump` run *slower* instead of faster.

Overlapping Partitions

Most disks contain one or more overlapping slices because slice 2 covers the entire disk. The other slices are of various sizes and usually do not overlap. For example, a common configuration places `root` on slice 0, `swap` on slice 1, `/opt` on slice 5 and `/usr` on slice 6.

It should be emphasized that `ufsdump` dumps one `ufs` file system at a time. Given the above scenario where slice 0 and slice 2 have the same starting offset, executing `ufsdump` on slice 2 with the intent of dumping the entire disk would instead dump only the `root` file system on slice 0. To dump the entire disk, the user must dump the file systems on each slice separately.

BUGS

The `/etc/vfstab` file does not allow the desired frequency of backup for file systems to be specified (as `/etc/fstab` did). Consequently, the `w` and `W` options assume file systems should be backed up daily, which limits the usefulness of these options.

NAME

ufsrestore – incremental file system restore

SYNOPSIS

```
/usr/sbin/ufsrestore i | r | R | t | x [abcdfhlmostvyLT]
      [archive_file] [factor] [dumpfile] [n] [label] [timeout] [ filename...]
```

DESCRIPTION

The `ufsrestore` utility restores files from backup media created with the `ufsdump` command. `ufsrestores`'s actions are controlled by the *key* argument. The *key* is exactly one `function` *letter* (`i`, `r`, `R` , `t`, or `x`) and zero or more *function modifier*s (letters). The *key* string contains no SPACE characters. Function modifier arguments are listed on the command line in the same order as their corresponding function modifiers appear in the *key* string.

filename arguments which appear on the command line, or as arguments to an interactive command, are treated as shell `glob` patterns by the `x` and `t` functions; any files or directories matching the patterns are selected. The metacharacters `*`, `?`, and `[ ]` must be protected from the shell if they appear on the command line. There is no way to quote these metacharacters to explicitly match them in a filename.

The temporary files `rstdir*` and `rstmode*` are placed in `/tmp` by default. If the environment variable `TMPDIR` is defined with a non-empty value, that location is used instead of `/tmp`.

Function Letters

You must specify one (and only one) of the function letters listed below. Note that `i`, `x`, and `r` are intended to restore files into an empty directory. The `R` function is intended for restoring into a populated directory.

`i` Interactive. After reading in the directory information from the media, `ufsrestore` invokes a shell-like interface that allows you to browse through the dump file's directory hierarchy and select individual files to be extracted. Restoration has the same semantics as `x` (see below). See `Interactive Commands`, below, for a description of available commands.

r Recursive. Starting with an empty directory and a level 0 dump, the `r` function recreates the filesystem relative to the current working directory, exactly as it appeared when the dump was made. Information used to restore incremental dumps on top of the full dump (for example, `restoresymtable`) is also included. Several `ufsrestore` runs are typical, one for each higher level of dump (0, 1, ..., 9). Files that were deleted between the level 0 and a subsequent incremental dump will not exist after the final restore. To completely restore a file system, use the `r` function restore the level 0 dump, and again for each incremental dump. Although this function letter is intended for a complete restore onto a new file system (one just created with `newfs`(1M)), if the file system contains files not on the backup media, they are preserved.

R Resume restoring. If an `r`-mode `ufsrestore` was interrupted, this function prompts for the volume from which to resume restoring and continues the restoration from where it was left off. Otherwise identical to `r`.

t — Table of contents. List each *filename* that appears on the media. If no *filename* argument is given, the root directory is listed. This results in a list of all files on the media, unless the `h` function modifier is in effect. The table of contents is taken from the media or from the specified archive file, when the `a` function modifier is used. The `a` function modifier is mutually exclusive with the `x` and `r` function letters.

x — Extract the named files from the media. If the *filename* argument matches a directory whose contents were written onto the media, and the `h` modifier is not in effect, the directory is recursively extracted, relative to the current directory, which is expected to be empty. For each file, the owner, modification time, and mode are restored (if possible). If you omit the *filename* argument or specify "`.`", the root directory is extracted. This results in the entire tape being extracted, unless the `h` modifier is in effect. With the `x` function, existing files are overwritten; `ufsrestore` displays the names of the overwritten files. Note that overwriting a currently-running executable can have unfortunate consequences.

Use the `x` option to restore partial file system dumps, as they are (by definition) not entire file systems.

Function Modifiers

a *archive_file* — Read the table of contents from *archive_file* instead of the media. This function modifier can be used in combination with the `t`, `i`, or `x` function letters, making it possible to check whether files are on the media without having to mount the media. When used with the `x` and interactive (`i`) function letters, it prompts for the volume containing the file(s) before extracting them.

b *factor* — Blocking factor. Specify the blocking factor for tape reads. For variable length SCSI tape devices, unless the data was written with the default blocking factor, a blocking factor at least as great as that used to write the tape must be used; otherwise, an error will be generated. Note that a tape block is 512 bytes. Refer to the man page for your specific tape driver for the maximum blocking factor.

c — Convert the contents of the media in 4.1BSD format to the new `ufs` file system format.

d — Debug. Turn on debugging output.

f *dump_file* — Use *dump_file* instead of `/dev/rmt/0` as the file to restore from. Typically *dump_file* specifies a tape or diskette drive. If *dump_file* is specified as '–', `ufsrestore` reads from the standard input. This allows `ufsdump`(1M) and `ufsrestore` to be used in a pipeline to copy a file system:

```
example# ufsdump 0f - /dev/rdsk/c0t0d0s7 \
 | (cd /home;ufsrestore xf -)
```

If the name of the file is of the form *machine*`:`*device*, the restore is done from the specified machine over the network using `rmt`(1M). Since `ufsrestore` is normally run

by root, the name of the local machine must appear in the `/.rhosts` file of the remote machine. If the file is specified as *user@machine*`:`*device*, `ufsrestore` will attempt to execute as the specified user on the remote machine. The specified user must have a `.rhosts` file on the remote machine that allows the user invoking the command from the local machine to access the remote machine.

h — Extract or list the actual directory, rather than the files that it references. This prevents hierarchical restoration of complete subtrees from the tape.

l — Autoload. When the end-of-tape is reached before the restore is complete, take the drive off-line and wait up to two minutes (the default, see the `T` function modifier) for the tape drive to be ready again. This gives autoloading (stackloader) tape drives a chance to load a new tape. If the drive is ready within two minutes, continue. If it is not, prompt for another tape and wait.

L *label* — The label that should appear in the header of the dump file. If the labels do not match, `ufsrestore` issues a diagnostic and exits. The tape label is specific to the `ufsdump` tape format, and bears no resemblance to IBM or ANSI-standard tape labels.

m — Extract by inode numbers rather than by filename to avoid regenerating complete pathnames. Regardless of where the files are located in the dump hierarchy, they are restored into the current directory and renamed with their inode number. This is useful if only a few files are being extracted.

o — Offline. Take the drive off-line when the restore is complete or the end-of-media is reached and rewind the tape, or eject the diskette. In the case of some autoloading 8mm drives, the tape is removed from the drive automatically.

s *n* — Skip to the *n*'th file when there are multiple dump files on the same tape. For example, the command:

```
example# ufsrestore xfs /dev/rmt/0hn 5
```

would position you to the fifth file on the tape when reading volume 1 of the dump. If a dump extends over more than one volume, all volumes except the first are assumed to start at position 0, no matter what "`s` *n*" value is specified.

If "`s` *n*" is specified, the backup media must be at BOT (beginning of tape). Otherwise, the initial positioning to read the table of contents will fail, as it is performed by skipping the tape forward *n*`-1` files rather than by using absolute positioning. This is because on some devices absolute positioning is very time consuming.

T timeout [hms] — Sets the amount of time to wait for an autoload command to complete. This function modifier is ignored unless the `l` function modifier has also been specified. The default timeout period is two minutes. The time units may be specified as a trailing `h` (hours), `m` (minutes), or `s` (seconds). The default unit is minutes.

v — Verbose. `ufsrestore` displays the name and inode number of each file it restores, preceded by its file type.

y — Do not ask whether to abort the restore in the event of tape errors. `ufsrestore` tries to skip over the bad tape block(s) and continue as best it can.

Interactive Commands

`ufsrestore` enters interactive mode when invoked with the `i` function letters. Interactive commands are reminiscent of the shell. For those commands that accept an argument, the default is the current directory. The interactive options are:

add [*filename*] — Add the named file or directory to the list of files to extract. If a directory is specified, add that directory and its files (recursively) to the extraction list (unless the `h` modifier is in effect).

cd *directory* — Change to *directory* (within the dump file).

delete [*filename*] — Delete the current directory, or the named file or directory from the list of files to extract. If a directory is specified, delete that directory and all its descendents from the extraction list (unless the `h` modifier is in effect). The most expedient way to extract a majority of files from a directory is to add that directory to the extraction list, and then delete specific files to omit.

extract — Extract all files on the extraction list from the dump media. `ufsrestore` asks which volume the user wishes to mount. The fastest way to extract a small number of files is to start with the last volume and work toward the first. If "`s` *n*" is given on the command line, volume 1 will automatically be positioned to file *n* when it is read.

help — Display a summary of the available commands.

ls [*directory*] — List files in *directory* or the current directory, represented by a '`.`' (period). Directories are appended with a '`/`' (slash). Entries marked for extraction are

	prefixed with a '*' (asterisk). If the verbose option is in effect, inode numbers are also listed.
marked [*directory*]	Like `ls`, except only files marked for extraction are listed.
pager	Toggle the pagination of the output from the `ls` and `marked` commands. The pager used is that defined by the `PAGER` environment variable, or more(1) if that envar is not defined. The `PAGER` envar may include white-space-separated arguments for the pagination program.
pwd	Print the full pathname of the current working directory.
quit	`ufsrestore` exits immediately, even if the extraction list is not empty.
setmodes	Prompts: `set owner/mode for '.'` (period). Type `y` for yes to set the mode (permissions, owner, times) of the current directory '.' (period) into which files are being restored equal to the mode of the root directory of the file system from which they were dumped. Normally, this is what you want when restoring a whole file system, or restoring individual files into the same locations from which they were dumped. Type `n` for no, to leave the mode of the current directory unchanged. Normally, this is what you want when restoring part of a dump to a directory other than the one from which the files were dumped.
setpager *command*	Sets the command to use for paginating output instead of the default or that inherited from the environment. The *command* string may include arguments in addition to the command itself.
verbose	Toggle the status of the `v` modifier. While `v` is in effect, the `ls` command lists the inode numbers of all entries, and `ufsrestore` displays information about each file as it is extracted.
what	Display the dump header on the media.

OPERANDS

The following operands are supported.

filename Specifies the pathname of files (or directories) to be restored to disk. Unless the `h` function modifier is also used, a directory name refers to the files it contains, and (recursively) its subdirectories and the files they contain. *filename* is associated with either the `x` or `t` function letters, and must come last.

USAGE

See `largefile`(5) for the description of the behavior of `ufsrestore` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`1` An error occurred. Verbose messages are displayed.

ENVIRONMENT VARIABLES

PAGER The command to use as a filter for paginating output. This can also be used to specify the options to be used. Default is `more`(1).

TMPDIR Selects the directory for temporary files. Defaults to `/tmp` if not defined in the environment.

FILES

`/dev/rmt/0` the default tape drive

`$TMPDIR/rstdir*` file containing directories on the tape

`$TMPDIR/rstmode*` owner, mode, and timestamps for directories

`./restoresymtable` information passed between incremental restores

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`more`(1), `mkfs`(1M), `mount`(1M), `rmt`(1M), `ufsdump`(1M), `attributes`(5), `largefile`(5)

DIAGNOSTICS

`ufsrestore` complains about bad option characters.

Read errors result in complaints. If `y` has been specified, or the user responds `y`, `ufsrestore` will attempt to continue.

If the dump extends over more than one tape, `ufsrestore` asks the user to change tapes. If the `x` or `i` function letter has been specified, `ufsrestore` also asks which volume the user wishes to mount. If the `s` modifier has been specified, and volume 1 is mounted, it is automatically positioned to the indicated file.

There are numerous consistency checks that can be listed by `ufsrestore`. Most checks are self-explanatory or can “never happen”. Common errors are given below.

`Converting to new file system format`
A dump tape created from the old file system has been loaded. It is automatically converted to the new file system format.

filename`: not found on tape`
The specified file name was listed in the tape directory, but was not found on the tape. This is caused by tape read errors while looking for the file, using a dump tape created on an active file system, or restoring a partial dump with the `r` function.

`expected next file` *inumber*`, got` *inumber*
A file that was not listed in the directory showed up. This can occur when using a dump tape created on an active file system.

`Incremental tape too low`
When doing an incremental restore, a tape that was written before the previous incremental tape, or that has too low an incremental level has been loaded.

`Incremental tape too high`
When doing incremental restore, a tape that does not begin its coverage where the previous incremental tape left off, or one that has too high an incremental level has been loaded.

`media read error: invalid argument`
Blocking factor specified for read is smaller than the blocking factor used to write data.

`Tape read error while restoring`
`Tape read error while skipping over inode inumber`
`Tape read error while trying to resynchronize`
`A tape read error has occurred`
If a file name is specified, then its contents are probably partially wrong. If an inode is being skipped or the tape is trying to resynchronize, then no extracted files have been corrupted, though files may not be found on the tape.

`resync ufsrestore, skipped` *num*
After a tape read error, `ufsrestore` may have to resynchronize itself. This message lists the number of blocks that were skipped over.

`Incorrect tape label. Expected 'foo', got 'bar'.`
The `L` option was specified, and its value did not match what was recorded in the header of the dump file.

NOTES

`ufsrestore` can get confused when doing incremental restores from dump tapes that were made on active file systems.

A `level 0` dump must be done after a full restore. Because `ufsrestore` runs in user mode, it has no control over inode allocation. This means that `ufsrestore` repositions the files, although it does not change their contents. Thus, a full dump must be done to get a new set of directories reflecting the new file positions, so that later incremental dumps will be correct.

NAME

unshare – make local resource unavailable for mounting by remote systems

SYNOPSIS

unshare [-F *FSType*] [-o *specific_options*] [*pathname* | *resourcename*]

DESCRIPTION

The unshare command makes a shared local resource unavailable as file system type *FSType*. If the option -F*FSType* is omitted, then the first file system type listed in file /etc/dfs/fstypes will be used as the default. *Specific_options*, as well as the semantics of *resourcename*, are specific to particular distributed file systems.

OPTIONS

-F *FSType* — Specify the file system type.

-o *specific_options* — Specify options specific to the file system provided by the -F option.

FILES

/etc/dfs/fstypes

/etc/dfs/sharetab

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

share(1M), shareall(1M), attributes(5)

NOTES

If *pathname* or *resourcename* is not found in the shared information, an error message will be sent to standard error.

When an unshare command completes successfully, a client mounting a file system specified in that unshare command no longer has access to that file system.

NAME

unshare_nfs – make local NFS file systems unavailable for mounting by remote systems

SYNOPSIS

unshare [-F nfs] *pathname*

DESCRIPTION

The unshare command makes local file systems unavailable for mounting by remote systems. The shared file system must correspond to a line with NFS as the *FSType* in the file /etc/dfs/sharetab.

OPTIONS

The following options are supported:

-F This option may be omitted if NFS is the first file system type listed in the file /etc/dfs/fstypes.

FILES

/etc/dfs/fstypes

/etc/dfs/sharetab

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnfssu

SEE ALSO

share(1M), attributes(5)

NOTES

If the file system being unshared is a symbolic link to a valid pathname, the canonical path (the path which the symbolic link follows) will be unshared.

For example, if /export/foo is a symbolic link to /export/bar (/export/foo -> /export/bar), the following unshare command will result in /export/bar as the unshared pathname (and not /export/foo):

```
example#  unshare -F nfs /export/foo
```

NAME

update_drv – modify device driver attributes

SYNOPSIS

update_drv [-f] *device_driver*

DESCRIPTION

The update_drv command is used to inform the system about attribute changes to an installed device driver. By default, the system rereads the driver.conf(4) file and reloads the driver binary module.

OPTIONS

The update_drv command has the following option:

-f Force the system to reread the driver.conf file even if the driver module cannot be unloaded.

EXIT STATUS

The following exit values are returned:

0 Successful completion.

>0 An error occurred.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

add_drv(1M), modunload(1M), driver.conf(4), attributes(5)

NOTES

driver.conf files can contain global properties shared by all driver instances. A forced update of the driver.conf file might change such global properties on the fly, which could have bad effects on attached instances. As a precautionary measure, instances that fail to detach continue to reference old driver global properties while newly attached instances reference new driver global properties.

NAME

useradd – administer a new user login on the system

SYNOPSIS

useradd [-c *comment*] [-d *dir*] [-e *expire*] [-f *inactive*] [-g *group*] [-G *group* [, *group*...]] [-m [-k *skel_dir*]] [-u *uid* [-o]] [-s *shell*] [-A *authorization* [,*authorization*...]] [-P *profile* [,*profile*...]] [-R *role* [,*role*...]] [-p *projname*] *login*

useradd -D [-b *base_dir*] [-e *expire*] [-f *inactive*] [-g *group*] [-p *projname*]

DESCRIPTION

useradd adds a new user to the /etc/passwd and /etc/shadow and /etc/user_attr files. The -A and -P options respectively assign authorizations and profiles to the user. The -R option assigns roles to a user. The -p option associates a project with a user.

useradd also creates supplementary group memberships for the user (-G option) and creates the home directory (-m option) for the user if requested. The new login remains locked until the passwd(1) command is executed.

Specifying useradd -D with the -g, -b, -f, -e, -A, -P, -p, or -R option (or any combination of these options) sets the default values for the respective fields. See the -D option, below. Subsequent useradd commands without the -D option use these arguments.

The system file entries created with this command have a limit of 512 characters per line. Specifying long arguments to several options can exceed this limit.

The login (login) and role (role) fields accept a string of no more than eight bytes consisting of characters from the set of alphabetic characters, numeric characters, period (.), underscore (_), and hyphen (-). The first character should be alphabetic and the field should contain at least one lower case alphabetic character. A warning message will be written if these restrictions are not met. A future Solaris release may refuse to accept login and role fields that do not meet these requirements.

The login and role fields must contain at least one character and must not contain a colon (:) or a newline (\n).

OPTIONS

The following options are supported:

-A *authorization* — One or more comma separated authorizations defined in auth_attr(4). Only a user or role who has grant rights to the authorization can assign it to an account.

-b *base_dir* — The default base directory for the system if -d *dir* is not specified. *base_dir* is concatenated with the account name to define the home directory. If the -m option is not used, *base_dir* must exist.

-c *comment* — Any text string. It is generally a short description of the login, and is currently used as the field for the user's full name. This information is stored in the user's /etc/passwd entry.

`-d` *dir*

The home directory of the new user. It defaults to *base_dir*/*account_name*, where *base_dir* is the base directory for new login home directories and *account_name* is the new login name.

`-D`

Display the default values for `group`, `base_dir`, `skel_dir`, `shell`, `inactive`, `expire`, `proj` and `projname`. When used with the `-g`, `-b`, `-f`, `-e`, `-A`, `-P`, `-p`, or `-R` options, the `-D` option sets the default values for the specified fields. The default values are:

group	`other` (GID of 1)
base_dir	`/home`
skel_dir	`/etc/skel`
shell	`/bin/sh`
inactive	`0`
expire	null
auths	null
profiles	null
proj	`3`
projname	`default`
roles	null

`-e` *expire*

Specify the expiration date for a login. After this date, no user will be able to access this login. The expire option argument is a date entered using one of the date formats included in the template file `/etc/datemsk`. See `getdate`(3C).

If the date format that you choose includes spaces, it must be quoted. For example, you can enter `10/6/90` or `"October 6, 1990"`. A null value (`" "`) defeats the status of the expired date. This option is useful for creating temporary logins.

`-f` *inactive*

The maximum number of days allowed between uses of a login ID before that ID is declared invalid. Normal values are positive integers. A value of `0` defeats the status.

`-g` *group*

An existing group's integer ID or character-string name. Without the `-D` option, it defines the new user's

primary group membership and defaults to the default group. You can reset this default value by invoking `useradd -D -g` *group*.

`-G` *group*
: An existing group's integer ID or character-string name. It defines the new user's supplementary group membership. Duplicates between *group* with the `-g` and `-G` options are ignored. No more than `NGROUPS_MAX` groups can be specified.

`-k` *skel_dir*
: A directory that contains skeleton information (such as `.profile`) that can be copied into a new user's home directory. This directory must already exist. The system provides the `/etc/skel` directory that can be used for this purpose.

`-m`
: Create the new user's home directory if it does not already exist. If the directory already exists, it must have read, write, and execute permissions by *group*, where *group* is the user's primary group.

`-o`
: This option allows a UID to be duplicated (non-unique).

`-P` *profile*
: One or more comma-separated execution profiles defined in `prof_attr`(4).

`-p` *projname*
: Name of the project with which the added user is associated. See the *projname* field as defined in `project`(4).

`-R` *role*
: One or more comma-separated execution profiles defined in `user_attr`(4). Roles cannot be assigned to other roles.

`-s` *shell*
: Full pathname of the program used as the user's shell on login. It defaults to an empty field causing the system to use `/bin/sh` as the default. The value of *shell* must be a valid executable file.

`-u` *uid*
: The UID of the new user. This UID must be a non-negative decimal integer below `MAXUID` as defined in `<sys/param.h>`. The UID defaults to the next available (unique) number above the highest number currently assigned. For example, if UIDs 100, 105, and 200 are assigned, the next default UID number will be 201. (UIDs from `0-99` are reserved for possible use in future applications.)

FILES

`/etc/datemsk`

`/etc/passwd`

/etc/shadow

/etc/group

/etc/skel

/usr/include/limits.h

/etc/user_attr

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

passwd(1), profiles(1), roles(1), users(1B), groupadd(1M), groupdel(1M), groupmod(1M), grpck(1M), logins(1M), pwck(1M), userdel(1M), usermod(1M), getdate(3C), auth_attr(4), passwd(4), prof_attr(4), project(4), user_attr(4), attributes(5)

DIAGNOSTICS

In case of an error, useradd prints an error message and exits with a non-zero status.

The following indicates that login specified is already in use:

```
UX: useradd: ERROR: login is already in use. Choose another.
```

The following indicates that the *uid* specified with the -u option is not unique:

```
UX: useradd: ERROR: uid uid is already in use. Choose another.
```

The following indicates that the *group* specified with the -g option is already in use:

```
UX: useradd: ERROR: group group does not exist. Choose another.
```

The following indicates that the *uid* specified with the -u option is in the range of reserved UIDs (from 0-99):

```
UX: useradd: WARNING: uid uid is reserved.
```

The following indicates that the *uid* specified with the -u option exceeds MAXUID as defined in <sys/param.h>:

```
UX: useradd: ERROR: uid uid is too big. Choose another.
```

The following indicates that the /etc/passwd or /etc/shadow files do not exist:

```
UX: useradd: ERROR: Cannot update system files - login cannot be created.
```

NOTES

The useradd utility adds definitions to only the local /etc/group, etc/passwd, /etc/passwd, /etc/shadow, /etc/project, and /etc/user_attr files. If a network name service such as NIS or NIS+ is being used to supplement the local

`/etc/passwd` file with additional entries, `useradd` cannot change information supplied by the network name service. However `useradd` will verify the uniqueness of the user name (or role) and user id and the existence of any group names specified against the external name service.

NAME

userdel – delete a user's login from the system

SYNOPSIS

`userdel` `[-r]` *login*

DESCRIPTION

The `userdel` utility deletes a user account from the system and makes the appropriate account-related changes to the system file and file system.

OPTIONS

The following options are supported:

`-r` Remove the user's home directory from the system. This directory must exist. The files and directories under the home directory will no longer be accessible following successful execution of the command.

OPERANDS

The following operands are supported:

login An existing login name to be deleted.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

`2` Invalid command syntax. A usage message for the `userdel` command is displayed.

`6` The account to be removed does not exist.

`8` The account to be removed is in use.

`10` Cannot update the `/etc/group` or `/etc/user_attr` file but the login is removed from the `/etc/passwd` file.

`12` Cannot remove or otherwise modify the home directory.

FILES

`/etc/passwd`	system password file
`/etc/shadow`	system file contain users' encrypted passwords and related information
`/etc/group`	system file containing group definitions
`/etc/user_attr`	system file containing additional user attributes

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`auths`(1), `passwd`(1), `profiles`(1), `roles`(1), `users`(1B), `groupadd`(1M), `groupdel`(1M), `groupmod`(1M), `logins`(1M), `roleadd`(1M), `rolemod`(1M), `useradd`(1M), `userdel`(1M), `usermod`(1M), `passwd`(4), `prof_attr`(4), `user_attr`(4), `attributes`(5)

NOTES The `userdel` utility only deletes an account definition that is in the local `/etc/group`, `/etc/passwd`, `/etc/shadow`, and `/etc/user_attr` file. file. If a network name service such as NIS or NIS+ is being used to supplement the local `/etc/passwd` file with additional entries, `userdel` cannot change information supplied by the network name service.

NAME

usermod – modify a user's login information on the system

SYNOPSIS

usermod [-u *uid* [-o]] [-g *group*] [-G *group* [, *group*...]] [-d *dir* [-m]] [-s *shell*] [-c *comment*] [-l *new_name*] [-f *inactive*] [-e *expire*] [-A *authorization* [, *authorization*]] [-P *profile* [, *profile*]] [-R *role* [, *role*]] *login*

DESCRIPTION

The `usermod` utility modifies a user's login definition on the system. It changes the definition of the specified login and makes the appropriate login-related system file and file system changes.

The system file entries created with this command have a limit of 512 characters per line. Specifying long arguments to several options may exceed this limit.

OPTIONS

The following options are supported:

`-A` *authorization*
: One or more comma separated authorizations as deined in `auth_attr`(4). Only a user or role who has `grant` rights to the `authorization` can assign it to an account. This replaces any existing authorization setting.

`-c` *comment*
: Specify a comment string. *comment* can be any text string. It is generally a short description of the login, and is currently used as the field for the user's full name. This information is stored in the user's `/etc/passwd` entry.

`-d` *dir*
: Specify the new home directory of the user. It defaults to *base_dir/login*, where *base_dir* is the base directory for new login home directories, and `login` is the new login.

`-e` *expire*
: Specify the expiration date for a login. After this date, no user will be able to access this login. The expire option argument is a date entered using one of the date formats included in the template file `/etc/datemsk`. See `getdate`(3C).

  For example, you may enter `10/6/90` or `October 6, 1990`. A value of `'' ''` defeats the status of the expired date.

`-f` *inactive*
: Specify the maximum number of days allowed between uses of a login ID before that login ID is declared invalid. Normal values are positive integers. A value of `0` defeats the status.

`-g` *group*
: Specify an existing group's integer ID or character-string name. It redefines the user's primary group membership.

`-G` *group*	Specify an existing group's integer "ID" "," or character string name. It redefines the user's supplementary group membership. Duplicates between *group* with the `-g` and `-G` options are ignored. No more than `NGROUPS_UMAX` groups may be specified as defined in `<param.h>`.
`-l` *new_logname*	Specify the new login name for the user. The *new_logname* argument is a string no more than eight bytes consisting of characters from the set of alphabetic characters, numeric characters, period (`.`), underline (_), and hypen (–). The first character should be alphabetic and the field should contain at least one lower case alphabetic character. A warning message will be written if these restrictions are not met. A future Solaris release may refuse to accept login fields that do not meet these requirements. The *new_logname* argument must contain at least one character and must not contain a colon (`:`) or NEWLINE (`\n`).
`-m`	Move the user's home directory to the new directory specified with the `-d` option. If the directory already exists, it must have permissions read/write/execute by *group*, where *group* is the user's primary group.
`-o`	This option allows the specified UID to be duplicated (non-unique).
`-P` *profile*	One or more comma-separated execution profiles defined in `auth_attr`(4). This replaces any existing profile setting.
`-R` *role*	One or more comma-separated execution profiles defined in `auth_attr`(4). This replaces any existing role setting.
`-s` *shell*	Specify the full pathname of the program that is used as the user's shell on login. The value of *shell* must be a valid executable file.
`-u` *uid*	Specify a new UID for the user. It must be a non-negative decimal integer less than `MAXUID` as defined in `<param.h>`. The UID associated with the user's home directory is not modified with this option; a user will not have access to their home directory until the UID is manually reassigned using `chown`(1M).

OPERANDS

The following operands are supported:

`login` An existing login name to be modified.

EXIT STATUS

In case of an error, `usermod` prints an error message and exits with one of the following values:

`2` The command syntax was invalid. A usage message for the `usermod` command is displayed.

`3` An invalid argument was provided to an option.

`4` The *uid* given with the `-u` option is already in use.

`5` The password files contain an error. `pwconv`(1M) can be used to correct possible errors. See `passwd`(4).

`6` The login to be modified does not exist, the *group* does not exist, or the login shell does not exist.

`8` The login to be modified is in use.

`9` The *new_logname* is already in use.

`10` Cannot update the `/etc/group` or `/etc/user_attr` file. Other update requests will be implemented.

`11` Insufficient space to move the home directory (`-m` option). Other update requests will be implemented.

`12` Unable to complete the move of the home directory to the new home directory.

FILES

`/etc/group`	system file containing group definitions
`/etc/datemsk`	system file of date formats
`/etc/passwd`	system password file
`/etc/shadow`	system file containing users' encrypted passwords and related information
`/etc/usr_attr`	system file containing additional user and role attributes

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`passwd`(1), `users`(1B), `chown`(1M), `groupadd`(1M), `groupdel`(1M), `groupmod`(1M), `logins`(1M), `pwconv`(1M), `roleadd`(1M), `roledel`(1M), `rolemod`(1M), `useradd`(1M), `userdel`(1M), `getdate`(3C), `auth_attr`(4), `passwd`(4), `attributes`(5)

NOTES

The `usermod` utility modifies `passwd` definitions only in the local `/etc/passwd` and `/etc/shadow` files. If a network nameservice such as `NIS` or `NIS+` is being used to supplement the local files with additional entries, `usermod` cannot change information supplied by the network nameservice. However `usermod` will verify the uniqueness of user name and user ID against the external nameservice.

The `usermod` utility uses the `/etc/datemsk` file, available with SUNWaccr, for date formatting.

NAME

utmpd – utmpx monitoring daemon

SYNOPSIS

utmpd [-debug]

DESCRIPTION

The utmpd daemon monitors the /var/adm/utmpx file. See utmpx(4) (and utmp(4) for historical information).

utmpd receives requests from pututxline(3C) by way of a named pipe. It maintains a table of processes and uses poll(2) on /proc files to detect process termination. When utmpd detects that a process has terminated, it checks that the process has removed its utmpx entry from /var/adm/utmpx. If the process' utmpx entry has not been removed, utmpd removes the entry. By periodically scanning the /var/adm/utmpx file, utmpd also monitors processes that are not in its table.

OPTIONS

-debug Run in debug mode, leaving the process connected to the controlling terminal. Write debugging information to standard output.

EXIT STATUS

The following exit values are returned:

0 Successful completion.

>0 An error occurred.

FILES

/etc/default/utmpd You can set default values for the flags listed below. For example: SCAN_PERIOD=600

- SCAN_PERIOD The number of seconds that utmpd sleeps between checks of /proc to see if monitored processes are still alive. The default is 300.
- MAX_FDS The maximum number of processes that utmpd attempts to monitor. The default value is 4096.

/var/adm/utmpx File containing user and accounting information for commands such as who(1), write(1), and login(1).

/proc Directory containing files for processes whose utmpx entries are being monitored.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

poll(2), pututxline(3C), proc(4), utmp(4), utmpx(4), attributes(5)

NAME

uucheck – check the uucp directories and permissions file

SYNOPSIS

/usr/lib/uucp/uucheck [-v] [-x *debug-level*]

DESCRIPTION

uucheck checks for the presence of the uucp system required files and directories. uucheck also does error checking of the Permissions file (/etc/uucp/Permissions).

uucheck is executed during package installation. uucheck can only be used by the super-user or uucp.

OPTIONS

The following options are supported:

-v — Give a detailed (verbose) explanation of how the uucp programs will interpret the Permissions file.

-x *debug-level* — Produce debugging output on the standard output. *debug-level* is a number from 0 to 9. Higher numbers give more detailed debugging information.

FILES

/etc/uucp/Devices

/etc/uucp/Limits

/etc/uucp/Permissions

/etc/uucp/Systems

/var/spool/locks/*

/var/spool/uucp/*

/var/spool/uucppublic/*

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWbnuu

SEE ALSO

uucp(1C), uustat(1C), uux(1C), uucico(1M), uusched(1M), attributes (5)

BUGS

The program does not check file/directory modes or some errors in the Permissions file such as duplicate login or machine name.

NAME

uucico – file transport program for the uucp system

SYNOPSIS

```
/usr/lib/uucp/uucico [-f] [-c type] [-d spool-directory] [-i interface]
     [-r role-number] [-s system-name] [-x debug-level]
```

DESCRIPTION

`uucico` is the file transport program for `uucp` work file transfers.

OPTIONS

The following options are supported:

`-f` — This option is used to "force execution" of `uucico` by ignoring the limit on the maximum number of uucicos defined in the `/etc/uucp/Limits` file.

`-c` *type* — The first field in the `Devices` file is the "Type" field. The `-c` option forces `uucico` to only use entries in the "Type" field that match the user specified `type`. The specified `type` is usually the name of a local area network.

`-d` *spool-directory* — This option specifies the directory *spool-directory* that contains the `uucp` work files to be transferred. The default spool directory is `/var/spool/uucp`.

`-i` *interface* — This option defines the *interface* used with `uucico`. The interface only affects slave mode. Known interfaces are UNIX (default), TLI (basic Transport Layer Interface), and TLIS (Transport Layer Interface with Streams modules, read/write).

`-r` *role-number* — The *role-number* `1` is used for master mode. *role-number* `0` is used for slave mode (default). When `uucico` is started by a program or `cron`, *role-number* `1` should be used for master mode.

`-s` *system-name* — The `-s` option defines the remote system (*system-name*) that `uucico` will try to contact. It is required when the role is master; *system-name* must be defined in the `Systems` file.

`-x` *debug-level* — Both `uux` and `uucp` queue jobs that will be transferred by `uucico`. These jobs are normally started by the `uusched` scheduler, for debugging purposes, and can be started manually. For example, the shell `Uutry` starts `uucico` with debugging turned on. The *debug-level* is a number between `0` and `9`. Higher numbers give more detailed debugging information.

FILES

/etc/uucp/Devconfig

/etc/uucp/Devices

/etc/uucp/Limits

/etc/uucp/Permissions

/etc/uucp/Sysfiles

/etc/uucp/Systems

/var/spool/locks/*

/var/spool/uucp/*

/var/spool/uucppublic/*

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWbnuu

SEE ALSO

uucp(1C), uustat(1C), uux(1C), Uutry(1M), cron(1M), uusched(1M), attributes(5)

NAME

uucleanup – uucp spool directory clean-up

SYNOPSIS

```
/usr/lib/uucp/uucleanup [-Ctime] [-Dtime] [-mstring] [-otime]
     [-ssystem] [-Wtime] [-xdebug-level] [-Xtime]
```

DESCRIPTION

`uucleanup` will scan the spool directories for old files and take appropriate action to remove them in a useful way:

- Inform the requester of send/receive requests for systems that can not be reached.
- Return undeliverable mail to the sender.
- Deliver `rnews` files addressed to the local system.
- Remove all other files.

In addition, there is a provision to warn users of requests that have been waiting for a given number of days (default 1 day). Note: `uucleanup` will process as if all option `times` were specified to the default values unless `time` is specifically set.

This program is typically started by the shell `uudemon.cleanup`, which should be started by `cron`(1M).

OPTIONS

`-Ctime`	Remove any `C.` files greater or equal to `time` days old and send appropriate information to the requester (default 7 days).
`-Dtime`	Remove any `D.` files greater or equal to `time` days old, make an attempt to deliver mail messages, and execute `rnews` when appropriate (default 7 days).
`-m`*string*	Include *string* in the warning message generated by the `-W` option. The default line is "See your local administrator to locate the problem".
`-otime`	Delete other files whose age is more than `time` days (default 2 days).
`-s`*system*	Execute for *system* spool directory only.
`-Wtime`	Any `C.` files equal to `time` days old will cause a mail message to be sent to the requester warning about the delay in contacting the remote. The message includes the *JOBID*, and in the case of mail, the mail message. The administrator may include a message line telling whom to call to check the problem (`-m` option) (default 1 day).
`-x`*debug-level*	Produce debugging output on standard ouput. *debug-level* is a single digit between 0 and 9; higher numbers give more detailed debugging information. (This option may not be available on all systems.)
`-Xtime`	Any `X.` files greater or equal to `time` days old will be removed. The `D.` files are probably not present (if they were, the `X.` could get executed). But if there are `D.` files, they will be taken care of by D. processing (default 2 days).

FILES

`/usr/lib/uucp`	directory with commands used by `uucleanup` internally
`/var/spool/uucp`	spool directory

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWbnuu

SEE ALSO

`uucp`(1C), `uux`(1C), `cron`(1M), `attributes`(5)

NAME

uusched – uucp file transport program scheduler

SYNOPSIS

/usr/lib/uucp/uusched [-u *debug-level*] [-x *debug-level*]

DESCRIPTION

uusched is the uucp(1C) file transport scheduler. It is usually started by the daemon *uudemon.hour* that is started by cron(1M) from an entry in user uucp's crontab file:

```
11,41 * * * * /etc/uucp/uucp/uudemon.hour
```

OPTIONS

The options are for debugging purposes only. *debug-level* is a number between 0 and 9. Higher numbers give more detailed debugging information:

The following options are supported:

-u *debug-level* Passes the -u *debug-level* option uucico(1M) as -x *debug-level.*

-x *debug-level* Outputs debugging messages from uusched.

FILES

/etc/uucp/Devices

/etc/uucp/Permissions

/etc/uucp/Systems

/var/spool/locks/*

/var/spool/uucp/*

/var/spool/uucppublic/*

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWbnuu

SEE ALSO

uucp(1C), uustat(1C), uux(1C), cron(1M), uucico(1M), attributes(5)

NAME

Uutry, uutry – attempt to contact remote system with debugging on

SYNOPSIS

/usr/lib/uucp/Uutry [-r] [-c *type*] [-x *debug-level*] *system-name*

DESCRIPTION

Uutry is a shell script that is used to invoke uucico(1M) to call a remote site. Debugging is initially turned on and is set to the default value of 5. The debugging output is put in file /tmp/*system-name.*

OPTIONS

The following options are supported:

-r — This option overrides the retry time that is set in file /var/uucp/.Status/*system-name.*

-c *type* — The first field in the Devices file is the "Type" field. The -c option forces uucico to use only entries in the "Type" field that match the user-specified type. The specified type is usually the name of a local area network.

-x *debug-level* — *debug-level* is a number from 0 to 9. Higher numbers give more detailed debugging information.

FILES

/etc/uucp/Devices

/etc/uucp/Limits

/etc/uucp/Permissions

/etc/uucp/Systems

/tmp/*system-name*

/var/spool/locks/*

/var/spool/uucp/*

/var/spool/uucppublic/*

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWbnuu

SEE ALSO

uucp(1C), uux(1C), uucico(1M), attributes(5)

NAME

uuxqt – execute remote command requests

SYNOPSIS

/usr/lib/uucp/uuxqt [-s *system*] [-x *debug-level*]

DESCRIPTION

uuxqt is the program that executes remote job requests from remote systems generated by the use of the uux command. (mail uses uux for remote mail requests). uuxqt searches the spool directories looking for execution requests. For each request, uuxqt checks to see if all the required data files are available, accessible, and the requested commands are permitted for the requesting system. The Permissions file is used to validate file accessibility and command execution permission.

There are two environment variables that are set before the uuxqt command is executed:

- UU_MACHINE is the machine that sent the job (the previous one).
- UU_USER is the user that sent the job.

These can be used in writing commands that remote systems can execute to provide information, auditing, or restrictions.

OPTIONS

The following options are supported:

-s *system* Specifies the remote *system* name.

-x *debug-level* *debug-level* is a number from 0 to 9. Higher numbers give more detailed debugging information.

FILES

/etc/uucp/Limits

/etc/uucp/Permissions

/var/spool/locks/*

/var/spool/uucp/*

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWbnuu

SEE ALSO

mail(1), uucp(1C), uustat(1C), uux(1C), uucico(1M), attributes(5)

NAME

vmstat – report virtual memory statistics

SYNOPSIS

vmstat [-cipsS] [*disks*] [*interval* [*count*]]

DESCRIPTION

vmstat reports virtual memory statistics regarding kernel thread, virtual memory, disk, trap, and CPU activity.

On MP (multi-processor) systems, vmstat averages the number of CPUs into the output. For per-processor statistics, see mpstat(1M).

vmstat only supports statistics for certain devices. For more general system statistics, use sar(1), iostat(1M), or sar(1M).

Without options, vmstat displays a one-line summary of the virtual memory activity since the system was booted.

During execution of this kernel status command, the "state" of the kernel can change. An example would be CPUs going online or offline. vmstat will report this as <<State change>>.

See *System Administration Guide: Advanced Administration* for device naming conventions for disks.

OPTIONS

The following options are supported:

-c Report cache flushing statistics. By default, report the total number of each kind of cache flushed since boot time. The types are: user, context, region, segment, page, and partial-page.

-i Report the number of interrupts per device. *count* and *interval* does not apply to the -i option.

-p Report paging activity in details. This option will display the following, respectively:

- epi Executable page-ins.
- epo Executable page-outs.
- epf Executable page-frees.
- api Anonymous page-ins.
- apo Anonymous page-outs.
- apf Anonymous page-frees.
- fpi File system page-ins.
- fpo File system page-outs.
- fpf File system page-frees.

-s Display the total number of various system events since boot. *count* and *interval* does not apply to the -s option.

`-S` Report on swapping rather than paging activity. This option will change two fields in `vmstat`'s ''paging'' display: rather than the ''re'' and ''mf'' fields, `vmstat` will report ''si'' (swap-ins) and ''so'' (swap-outs).

OPERANDS The following operands are supported:

count Specifies the number of times that the statitics are repeated. *count* does not apply to the `-i` and `-s` options.

disks Specifies which disks are to be given priority in the output (only four disks fit on a line). Common disk names are `id`, `sd`, `xd`, or `xy`, followed by a number (for example, `sd2`, `xd0`, and so forth).

interval Specifies the last number of seconds over which `vmstat` summarizes activity. This number of seconds repeats forever. *interval* does not apply to the `-i` and `-s` options.

EXAMPLES **EXAMPLE 1** Using `vmstat`

The following command displays a summary of what the system is doing every five seconds.

```
example% vmstat 5
kthr   memory          page              disk      faults        cpu
r b w swap  free re mf pi p fr de sr s0 s1 s2 s3  in  sy  cs us sy id
0 0 0 11456 4120 1  41 19 1  3  0  2  0  4  0  0  48 112 130  4 14 82
0 0 1 10132 4280 0   4 44 0  0  0  0  0 23  0  0 211 230 144  3 35 62
0 0 1 10132 4616 0   0 20 0  0  0  0  0 19  0  0 150 172 146  3 33 64
0 0 1 10132 5292 0   0  9 0  0  0  0  0 21  0  0 165 105 130  1 21 78
1 1 1 10132 5496 0   0  5 0  0  0  0  0 23  0  0 183  92 134  1 20 79
1 0 1 10132 5564 0   0 25 0  0  0  0  0 18  0  0 131 231 116  4 34 62
1 0 1 10124 5412 0   0 37 0  0  0  0  0 22  0  0 166 179 118  1 33 67
1 0 1 10124 5236 0   0 24 0  0  0  0  0 14  0  0 109 243 113  4 56 39
^Cexample%
```

The fields of `vmstat`'s display are

`kthr` Report the number of kernel threads in each of the three following states:

- `r` in run queue
- `b` blocked for resources I/O, paging, and so forth
- `w` swapped

`memory` Report on usage of virtual and real memory.

- `swap` amount of swap space currently available (Kbytes)
- `free` size of the free list (Kbytes)

`page` Report information about page faults and paging activity. The information on each of the following activities is given in units per second.

EXAMPLE 1 Using `vmstat` *(Continued)*

	`re`	page reclaims — but see the `-S` option for how this field is modified.
	`mf`	minor faults — but see the `-S` option for how this field is modified.
	`pi`	kilobytes paged in
	`po`	kilobytes paged out
	`fr`	kilobytes freed
	`de`	anticipated short-term memory shortfall (Kbytes)
	`sr`	pages scanned by clock algorithm
`disk`	Report the number of disk operations per second. There are slots for up to four disks, labeled with a single letter and number. The letter indicates the type of disk (s = SCSI, i = IPI, and so forth); the number is the logical unit number.	
`faults`	Report the trap/interrupt rates (per second).	
	`in`	interrupts
	`sy`	system calls
	`cs`	CPU context switches
`cpu`	Give a breakdown of percentage usage of CPU time. On MP systems, this is an average across all processors.	
	`us`	user time
	`sy`	system time
	`id`	idle time

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`sar`(1), `iostat`(1M), `mpstat`(1M), `sar`(1M), `attributes`(5)

System Administration Guide: Basic Administration

System Administration Guide: Advanced Administration

NOTES The sum of CPU utilization might vary slightly from 100 because of rounding errors in the production of a percentage figure.

NAME

volcopy – make an image copy of file system

SYNOPSIS

volcopy [-F *FSType*] [-V] [*generic_options*] [-o *FSType-specific_options*] *operands*

DESCRIPTION

volcopy makes a literal copy of the file system. This command may not be supported for all *FSTypes*.

OPTIONS

The following options are supported:

-F *FSType*
: Specify the *FSType* on which to operate. The *FSType* should either be specified here or be determinable from /etc/vfstab by matching the *operands* with an entry in the table. Otherwise, the default file system type specified in /etc/default/fs will be used.

-V
: Echo the complete command line, but do not execute the command. The command line is generated by using the options and arguments provided by the user and adding to them information derived from /etc/vfstab. This option should be used to verify and validate the command line.

generic_options
: Options that are commonly supported by most *FSType*-specific command modules. The following options are available:

  -a — Require the operator to respond "yes" or "no" instead of simply waiting ten seconds before the copy is made.

  -s — (Default) Invoke the DEL if wrong verification sequence.

-o *FSType-specific_options*
: Specify *FSType*-specific options in a comma separated (without spaces) list of suboptions and keyword-attribute pairs for interpretation by the *FSType*-specific module of the command.

OPERANDS

The following operands are supported:

operands
: generally include the device and volume names and are file system specific. A detailed description of the *operands* can be found on the *FSType*-specific man pages of volcopy.

EXIT STATUS

The following exit values are returned:

0	Successful file system copy
1	An error has occurred.

FILES

/etc/vfstab	list of default parameters for each file system
/etc/default/fs	default local file system type. Default values can be set for the following flags in /etc/default/fs. For example: LOCAL=ufs.
	LOCAL: The default partition for a command if no *FSType* is specified.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

labelit(1M), vfstab(4), attributes(5) Manual pages for the *FSType*-specific modules of volcopy.

NAME

volcopy_ufs – make an image copy of a ufs file system

SYNOPSIS

volcopy [-F ufs] [*generic_options*] *fsname srcdevice volname1 destdevice volname2*

DESCRIPTION

volcopy makes a literal copy of the ufs file system using a blocksize matched to the device.

OPTIONS

The following option is supported:

generic_options — options supported by the generic volcopy command. See volcopy(1M).

OPERANDS

The following operands are supported:

fsname — represents the mount point (for example, root, u1, etc.) of the file system being copied.

srcdevice or *destdevice* — the disk partition specified using the raw device (for example, /dev/rdsk/cld0s8, /dev/rdsk/cld1s8, etc.).

srcdevice and *volname1* — the device and physical volume from which the copy of the file system is being extracted.

destdevice and *volname2* — the target device and physical volume.

fsname and *volname* are limited to six or fewer characters and recorded in the superblock. *volname* may be '–' to use the existing volume name.

EXIT STATUS

The following exit values are returned:

0 — Successful file system copy.

non-zero — An error has occurred.

FILES

/var/adm/filesave.log — a record of file systems/volumes copied

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

cpio(1), dd(1M), labelit(1M), volcopy(1M), fs_ufs(4), attributes(5)

NOTES

volcopy does not support copying to tape devices. Use dd(1M) for copying to and from tape devices.

NAME

vold – Volume Management daemon to manage CD-ROM and floppy, ZIP/JAZ, and DVD-ROM devices

SYNOPSIS

/usr/sbin/vold [-n] [-t] [-v] [-f *config-file*] [-l *log-file*] [-d *root-dir*] [-L *debug-level*]

DESCRIPTION

The Volume Management daemon, `vold`, creates and maintains a file system image rooted at *root-dir* that contains symbolic names for floppies, CD-ROMs and other removable devices. The default *root-dir* is set to `/vol` if no directory is specified by the `-d` option.

`vold` reads the `/etc/vold.conf` configuration file upon startup. If the configuration file is modified later, `vold` must be told to reread the `/etc/vold.conf` file. Do this by entering

```
example# kill -HUP vold_pid
```

To tell vold to clean up and exit, the `SIGTERM` signal is used:

```
example# kill -TERM vold_pid
```

where `vold_pid` is the process ID of `vold`.

OPTIONS

The following options are supported:

`-n`	Never writeback. Volume Management updates media labels with unique information if labels are not unique. This flag keeps Volume Management from changing your media. The default setting is FALSE.
`-t`	Dump NFS trace information to the log file. The default setting is FALSE.
`-v`	Provide lots of status information to the log file. The default setting is FALSE (do not provide status info to log file).
`-d` *root-dir*	Specify an alternate root directory. The default location is `/vol`. Setting this will also cause other Volume Management utilities to use this as the default root directory.
`-f` *config-file*	Specify an alternate configuration file. The default file is `/etc/vold.conf`.
`-l` *log-file*	Specify an alternate log file. The default log file is `/var/adm/vold.log`.
`-L` *debug-level*	Change the level (verbosity) of debug messages sent to the log file. The range is `0` to `99` where `0` is nothing and `99` is everything. The default level is `0`.

ENVIRONMENT VARIABLES

`vold` sets the following environment variables to aid programs which are called when events such as `insert`, `notify`, and `eject` occur:

VOLUME_ACTION	Event that caused this program to be executed.
VOLUME_PATH	Pathname of the matched *regex* from the vold.conf file.
VOLUME_DEVICE	Device (in /vol/dev) that applies to the media.
VOLUME_NAME	Name of the volume in question.
VOLUME_USER	User ID of the user causing the event to occur.
VOLUME_SYMNAME	Symbolic name of a device containing the volume.
VOLUME_MEDIATYPE	Name of the type of media (CD-ROM, floppy or rmdisk)

FILES

/etc/vold.conf	Volume Management daemon configuration file. Directs the Volume Management daemon to control certain devices, and causes events to occur when specific criteria are met.
/usr/lib/vold/*.so.1	Shared objects called by Volume Management daemon when certain actions occur.
/var/adm/vold.log	the default log file location (see the -l option for a description).
/vol	the default Volume Management root directory.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWvolu

SEE ALSO

volcancel(1), volcheck(1), volmissing(1), rmmount(1M), rpc.smserverd(1M), rmmount.conf(4), vold.conf(4), attributes(5), volfs(7FS)

System Administration Guide: Basic Administration

NAME

wall – write to all users

SYNOPSIS

/usr/sbin/wall [-a] [-g *grpname*] [*filename*]

DESCRIPTION

`wall` reads its standard input until an end-of-file. It then sends this message to all currently logged-in users preceded by:

```
Broadcast Message from . . .
```

If *filename* is given, then the message is read in from that file. Normally, pseudo-terminals that do not correspond to rlogin sessions are ignored. Thus, when using a window system, the message appears only on the console window. However, `-a` will send the message even to such pseudo-terminals.

It is used to warn all users, typically prior to shutting down the system.

The sender must be superuser to override any protections the users may have invoked See `mesg`(1).

`wall` runs `setgid()` to the group ID `tty`, in order to have write permissions on other user's terminals. See `setuid`(2).

`wall` will detect non-printable characters before sending them to the user's terminal. Control characters will appear as a " ^ " followed by the appropriate ASCII character; characters with the high-order bit set will appear in "meta" notation. For example, '`\003`' is displayed as '`^C`' and '`\372`' as '`M-z`'.

OPTIONS

The following options are supported:

`-a`	Broadcast message to the console and pseudo-terminals.
`-g` *grpname*	Broadcast to the users in a specified group only, per the group database (see `group`(4)).

ENVIRONMENT VARIABLES

If the `LC_*` variables (`LC_CTYPE`, `LC_TIME`, `LC_COLLATE`, `LC_NUMERIC`, and `LC_MONETARY`) are not set in the environment, the operational behavior of `wall` for each corresponding locale category is determined by the value of the `LANG` environment variable. See `environ`(5). If `LC_ALL` is set, its contents are used to override both the `LANG` and the other `LC_*` variables. If none of the above variables are set in the environment, the "C" (U.S. style) locale determines how `wall` behaves.

FILES

`/dev/tty*`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`mesg`(1), `write`(1), `setuid`(2), `attributes`(5), `environ`(5)

NOTES `wall` displays "Cannot send to . . ." when the open on a user's `tty` file fails.

NAME

wbemadmin – start Sun WBEM User Manager

SYNOPSIS

/usr/sadm/bin/wbemadmin

DESCRIPTION

The `wbemadmin` utility starts Sun WBEM User Manager, a graphical user interface that enables you to add and delete authorized WBEM users and to set their access privileges. Use this application to manage access to groups of managed resources, such as disks and installed software, in the Solaris operating environment.

The `wbemadmin` utility allows you to perform the following tasks:

Manage user access rights	Use the `wbemadmin` utility to add, delete, or modify an individual user's access rights to a namespace on a WBEM-enabled system.
Manage namespace access rights	Use the `wbemadmin` utility to add, delete, or modify access rights for all users to a namespace.

The Sun WBEM User Manager displays a Login dialog box. You must log in as root or a user with write access to the `root\security` namespace to grant access rights to users. By default, Solaris users have guest privileges, which grants them read access to the default namespaces.

Managed resources are described using a standard information model called Common Information Model (CIM). A CIM object is a computer representation, or model, of a managed resource, such as a printer, disk drive, or CPU. CIM objects can be shared by any WBEM-enabled system, device, or application. CIM objects are grouped into meaningful collections called schema. One or more schemas can be stored in directory-like structures called namespaces.

All programming operations are performed within a namespace. Two namespaces are created by default during installation:

- `root\cimv2` — Contains the default CIM classes that represent objects on your system.
- `root\security` — Contains the security classes used by the CIM Object Manager to represent access rights for users and namespaces.

When a WBEM client application connects to the CIM Object Manager in a particular namespace, all subsequent operations occur within that namespace. When you connect to a namespace, you can access the classes and instances in that namespace (if they exist) and in any namespaces contained in that namespace.

When a WBEM client application accesses CIM data, the WBEM system validates the user's login information on the current host. By default, a validated WBEM user is granted read access to the Common Information Model (CIM) Schema. The CIM Schema describes managed objects on your system in a standard format that all WBEM-enabled systems and applications can interpret.

You can set access privileges on individual namespaces or for a user-namespace combination. When you add a user and select a namespace, by default the user is granted read access to CIM objects in the selected namespace. An effective way to combine user and namespace access rights is to first restrict access to a namespace. Then grant individual users read, read and write, or write access to that namespace.

You cannot set access rights on individual managed objects. However you can set access rights for all managed objects in a namespace as well as on a per-user basis.

If you log in to the root account, you can set the following types of access to CIM objects:

- Read Only — Allows read-only access to CIM Schema objects. Users with this privilege can retrieve instances and classes, but cannot create, delete, or modify CIM objects.
- Read/Write — Allows full read, write, and delete access to all CIM classes and instances.
- Write — Allows write and delete, but not read access to all CIM classes and instances.
- None — Allows no access to CIM classes and instances.

Context help is displayed in the left side of the `wbemadmin` dialog boxes. When you click on a field, the help content changes to describe the selected field. No context help is available on the main User Manager window.

The `wbemadmin` security administration tool updates the following Java classes in the `root\security` namespace:

- `Solaris_UserAcl` — Updated when access rights are granted or changed for a user.
- `Solaris_namespaceAcl` — Updated when access rights are granted or changed for a namespace.

USAGE

The `wbemadmin` utility is not the tool for a distributed environment. It is used for local administration on the machine on which the CIM Object Manager is running.

EXIT STATUS

The `wbemadmin` utility terminates with exit status `0`.

WARNING

The `root\security` namespace stores access privileges. If you grant other users access to the `root\security` namespace, those users can grant themselves or other users rights to all other namespaces.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWwbcor

SEE ALSO | `mofcomp`(1M), `wbemlogviewer`(1M), `init.wbem`(1M), `attributes`(5)

NAME

wbemconfig – convert a JavaSpaces datastore to the newer Reliable Log datastore format

SYNOPSIS

/usr/sadm/lib/wbem/wbemconfig convert

DESCRIPTION

A Reliable Log directory is created that contains the converted data. This directory is named /var/sadm/wbem/logr.

The convert argument is the only supported option of this command. You should only run this command after stopping WBEM (CIM Object Manager) with the init.wbem stop command. Otherwise your data may be corrupted.

This command successfully converts any proprietary custom MOFs you have created in the datastore, but not any CIM or Solaris MOFs you have modified. These will be destroyed. To recompile any modified CIM or Solaris MOFs into the new datastore, run the mofcomp command on the MOF files containing the class definitions.

Because the wbemconfig convert command invokes the JVM (Java Virtual Machine) to perform conversion of the JavaSpaces datastore, you must be running the same version of the JVM as when the original JavaSpaces storage was created. After the wbemconfig convert command is completed, you can change to any version of the JVM you want.

To see what version of the JVM you are running, issue the java -version command.

OPTIONS

The following options are supported:

convert — Convert a JavaSpaces datastore to the newer Reliable Log datastore format.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWwbcou

SEE ALSO

init.wbem(1M), wbemadmin(1M), wbemlogviewer(1M), mofcomp(1M), attributes(5)

NAME

wbemlogviewer – start WBEM Log Viewer

SYNOPSIS

/usr/sadm/bin/wbemlogviewer

DESCRIPTION

The `wbemlogviewer` utility starts the WBEM Log Viewer graphical user interface, which enables administrators to view and maintain log records created by WBEM clients and providers. The WBEM Log Viewer displays a Login dialog box. You must log in as root or a user with write access to the `root\cimv2` namespace to view and maintain log files. Namespaces are described in `wbemadmin`(1M).

Log events can have three severity levels.

- Errors
- Warnings
- Informational

The WBEM log file is created in the `/var/sadm/wbem/log` directory, with the name `wbem_log`. The first time the log file is backed up, it is renamed `wbem_log.1`, and a new `wbem_log` file is created. Each succeeding time the `wbem_log` file is backed up, the file extension number of each backup log file is increased by 1, and the oldest backup log file is removed *if* the limit, which in turn is specified in the log service settings, on the number of logfiles is exceeded. Older backup files have higher file extension numbers than more recent backup files.

The log file is renamed with a `.1` file extension and saved when one of the following two conditions occur:

- The current file reaches the specified file size limit.
- A WBEM client application uses the `clearLog()` method in the `Solaris_LogService` class to clear the current log file.
- A WBEM client application uses the `clearLog()` method in the Solaris_LogService class to clear the current log file.
- A user chooses Action->Back Up Now in the Log Viewer application.

Help is displayed in the left panel of each dialog box. Context help is not displayed in the main Log Viewer window.

USAGE

The WBEM Log Viewer is not the tool for a distributed environment. It is used for local administration.

The WBEM Log Viewer allows you to perform the following tasks:

View the logs

Set properties of log files	Click Action->Log File Settings to specify log file parameters and the log file directory.
Back up a log file	Click Action->Back Up Now to back up and close the current log file and start a new log file.

Open historical log files	Click Action->Open Log File to open a backed-up log file.
Delete an old log file	Open the file and then click Action->Delete Log File. You can only delete backed-up log files.
View log record details	Double-click a log entry or click View->Log Entry Details to display the details of a log record.
Sort the logs	Click View->Sort By to sort displayed entries. You can also click any column heading to sort the list. By default, the log entries are displayed in reverse chronological order (new logs first).

EXIT STATUS

The `wbemlogviewer` utility terminates with exit status `0`.

FILES

`/var/sadm/wbem/log/wbem_log`
WBEM log file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWwbcor

SEE ALSO

`wbemadmin`(1M), `init.wbem`(1M), `mofcomp`(1M), `attributes`(5)

NAME

whodo – who is doing what

SYNOPSIS

/usr/sbin/whodo [-h] [-l] [*user*]

DESCRIPTION

The whodo command produces formatted and dated output from information in the /var/adm/utmpx, /tmp/ps_data, and /proc/pid files.

The display is headed by the date, time, and machine name. For each user logged in, device name, user-ID and login time is shown, followed by a list of active processes associated with the user-ID. The list includes the device name, process-ID, CPU minutes and seconds used, and process name.

If *user* is specified, output is restricted to all sessions pertaining to that user.

OPTIONS

The following options are supported:

-h Suppress the heading.

-l Produce a long form of output. The fields displayed are: the user's login name, the name of the tty the user is on, the time of day the user logged in (in *hours* : *minutes*), the idle time — that is, the time since the user last typed anything (in *hours* : *minutes*), the CPU time used by all processes and their children on that terminal (in *minutes* : *seconds*), the CPU time used by the currently active processes (in *minutes* : *seconds*), and the name and arguments of the current process.

EXAMPLES

EXAMPLE 1 Using the whodo Command

The command:

```
example% whodo
```

produces a display like this:

```
Tue Mar 12 15:48:03 1985
bailey
tty09    mcn       8:51
    tty09   28158    0:29 sh

tty52    bdr      15:23
    tty52   21688    0:05 sh
    tty52   22788    0:01 whodo
    tty52   22017    0:03 vi
    tty52   22549    0:01 sh

xt162    lee      10:20
    tty08    6748    0:01 layers
    xt162    6751    0:01 sh
    xt163    6761    0:05 sh
    tty08    6536    0:05 sh
```

EXAMPLE 1 Using the whodo Command *(Continued)*

ENVIRONMENT VARIABLES

If any of the `LC_*` variables (`LC_CTYPE, LC_MESSAGES, LC_TIME, LC_COLLATE, LC_NUMERIC,` and `LC_MONETARY`) (see `environ`(5)) are not set in the environment, the operational behavior of `whodo`(1) for each corresponding locale category is determined by the value of the `LANG` environment variable. If `LC_ALL` is set, its contents are used to override both the `LANG` and the other `LC_*` variables. If none of the above variables is set in the environment, the "C" (U.S. style) locale determines how `whodo` behaves.

`LC_CTYPE` Determines how `whodo` handles characters. When `LC_CTYPE` is set to a valid value, `whodo` can display and handle text and filenames containing valid characters for that locale. The `whodo` command can display and handle Extended Unix code (EUC) characters where any individual character can be 1, 2, or 3 bytes wide. `whodo` can also handle EUC characters of 1, 2, or more column widths. In the "C" locale, only characters from ISO 8859-1 are valid.

`LC_MESSAGES` Determines how diagnostic and informative messages are presented. This includes the language and style of the messages, and the correct form of affirmative and negative responses. In the "C" locale, the messages are presented in the default form found in the program itself (in most cases, U.S. English).

`LC_TIME` Determines how `whodo` handles date and time formats. In the "C" locale, date and time handling follow the U.S. rules.

EXIT STATUS

The following exit values are returned:

`0` Successful completion.

non-zero An error occurred.

FILES

`/etc/passwd` System password file

`/tmp/ps_data`

`/var/adm/utmpx` User access and administration information

`/proc/pid`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`ps`(1), `who`(1), `attributes`(5), `environ`(5)

NAME

wracct – write extended accounting records for active processes and tasks

SYNOPSIS

/usr/sbin/wracct [-t*record_type*] *i id_list* {*task* | *process*}

DESCRIPTION

The `wracct` utility allows the administrator to invoke the extended accounting system, if active, to write intermediate records representing the resource usage of a selected set of processes or tasks. For tasks, a record_type option is also supported, allowing the administrator to request the writing of a partial record, which leaves the current task usage unchanged, or an interval record, which resets the task usage to zero for that system task, with respect to the extended accounting subsystem. If interval records are used, the total task usage is the sum of all interval records and the final record written at the task's completion. If partial records are used, the completion record reflects the total resource usage.

OPTIONS

The following options are supported:

`-i` *id_list* — Select the IDs of the tasks or processes to write records for. Specify *id_list* as a comma- or space-separated list of IDs, presented as a single argument. For some shells, this requires appropriate quoting of the argument.

`-t` *record_type* — Select type of record to write for the selected task or process. For tasks, *record_type* can be `partial` or `interval`. `partial` is the the default type, and the only type available for process records.

OPERANDS

The following operands are supported:

process — Treat the given ID as a process ID for the purposes of constructing and writing an extended accounting record.

task — Treat the given ID as a task ID for the purposes of constructing and writing an extended accounting record.

EXAMPLES

EXAMPLE 1 Writing a partial record

Write a partial record for all active `sendmail` processes.

```
# /usr/sbin/wracct -i "`pgrep sendmail`" process
```

EXAMPLE 2 Writing an interval record

Write an interval record for the task with `ID 182`.

```
# /usr/sbin/wracct -t interval -i 182 task
```

EXIT STATUS

The following exit values are returned:

`0` — Successful completion.

`1` — An error occurred.

`2` — Invalid command line options were specified.

3 Pertinent components of extended accounting facility are not active.

FILES

`/var/adm/exacct/task`

`/var/adm/exacct/proc` Extended accounting data files.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

`attributes`(5)

NAME

wrsmconf – manage WCI RSM controller configurations

SYNOPSIS

/opt/SUNWwrsm/bin/wrsmconf `create -c` *controller_id* `-f` *config_file*

wrsmconf `initial -f` *config_file* `[-c` *controller_id*`]`

wrsmconf `remove [-c` *controller_id*`]`

wrsmconf `topology [-c` *controller_id*`]`

wrsmconf `dump -c` *controller_id* `-f` *config_file*

DESCRIPTION

`wrsmconf` provides a means to create, install, retrieve, and remove configurations for WCI remote shared memory (RSM) controllers.

OPTIONS

The following options are supported:

`create -c` *controller_id* `-f` *config_file*

Create a set of per-node configurations for the specified controller and store them to the file *config_file*. The created file contains a per-node configuration for each node specified in the input for the specified controller. The file has a checksum on it and cannot be modified directly. This file can be used in a `wrsmconf initial` call on each node to install the node's configuration into the local driver.

A list of nodes and WCI devices connected to those nodes is read from standard input. For each connected pair of links, specify the nodename (`uname -n`), safari port id, and link number on both sides of the connection. The format of the information looks like this:

```
<nodename>.<wrsm-portid>.<linkno>=<nodename>.<wrsm-portid>.<linkno>
<nodename>.<wrsm-portid>.<linkno>=<nodename>.<wrsm-portid>.<linkno>
```

This interface is intended for installing configurations for testing (such as for SunVTS). Only direct-connect, non-striped configurations for 1 to 3 nodes can be specified. FM node ids and RSM hardware addresses are assigned to the specified nodes contiguously and in order starting from 0.

`initial -f` *config_file* [`-c` *controller_id*]

Install the configuration for the local node stored in the file *config_file* into the driver as the initial configuration for the specified controller. This command fails under the following circumstances:

- If *controller_id* is specified and the configuration in the file is not for the specified controller.
- If the file does not contain a valid configuration for the local node or if the checksum in the file shows it has been modified.
- If a configuration has already been installed for the controller. If this happens, use `wrsmconf remove` to remove the existing configuration.

`remove` [`-c` *controller_id*]

Disable communication through the installed configuration for all controllers or the specified controller and remove the configuration(s) from the driver.

`topology -c` *controller_id*

For each installed controller (or for the specified controller), print to stdout the set of nodes this controller is configured to reach, including the nodename, FM node id, and RSM hardware address for each node. The following is example output:

```
FM Node ID   Node Name  Controller Instance    Controller HW Addr
0            hpc00              0                    101
0            hpc00              1                    333
1            hpc01              0                    102
1            hpc01              1                    54
1            hpc01              2                    34
2            hpc03              0                    103
2            hpc03              1                    103
2            hpc03              2                    103
```

`dump -c` *controller-id* `-f` *config_file*

Fetch the installed configuration for the specified controller from the driver and store it into the file *config_file* along with a checksum to protect the data. This configuration can later be installed with the command `wrsmconf initial`.

EXIT STATUS

This command returns 0 on successful completion, and a non-zero value if an error occured.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWwrsm

SEE ALSO

`kstat`(1M), `wrsmstat`(1M), `attributes`(5)

NAME

wrsmstat – report WCI RSM driver statistics

SYNOPSIS

/opt/SUNWwrsm/bin/wrsmstat `controller [-c` *controller_id*`]`

wrsmstat `wrsm [-i` *wrsm_instance_num*`] [-v]`

wrsmstat `route [-c` *controller_id*`] [-h` *node_hostname*`]`

wrsmstat `set [-i` *wrsm_instance_num*`] -c cmmu -s` *start* `-e` *end*

DESCRIPTION

The `wrsmstat` command provides statistics on remote shared memory (RSM) controllers, routes to nodes, and WCI interfaces managed by the WCI RSM driver (`wrsm`). It also provides an interface for setting extended performance counter control registers that constrain the `wrsm` counters available through `busstat`(1M).

OPTIONS

The following options are supported:

`controller` [`-c` *controller_id*]

Displays information describing the state of the specified controller, or of all controllers if none is specified. The following is sample output:

```
$ wrsmstat controller -c 5
    Controller 5
    ------------
    Controller state: up
    Local RSM Hardware address: 0x4
    Exported segments: 0
        # published: 0
        # connections: 0
        total bound memory: 0
    Imported segments: 0
    Send Queues: 0
    Registered Handlers: 0
    Assigned WCIs: 4
    Available WCIs: 2
```

`wrsm` [`-i` *wrsm wrsm_instance_num*] [`-v`]

Displays information describing the state of the specified RSM WCI, or of all RSM WCIs if none is specified. The following is sample output:

```
$ wrsmstat wrsm -i 7
    WCI instance 7
    --------------
    Portid: 5
    Controller ID: 0
    Config Version: 5
    Link Error Shutdown Trigger: 40000
    Link #0 is not present.
    Link #1
        Link Enabled: yes
        Link State: up
        Remote RSM HW addr: 1
        Remote wnode ID: 1
        Remote link num: 1
        Remote WCI port ID: 3
```

```
        Error takedowns: 0
        Bad Config takedowns: 0
        Failed bringups: 0
        Total link errors: 0
        Maximum link errors: 0
        Average link errors: 0
        Auto shutdown enabled: yes
    Link #2 is not present.
```

If you specify the `-v` option, the following additional information is displayed:

```
Cluster Error Count:        0
Uncorrectable SRAM ECC error:   no
Maximum SRAM ECC errors:    0
Average SRAM ECC errors:    0
```

`route` [`-c` *controller_id*] [`-h` *nodename*]

Displays the route to the specified node through the specified controller. If no node is specified, displays the routes to all nodes. If no controller is specified, displays the specified node's route through all controllers. If neither is specified, displays the routes to all nodes through all controllers. The following is sample output:

```
$ wrsmstat node -c 3 -h fred
Controller 3 - Route to fred
----------------------------
Config Version: 1
FM node id: 0x345543
RSM hardware address: 0x9
Route Changes: 3
Route Type: Passthrough
Number of WCIs: 2
Stripes: 4
WCI #0
    Port ID: 3
    Instance: 0
    Num of hops: 2
    Num of links: 2
        link# 1, first hop RSM HW addr: 0x4
        link# 2, first hop RSM HW addr: 0x2
WCI #1
    Port ID: 13
    Instance: 1
    Num of hops: 2
    Num of links: 2
        link# 0, first hop RSM HW addr: 0x4
        link# 2, first hop RSM HW addr: 0x2
```

`set` [`-i` *wrsm_instance_num*] `-c cmmu -s` <*start*> `-e` <*end*>

For the specified WCI (or for each RSM WCI if none specified), configure the specified range of CMMU entries so that transactions through them are counted by *wrsm* `busstat kstats`. Each call will cause transactions to be counted through the new ranges of `cmmu` entries in addition to the previously specified ranges. To count transactions through all CMMUs, specify a start value of 0 and end value of 0. To clear all ranges (and not count transactions through any `cmmu` entries), specify a start value of 0 and end value of -1.

EXIT STATUS

This command returns 0 on successful completion, and a non-zero value if an error occurred.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWwrsm

SEE ALSO

`busstat`(1M), `kstat`(1M), `wrsmconf`(1M), `attributes`(5)

NAME

xntpd – Network Time Protocol daemon

SYNOPSIS

/usr/lib/inet/xntpd [-aAbdm] [-c *conffile*] [-e *authdelay*] [-f *driftfile*] [-k *keyfile*] [-l *logfile*] [-p *pidfile*] [-r *broadcastdelay*] [-s *statsdir*] [-t *trustedkey*] [-v *variable*] [-V *variable*]

DESCRIPTION

`xntpd` is a daemon which sets and maintains a UNIX system time-of-day in agreement with Internet standard time servers. `xntpd` is a complete implementation of the Network Time Protocol (NTP) version 3 standard, as defined by *RFC 1305*. It also retains compatibility with version 1 and 2 servers as defined by *RFC 1059* and *RFC 1119*, respectively. The computations done in the protocol and clock adjustment code are carried out with high precision and with attention to the details which might introduce systematic bias into the computations. This is done to try to maintain an accuracy suitable for synchronizing with even the most precise external time source.

Ordinarily, `xntpd` reads its configuration from a configuration file at startup time. The default configuration file name is `/etc/inet/ntp.conf`, although this may be overridden from the command line. It is also possible to specify a working, although limited, `xntpd` configuration entirely on the command line, obviating the need for a configuration file. This may be particularly appropriate when `xntpd` is to be configured as a broadcast or multicast client, with all peers being determined by listening to broadcasts at run time. Through the use of the `ntpq`(1M) program, various internal `xntpd` variables can be displayed and configuration options altered while the daemon is running.

The daemon can operate in any of several modes, including symmetric active/passive, client/server and broadcast/multicast. A broadcast/multicast client can automatically discover remote servers, compute one-way delay correction factors and configure itself automatically. This makes it possible to deploy a fleet of workstations without specifying a configuration file or configuration details specific to its environment.

OPTIONS

The following command line arguments are understood by `xntpd`. See `Configuration Commands` for a more complete functional description:

`-a`	Run in authentication mode.
`-A`	Disable authentication mode.
`-b`	Listen for broadcast NTP and sync to this if available.
`-c` *conffile*	Specify an alternate configuration file.
`-d`	Specify debugging mode. This flag may occur multiple times, with each occurrence indicating greater detail of display.
`-e` *authdelay*	Specify the time (in seconds) it takes to compute the NTP encryption field on this computer.
`-f` *driftfile*	Specify the location of the drift file.
`-k` *keyfile*	Specify the location of the file which contains the NTP authentication keys.

`-l` *logfile*	Specify a log file instead of logging to syslog.
`-m`	Listen for multicast messages and synchronize to them if available (requires multicast kernel).
`-p` *pidfile*	Specify the name of the file to record the daemon's process id.
`-r` *broadcast*	Ordinarily, the daemon automatically compensates for the network delay between the broadcast/multicast server and the client; if the calibration procedure fails, use the specified default delay (in seconds).
`-s` *statsdir*	Specify the directory to be used for creating statistics files.
`-t` *trustedkey*	Add a key number to the trusted key list.
`-v` *variable*	Add a system variable.
`-V` *variable*	Add a system variable listed by default.

USAGE

`xntpd`'s configuration file format is similar to other Unix configuration files. Comments begin with a '#' character and extend to the end of the line. Blank lines are ignored. Configuration commands consist of an initial keyword followed by a list of arguments, separated by whitespace. Some arguments may be optional. These commands may not be continued over multiple lines. Arguments may be host names, host addresses written in dotted–decimal, integers, floating point numbers (when specifying times in seconds) and text strings.

Configuration Commands

In the following descriptions, optional arguments are delimited by '[ ]', while alternatives are separated by '`|`'. The first three commands specify various time servers to be used and time services to be provided.

`peer` *host_address* [`key` #] [`version` #][`prefer`]

Specifies that the local server is to operate in "symmetric active" mode with the remote server *host_address* named in the command. In this mode, the local server can be synchronized to the remote server. In addition, the remote server can be synchronized by the local server. This is useful in a network of servers where, depending on various failure scenarios, either the local or remote server host may be the better source of time. The `peer` command, and the `server` and `broadcast` commands that follow, can take the following arguments:

`key`	Indicates that all packets sent to the address are to include authentication fields, encrypted using the specified key number. The range of this number is that of an unsigned 32 bit integer. By default, an encryption field is not included.
`version`	Specifies the version number to be used for outgoing NTP packets. Versions 1, 2, and 3 are the choices; version 3 is the default.
`prefer`	Marks the host as a preferred host. This host will be preferred for synchronization over other comparable hosts.

server *host_address* [key #] [version f1#]
[prefer] [mode f1#] server
Specifies that the local server is to operate in "client" mode with the remote server named in the command. In this mode the local server can be synchronized to the remote server, but the remote server can never be synchronized to the local server.

broadcast *host_address* [key #] [version #] [ttl #]
Specifies that the local server is to operate in "broadcast" mode where the local server sends periodic broadcast messages to a client population at the broadcast/multicast address named in the command. Ordinarily, this specification applies only to the local server operating as a transmitter. For operation as a broadcast client, see `broadcastclient` or `multicastclient` commands elsewhere in this document. In broadcast mode the *host_address* is usually the broadcast address on a local network or a multicast address assigned to NTP. The IANA has assigned the network, 224.0.1.1 to NTP. This is presently the only network that should be used. The following option is used only with the broadcast mode:

`ttl`	Specifies the time-to-live (TTL) to use on multicast packets. Selection of the proper value, which defaults to 127, is something of a black art and must be coordinated with the network administrator(s).

`broadcastclient`
Directs the local server to listen for broadcast messages on the local network, in order to discover other servers on the same subnet. Upon hearing a broadcast message for the first time, the local server measures the nominal network delay using a brief client/server exchange with the remote server. Then the server enters the "broadcastclient" mode, in which it listens for and synchronizes to succeeding broadcast messages. In order to avoid accidental or malicious disruption in this mode, both the local and remote servers must operate using authentication, with the same trusted key and key identifier.

`multicastclient`
[*IP address . . .*] Used in the same way as the `broadcastclient` command, but operates using IP multicasting. Support for this command requires the use of authentication. If one or more IP addresses are given, the server joins the respective multicast group(s). If none are given, the IP address assigned to NTP (224.0.1.1) is assumed.

`driftfile` *filename*
Specifies the name of the file used to record the frequency offset of the local clock oscillator. If the file exists, it is read at startup in order to set the initial frequency offset. Then the file is updated once per hour with the current offset computed by the daemon. If the file does not exist or this command is not given, the initial frequency offset is assumed to be zero. In this case, it may take some hours for the frequency to stabilize and the residual timing errors to subside. The file contains a single floating point value equal to the offset in parts-per-million (ppm). The file is updated by first writing the current drift value into a temporary file and then using `rename`(2) to replace the old version. This implies that `xntpd` must have write

permission for the directory the drift file is located in, and that file system links, symbolic or otherwise, should probably be avoided.

`enable auth | bclient | pll | monitor | stats [ ... ]`
`disable auth | bclient | pll | monitor | stats [ ... ]`

Provides a way to enable or disable various server options. To do so, execute a two word command, where the first word is `enable` or `disable` and the second is the flag. Flags not mentioned are unaffected. Flags that can be changed are described below, along with their default values.

Flag	Default	Description
auth	disable	Causes the server to synchronize with unconfigured peers only if the peer has been correctly authenticated using a trusted key and key identifier.
bclient	disable	Causes the server to listen for a message from a broadcast or multicast server. After this occurs, an association is automatically instantiated for that server. default for this flag is disable (off).
pll	enable	Enables the server to adjust its local clock. If not set, the local clock free-runs at its intrinsic time and frequency offset. This flag is useful in case the local clock is controlled by some other device or protocol and NTP is used only to provide synchronization to other clients.
monitor	disable	Enables the monitoring facility (see elsewhere).
stats	enable	Enables statistics facility filegen (see `Monitoring Commands` below).

`slewalways [ yes|no ]`

Force `xntpd` to always slew the time.

Authentication Commands

`keys` *filename* — Specifies the name of a file which contains the encryption keys and key identifiers used by `xntpd` when operating in authenticated mode. The format of this file is described later in this document.

`trustedkey` — # [# . . .] Specifies the encryption key identifiers which are trusted for the purposes of authenticating peers suitable for synchronization. The authentication procedures require that both the local and remote servers share the same key and key identifier, defined to be used for this purpose. However, different keys can be used with different servers. The arguments are 32 bit unsigned integers. Note, however, that key `0` is

fixed and globally known. If meaningful authentication is to be performed, the `0` key should not be trusted.

`controlkey #` — Specifies the key identifier to use with the `ntpq`(1M) program, which is useful to diagnose and repair problems that affect `xntpd` operation. The operation of the `ntpq` program and `xntpd` conform to those specified in *RFC 1305*. Requests from a remote `ntpq` program which affect the state of the local server must be authenticated. This requires that both the remote program and local server share a common key and key identifier. The argument to this command is a 32 bit unsigned integer. If no `controlkey` command is included in the configuration file, or if the keys don't match. These requests are ignored.

Access Control Commands

`restrict`*address* [`mask` *numeric_mask*] [*flag*] [. . .]

`xntpd` implements a general purpose address–and–mask based restriction list. The list is sorted by IP address and mask, and the list is searched in this order for matches, with the last match found defining the restriction flags associated with the incoming packets. The source address of incoming packets is used for the match, with the 32 bit address being logically and'ed with the mask associated with the restriction entry and then compared with the entry's address (which has also been and'ed with the mask) to look for a match. The "mask" argument defaults to 255.255.255.255, meaning that the "address" is treated as the address of an individual host. A default entry (address 0.0.0.0, mask 0.0.0.0) is always included and, given the sort algorithm, is always the first entry in the list. Note that, while "address" is normally given in dotted–quad format, the text string "default", with no mask option, may be used to indicate the default entry.

In the current implementation, flags always restrict access, i.e., an entry with no flags indicates that free access to the server is to be given. The flags are not orthogonal, in that more restrictive flags often make less restrictive ones redundant. The flags can generally be classed into two categories, those which restrict time service and those which restrict informational queries and attempts to do run time reconfiguration of the server.

One or more of the following flags may be specified:

`ignore` — Ignore all packets from hosts which match this entry. If this flag is specified neither queries nor time server polls will be responded to.

`noquery` — Ignore all NTP mode 7 packets (i.e., information queries and

	configuration requests) from the source. Time service is not affected.
`nomodify`	Ignore all NTP mode 7 packets which attempt to modify the state of the server (i.e., run time reconfiguration). Queries which return information are permitted.
`notrap`	Decline to provide mode 6 control message trap service to matching hosts. The trap service is a subsystem of the mode 6 control message protocol which is intended for use by remote event logging programs.
`lowpriotrap`	Declare traps set by matching hosts to be low priority. The number of traps a server can maintain is limited. The current limit is 3. Traps are usually assigned on a first come, first served basis, with later trap requestors being denied service. This flag modifies the assignment algorithm by allowing low priority traps to be overridden by later requests for normal priority traps.
`noserve`	Ignore NTP packets whose mode is other than 7. In effect, time service is denied, though queries may still be permitted.
`nopeer`	Provide stateless time service to polling hosts, but do not allocate peer memory resources to these hosts even if they otherwise might be considered useful as future synchronization partners.
`notrust`	Treat these hosts normally in other respects, but never use them as synchronization sources.
`limited`	These hosts are subject to a limitation on number of clients from the same net that will be accepted. Net in this context refers

	to the IP notion of net (class A, class B, class C, etc.). Only the first *client_limit* hosts that have shown up at the server and that have been active during the last *client_limit_period* seconds are accepted. Requests from other clients from the same net are rejected. Only time request packets are taken into account. "Private", "control", and "broadcast" packets are not subject to client limitation and therefore do not contribute to client count. A history of clients is kept using the monitoring capability of `xntpd`. Thus, monitoring is active as long as there is a restriction entry with the `limited` flag. The default value for *client_limit* is 3. The default value for *client_limit_period* is 3600 seconds. Currently both variables are not runtime configurable.
`ntpport`	This is actually a match algorithm modifier, rather than a restriction flag. Its presence causes the restriction entry to be matched only if the source port in the packet is the standard NTP UDP port (123). Both `ntpport` and `non–ntpport` may be specified. The `ntpport` is considered more specific and is sorted later in the list. Default restriction list entries, with the flags, `ignore`, `ntpport`, for each of the local host's interface addresses are inserted into the table at startup to prevent the server from attempting to synchronize to its own time. A default entry is also always present, though if it is otherwise unconfigured no flags are associated with the default entry (i.e., everything besides your own NTP server is unrestricted).

The restriction facility was added to allow the current access policies of the time servers running on the NSF net backbone to be implemented with `xntpd` as well. This facility may be useful for keeping unwanted or broken remote time servers from affecting your own. However, it should not be considered an alternative to the standard NTP authentication facility.

`clientlimit limit` — Sets *client_limit* to `limit`; allows configuration of client limitation policy. This variable defines the number of clients from the same network that are allowed to use the server.

`clientperiod period` — Sets *client_limit_period*; allows configuration of client limitation policy. This variable specifies the number of seconds after which a client is considered inactive and thus no longer is counted for client limit restriction.

Monitoring Commands

`statsdir` */directory path/*
Indicates the full path of a directory where statistics files should be created (see below). This keyword allows the (otherwise constant) filegen filename prefix to be modified for file generation sets used for handling statistics logs (see `filegen` statement below).

`statistics` *name* ...
Enables writing of statistics records. Currently, three kinds of statistics are supported. Each type is described below by giving its *name*, a sample line of data, and an explanation of each field:

`loopstats` — enables recording of loop filter statistics information. Each update of the local clock outputs a line of the following form to the file generation set named "loopstats":

```
48773  10847.650  0.0001307  17.3478  2
```

Field No.	Description
1	The date (Modified Julian day)
2	The time (seconds and fraction past UTC midnight)
3	Time offset in seconds

4	Frequency offset in parts-per-million
5	Time constant of the clock-discipline algorithm at each update of the clock

`peerstats` enables recording of peer statistics information. This includes statistics records of all peers of a NTP server and of the 1-pps signal, where present and configured. Each valid update appends a line similar to the one below, to the current element of a file generation set named "peerstats":

```
48773  10847.650  127.127.4.1  9714  -0.001605  \
     0.00000  0.00142
```

`Field No.`	`Description`
1	The date (Modified Julian Day)
2	The time (seconds and fraction past UTC midnight)
3	The peer address in dotted-quad notation
4	peer status. The status field is encoded in hex in the format described in Appendix A of the NTP specification, *RFC 1305*.
5	Offset in seconds
6	Delay in seconds
7	Dispersion in seconds

`clockstats` enables recording of clock driver statistics information. Each update received from a clock driver outputs a line of the following form to the file generation set named "clockstats":

```
49213  525.624  127.127.4.1  93  226  \
   00:08:29.606  D
```

`Field No.`	`Description`
1	The date (Modified Julian Day)
2	The time (seconds and fraction past UTC midnight)
3	The clock address in dotted-quad notation
4	The last timecode received from the clock in decoded ASCII format, where meaningful

In some clock drivers a good deal of additional information can be gathered and displayed as well.

Statistic files are managed using file generation sets (see `filegen` below). The information obtained by enabling statistics recording allows analysis of temporal properties of a `xntpd` server. It is usually only useful to primary servers or maybe main campus servers.

`filegen` *name* [`file` *filename*] [`type` *typename*][`flag` *flagval*] [`link` | `nolink`] [enable | disable]

Configures setting of generation file set *name*. Generation file sets provide a means for handling files that are continuously growing during the lifetime of a server. Server statistics are a typical example for such files. Generation file sets provide access to a set of files used to store the actual data. At any time at most one element of the set is being written to. The `type` given specifies when and how data will be directed to a new element of the set. This way, information stored in elements of a file set that are currently unused are available for administrational operations without the risk of disturbing the operation of `xntpd`. (Most important: they can be removed to free space for new data produced.)

Filenames of set members are built from three elements:

prefix This is a constant filename path. It is not subject to modifications via the `filegen` statement. It is defined by the server, usually specified as a compile time constant. It may, however, be configurable for individual file generation sets via other commands. For example, the prefix used with "loopstats" and "peerstats" filegens can be configured using the `statsdir` statement explained above.

filename This string is directly concatenated to the *prefix* mentioned above (no intervening '/' (slash)). This can be modified using the `file` argument to the `filegen` statement. No '. .' elements are allowed in this component to prevent filenames referring to parts outside the filesystem hierarchy denoted by *prefix*.

suffix This part is reflects individual elements of a file set. It is generated according to the `type` of a file set as explained below. A file generation set is characterized by its type. The following types are supported:

`none` The file set is actually a single plain file.

`pid` One element of file set is used per incarnation of a `xntpd` server. This type does not perform any changes to file set members during runtime. However it provides an easy way of separating files belonging to different `xntpd` server incarnations. The set member filename is built by appending a '.' (dot) to concatenated *prefix* and *filename* strings, and appending the decimal representation of the process id of the `xntpd` server process.

`day` One file generation set element is created per day. The term *day* is based on UTC . A day is defined as the period between 00:00 and 24:00 UTC . The file set member suffix consists of a '.' (dot) and a day specification in the form, *YYYYMMDD*. *YYYY* is a 4 digit year number (e.g., 1992). *MM* is a two digit month number. *DD* is a two digit day number. Thus, all information written at December 10th, 1992 would end up in a file named, *PrefixFilename*.19921210.

`week` Any file set member contains data related to a certain week of a year. The term *week* is defined by computing "day of year" modulo 7. Elements of such a file generation set are distinguished by appending the following

suffix to the file set filename base: a dot, a four digit year number, the letter 'W', and a two digit week number. For example, information from January, 5th 1992 would end up in a file with suffix ".1992W1".

`month` One generation file set element is generated per month. The file name suffix consists of a dot, a four digit year number, and a two digit month.

`year` One generation file elment is generated per year. The filename suffix consists of a dot and a 4 digit year number.

`age` This type of file generation sets changes to a new element of the file set every 24 hours of server operation. The filename suffix consists of a dot, the letter 'a', and an eight digit number. This number is taken to be the number of seconds the server is running at the start of the corresponding 24 hour period.

Information is only written to a file generation set when this set is `enabled`. Output is prevented by specifying, `disabled`.

It is convenient to be able to access the current element of a file generation set by a fixed name. This feature is enabled by specifying `link` and disabled using `nolink`. If `link` is specified, a hard link from the current file set element to a file without suffix is created. When there is already a file with this name and the number of links of this file is one, it is renamed appending a dot, the letter, 'C', and the pid of the `xntpd` server process. When the number of links is greater than one, the file is unlinked. This allows the current file to be accessed by a constant name.

Miscellaneous Commands

`broadcastdelay` *seconds*
The broadcast and multicast modes require a special calibration to determine the network delay between the local and remote servers. Ordinarily, this is done automatically by the initial protocol exchanges between the local and remote servers. In some cases, the calibration procedure may fail due to, for example, network or server access controls. This command specifies the default delay to be used under these circumstances. Typically (for Ethernet), a number between 0.003 and 0.007 is appropriate for *seconds*. When this command is not used, the default is 0.004 seconds.

`trap` *host_address* [`port` *port_number*]
[`interface` *interface_address*]
Configures a trap receiver at the given *host_address* and *port_number* for sending messages with the specified local *interface_address.* If the port number is unspecified, a value of 18447 is used. If the interface address is not specified, the message is sent with the source address of the local interface the message is sent through. On a multi-homed host, the interface used may change with routing changes.

C information from the server in a log file. While such monitor programs may also request their own trap dynamically, configuring a trap receiver ensures that no messages are lost when the server is started.

`setvar` *variable* [*default*]
This command adds an additional system variable. Variables like this can be used to distribute additional information such as the access policy. If the variable of the form, *variable_name=value* is followed by the *default* keyword, the variable will be listed as one of the default system variables (see the `ntpq`(1M) command). Additional variables serve informational purposes only. They can be listed; but they are not related to the protocol. The known protocol variables always override any variables defined via the `setvar` mechanism.

Three special variables contain the names of all variable of the same group. *sys_var_list* holds the names of all system variables. *peer_var_list* holds the names of all peer variables. And *clock_var_list* hold the names of the reference clock variables.

`monitor` [`yes` | `no`]
`authenticate` [`yes` | `no`]
These commands have been superseded by the `enable` and `disable` commands. They are listed here for historical purposes.

`logconfig` *configkeyword*
Controls the amount of output written to syslog or the logfile. By default all output is turned on. *configkeyword* is formed by concatenating the message class with the event class. It is permissible to use the prefix, `all`, instead of a message class. A message class may also be followed by the keyword, `all`, meaning to enable/disable all of the respective message class. All *configkeywords* can be prefixed with the symbols, '=', '+' and '–' . Here, '=' sets the syslogmask, '+' adds messages, and '–' removes messages. Syslog messages can be controlled in four classes: *sys*, *peer*, *clock*, `sync`. Within these classes four types of messages can be controlled. Each is described below, along with its `configkeyword`:

`Configkeyword`	`Message type`
`info`	Informational messages control configuration information.
`events`	Event messages control logging of events (reachability, synchronization, alarm conditions).
`statistics`	Statistical messages control statistical output.
`status`	Status messages describe mainly the synchronization status.

A minimal log configuration might look like this:

```
logconfig =syncstatus +sysevents
```

A configuration like this lists, just the synchronization state of `xntp` and the major system events. For a simple reference server, the following minimum message configuration could be useful:

```
logconfig =syncall +clockall
```

This configuration lists all clock information and synchronization information. All other events and messages about peers, system events and so on, is suppressed.

Authentication Key File Format

The NTP standard specifies an extension to allow verification of the authenticity of received NTP packets, and to provide an indication of authenticity in outgoing packets. This is implemented in `xntpd` using the DES or MD5 algorithms to compute a digital signature, or message-digest. The specification allows any one of possibly 4 billion keys, numbered with 32 bit key identifiers, to be used to authenticate an association. The servers involved in an association must agree on the key and key identifier used to authenticate their data. However they must each learn the key and key identifier independently. In the case of DES, the keys are 56 bits long with, depending on type, a parity check on each byte. In the case of MD5, the keys are 64 bits (8 bytes). `xntpd` reads its keys from a file specified using the `-k` command line option or the `keys` statement in the configuration file. While key number 0 is fixed by the NTP standard (as 56 zero bits) and may not be changed, one or more of the keys numbered 1 through 15 may be arbitrarily set in the keys file.

The key file uses the same comment conventions as the configuration file. Key entries use a fixed format of the form, *keyno type key*. Here, *keyno* is a positive integer, `type` is a single character which defines the format the key is given in, and *key* is the key itself.

The *key* may be given in one of several different formats, controlled by the `type` character. The different key types, and corresponding formats, are described below:

`Key:` S

`Format:` A 64 bit hexadecimal number in DES format

In this format, the high order 7 bits of each octet are used to form the 56 bit key while the low order bit of each octet is given a value such that odd parity is maintained for the octet. Leading zeroes must be specified (i.e., the key must be exactly 16 hex digits long) and odd parity must be maintained. Hence a zero key, in standard format, would be given as: 0101010101010101.

`Key:` N

`Format:` A 64 bit hexadecimal number in NTP format

This format is the same as the DES format except the bits in each octet have been rotated one bit right so that the parity bit is now the high order bit of the octet. Leading zeroes must be specified and odd parity must be maintained. A zero key in NTP format would be specified as: `8080808080808080`.

`Key:` A

`Format:` A 1–to–8 character ASCII string

A key is formed from this by using the lower order 7 bits of the ASCII representation of each character in the string. Zeroes are added on the right when necessary to form a full width 56 bit key.

`Key:` M

`Format:` A 1–to–8 character ASCII string, using the MD5 authentication scheme.

Note that both the keys and the authentication schemes (DES or MD5) must be identical between a set of peers sharing the same key number.

Primary Clock Support

`xntpd` has been built to be compatible with all supported types of reference clocks. A reference clock is generally (though not always) a radio timecode receiver which is synchronized to a source of standard time such as the services offered by the NRC in Canada and NIST in the U.S. The interface between the computer and the timecode receiver is device dependent and will vary, but it is often a serial port.

For the purposes of configuration, `xntpd` treats reference clocks in a manner analogous to normal NTP peers as much as possible. Reference clocks are referred to by address, much as a normal peer is. However, an invalid IP address is used to distinguish them from normal peers. Reference clock addresses are of the form *127.127.t.u* where *t* is an integer denoting the clock type and *u* indicates the type–specific unit number. Reference clocks are configured using a `server` statement in the configuration file where the *host_address* is the clock address. The `key`, `version` and `ttl` options are not used for reference clock support. Some reference clocks require a `mode` option to further specify their operation. The `prefer` option can be useful to persuade the server to cherish a reference clock with somewhat more enthusiasm than other reference clocks or peers. Clock addresses may generally be used anywhere in the configuration file that a normal IP address can be used. For example, they can be used in `restrict` statements, although such use would normally be considered strange.

Reference clock support provides the `fudge` command, which can be used to configure reference clocks in special ways. The generic format that applies to this command is,

```
fudge 127.127.t.u [  time1 secs ] [  time2 secs ]\
     [  stratum int ] [  refid int ] \
     [  flag1 0 | 1 ] [  flag2 0 | 1 ] [  flag3 0 | 1 ] [  flag4  0 | 1 ]
```

with options described as follows:

time1
time2 Are specified in fixed point seconds and used in some clock drivers as calibration constants. By convention, and unless indicated otherwise, `time1` is used as a calibration constant to adjust the nominal time offset of a particular clock to agree with an external standard, such as a precision PPS signal. The specified offset is in addition to the propagation delay provided by other means, such as internal DIP switches.

stratum Is a number in the range zero to 15 and is used to assign a nonstandard operating stratum to the clock.

refid Is an ASCII string in the range one to four characters and is used to assign a nonstandard reference identifier to the clock.

flag1
flag2
flag3
flag4 Are binary flags used for customizing the clock driver. The interpretation of these values, and whether they are used at all, is a function of the needs of the particular clock driver. However, by convention, and unless indicated otherwise, *flag3* is used to attach the ppsclock streams module to the configured driver, while *flag4* is used to enable recording verbose monitoring data to the clockstats file configured with the `filegen` command. Further information on the ppsclock streams module is in the `README` file in the `./kernel` directory in the current xntp3 program distribution. Further information on this feature is available in the `./scripts/stats` directory in the same distribution.

Ordinarily, the stratum of a reference clock is zero, by default. Since the `xntpd` daemon adds one to the stratum of each peer, a primary server ordinarily displays stratum one. In order to provide engineered backups, it is often useful to specify the reference clock stratum as greater than zero. The `stratum` option is used for this purpose. Also, in cases involving both a reference clock and a 1-pps discipline signal, it is useful to specify the reference clock identifier as other than the default, depending on the driver. The *refid* option is used for this purpose. Except where noted, these options apply to all clock drivers.

`xntpd` on Unix machines currently supports several different types of clock hardware. It also supports a special pseudo–clock used for backup or when no other clock source is available. In the case of most of the clock drivers, support for a 1-pps precision timing signal is available as described in the `README` file in the `./doc` `directory of the` `xntp3` program distribution. The clock drivers, and the addresses used to configure them, are described in the file, `README.refclocks`, in the doc directory of the current program distribution.

Variables

Most variables used by the NTP protocol can be examined with `ntpq` (mode 6 messages). Currently very few variables can be modified via mode 6 messages. These variables are either created with the `setvar` directive or the leap warning variables. The leap warning bits that can be set in the *leapwarning* variable (up to one month ahead). Both, the *leapwarning* and in the *leapindication* variable, have a slightly different encoding than the usual `leap` bits interpretation:

`00` The daemon passes the leap bits of its synchronization source (usual mode of operation).

`01/10` A leap second is added/deleted (operator forced leap second).

`11` Leap information from the synchronization source is ignored (thus LEAP_NOWARNING is passed on).

FILES

`/etc/inet/ntp.conf` Default name of the configuration file

`/etc/ntp/ntp.drift`	Conventional name of the drift file
`/etc/inet/ntp.keys`	Conventional name of the key file
`/etc/inet/ntp.server`	Sample server configuration file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWntpu

SEE ALSO

`ntpdate`(1M), `ntpq`(1M), `ntptrace`(1M), `xntpdc`(1M), `rename`(2), `attributes`(5)

NAME

xntpdc – special NTP query program

SYNOPSIS

xntpdc [-ilnps] [-c *command*] [*host*] [...]

DESCRIPTION

xntpdc queries the xntpd daemon about its current state and requests changes in that state. You can run xntpdc in interactive mode or in controlled using command line arguments.

Extensive state and statistics information is available through the xntpdc interface. In addition, nearly all the configuration options which can be specified at start up using xntpd's configuration file may also be specified at run time using xntpdc.

If one or more request options is included on the command line when xntpdc is executed, each of the requests is sent to the NTP servers running on each of the hosts given as command line arguments, or on the local host by default. If no request options are given, xntpdc attempts to read commands from the standard input and execute these on the NTP server running on the first host specified on the command line, again defaulting to the local host when no other host is specified. xntpdc prompts for commands if the standard input is a terminal device.

xntpdc uses NTP mode 7 packets to communicate with the NTP server, and can be used to query any compatable server on the network which permits it. As NTP is a UDP protocol, this communication is somewhat unreliable, especially over large distances. xntpdc does not attempt to re-transmit requests, and times requests out if the remote host is not heard from within a suitable timeout time.

The operation of xntpdc is specific to the particular implementation of the xntpd daemon. You can expect xntpdc to work only with this and maybe some previous versions of the daemon. Requests from a remote xntpdc program that affect the state of the local server must be authenticated. This requires that both the remote program and local server share a common key and key identifier.

OPTIONS

xntpdc reads interactive format commands from the standard input. If you specify the -c, -l, -p or -s option, the specified queries are sent to the hosts immediately.

The following command line options are supported:

-c *command* . . .
: Add *command* to the list of commands to execute on the specified hosts. *command* is interpreted as an interactive format command.

  Multiple -c options may be specified.

-i
: Force xntpdc to operate in interactive mode.

  Prompts are written to the standard output. Commands are read from the standard input.

-l
: Obtain a list of peers which are known to the servers.

  This option is equivalent to -c listpeers. See listpeers in Control Message Commands.

`-n`	Output all host addresses in dotted-quad numeric format rather than converting to the canonical host names.
`-p`	Print a list of the peers known to the server as well as a summary of their state. This option is equivalent to `-c peers`. See `peers` in `Control Message Commands`.
`-s`	Print a list of the peers known to the server as well as a summary of their state, but in a slightly different format than the `-p` option. This option is equivalent to `-c dmpeers`. See `dmpeers` in `Control Message Commands`.

OPERANDS

The following operands are supported:

Interactive Commands

The interactive commands consist of a keyword (*command_keyword*) followed by zero to four arguments. You need to entry only enough characters of the *command_keyword* to uniquely identify it. The output of an interactive command is sent to the standard output by default. You can send the output of an interactive command to a file by appending a `<`, followed by a file name, to the command line.

A number of interactive format commands are executed entirely within the `xntpdc` program itself and do not result in NTP mode.

The following interactive commands are supported:

? [*command_keyword*]	Without an argument, print a list of `ntpq` command keywords. If *command_keyword* is specified, print function and usage information about the *command_keyword*.
delay *milliseconds*	Specify a time interval to add to timestamps included in requests which require authentication. This enables (unreliable) server reconfiguration over long delay network paths or between machines whose clocks are unsynchronized. Because the server no longer requires timestamps in authenticated requests, this command may be obsolete.
help [*command_keyword* **]**	Without an argument, print a list of `ntpq` command keywords. If *command_keyword* is specified, print function and usage information about the *command_keyword*.

host *hostname*	Set the host (*hostname*) to which future queries are sent. Specify *hostname* as a host name or a numeric address.
hostnames [`yes` | `no`]	Print hostnames or numeric addresses in information displays. Specify `yes` to print host names. Specify `no` to print numeric addresses. The default is `yes`, unless the `-n` command line option is specified.
keyid *keyid*	Enable specification of a key number (*keyid*) to authenticate configuration requests. *keyid* must correspond to a key number the server has been configured to use for this purpose.
passwd	Prompt user to enter a password to authenticate configuration requests. The password is not displayed , and must correspond to the key configured for use by the NTP server for this purpose. If the password does not correspond to the key configured for use by the NTP server, requests are not successful.
quit	Exit `xntpdc`.
timeout *millseconds*	Specify a timeout period for responses to server queries. The default is approximately `8000` milliseconds. As `xntpdc` retries each query once after a timeout, the total waiting time for a timeout is twice the timeout value set.

Control Message Commands

Query commands result in `NTP` mode 7 packets containing requests for information being sent to the server. These control message commands are read-only commands in that they make no modification of the server configuration state.

The following control message commands are supported:

clkbug
  Obtain debugging information for a reference clock driver. This information is provided only by some clock drivers.

clockinfo *clock_peer_address* [...]
  Obtain and print information concerning a peer clock.

The values obtained provide information on the setting of fudge factors and other clock performance information.

dmpeers
Obtain a list of peers for which the sserver is maintaining state, along with a summary of that state.

The peer summary list is identical to the output of the `peers` command, except for the character in the leftmost column. Characters only appear beside peers which were included in the final stage of the clock selection algorithm. A `.` indicates that this peer was cast off in the falseticker detection, while a `+` indicates that the peer made it through. A `*` denotes the peer with which the server is currently synchronizing.

iostats
Print statistics counters maintained in the input-output module.

kerninfo
Obtain and print kernel phase-lock loop operating parameters.

This information is available only if the kernel has been specially modified for a precision timekeeping function.

listpeers
Obtain and print a brief list of the peers for which the server is maintaining state.

These should include all configured peer associations as well as those peers whose stratum is such that they are considered by the server to be possible future synchonization candidates. candidates.

loopinfo [oneline | multiline]
Print the values of selected loop filter variables.

The loop filter is the part of NTP which deals with adjusting the local system clock.

The `oneline` and `multiline` options specify the format in which this information is printed. `multiline` is the default.

The offset is the last offset given to the loop filter by the packet processing code. The frequency is the frequency error of the local clock in parts-per-million (ppm). The `time_const` controls the stiffness of the phase-lock loop and thus the speed at which it can adapt to oscillator drift. The watchdog timer value is the number of seconds which have elapsed since the last sample offset was given to the loop filter.

memstats
Print statistics counters related to memory allocation code.

monlist [version]
Obtain and print traffic counts collected and maintained by the monitor facility. The version number should not normally need to be specified.

peers
Obtain a list of peers for which the server is maintaining state, along with a summary of that state.

The following summary information is included:

- Address of the remote peer.
- Local interface address. If a local address has yet to be determined it is `0.0.0.0`.
- Stratum of the remote peer. A stratum of `16` indicates the remote peer is unsynchronized.
- Polling interval, in seconds.
- Reachability register, in octal.
- Current estimated delay, offset and dispersion of the peer, in seconds.
- Mode in which the peer entry is operating.

  This is represented by the character in the left margin. A `+` denotes symmetric active, a `-` indicates symmetric passive, a `=` means the remote server is being polled in client mode, a `^` indicates that the server is broadcasting to this address, a `~` denotes that the remote peer is sending broadcasts and a `*` marks the peer the server is currently synchonizing to.
- Host.

  This field may contain a host name, an IP address, a reference clock implementation name with its parameter or `REFCLK` (implementation number, parameter). On hostnames no only IP-addresses is displayed.

pstats *peer_address* [...]
Show the per-peer statistic counters associated with the specified peers.

reslist
Obtain and print the server's restriction list.

Generally, this list is printed in sorted order.

showpeer *peer_address* [...]
Show a detailed display of the current peer variables for one or more peers. Most of these values are described in the NTP Version 2 specification.

sysinfo
Print a variety of system state variables that are related to the local server.

The output from `sysinfo` is described in NTP Version 3 specification, RFC-1305. All except the last four lines are described in the NTP Version 3 specification, RFC-1305.

The `system flags` show various system flags, some of which can be set and cleared by the `enable` and `disable` configuration commands, respectively. These are the `auth`, `bclient`, `monitor`, `pll`, `pps` and `stats` flags. See the xntpd

documentation for the meaning of these flags. There are two additional flags which are read only, the `kernel_pll` and `kernel_pps`. These flags indicate the synchronization status when the precision time kernel modifications are in use. The `kernel_pll` indicates that the local clock is being disciplined by the kernel, while the `kernel_pps` indicates the kernel discipline is provided by the `PPS` signal. The stability is the residual frequency error remaining after the system frequency correction is applied and is intended for maintenance and debugging. In most architectures, this value initially decreases from as high as `500` ppm to a nominal value in the range `.01` to `0.1` ppm. If it remains high for some time after starting the daemon, something may be wrong with the local clock, or the value of the kernel variable tick may be incorrect. The `broadcastdelay` shows the default broadcast delay, as set by the `broadcastdelay` configuration command. The `authdelay` shows the default authentication delay, as set by the `authdelay` configuration command.

sysstats
Print statistics counters maintained in the protocol module.

timerstats
Print statistics counters maintained in the timer/event queue support code.

Runtime Configuration Requests

The server authenticates all requests that cause state changes in the server. The server uses a configured NTP key to accomplish this. This facility can also be disabled by the server by not configuring a key).

You must make the key number and the corresponding key known to `xtnpdc`. Use the `keyid` or `passwd` commands to do so.

The `passwd` command prompts users for a password to use as the encryption key. It also prompts automatically for both the key number and password the first time a command which would result in an authenticated request to the server is given. Authentication provides verification that the requester has permission to make such changes. It also gives an extra degree of protection against transmission errors.

Authenticated requests always include a time stamp in the packet data. The time stamp is included in the computation of the authentication code. This timestamp is compared by the server to its receive time stamp. If the time stamps differ by more than a small amount the request is rejected.

Time stamps are rejected for two reasons. First, it makes simple replay attacks on the server, by someone who might be able to overhear traffic on your LAN, much more difficult. Second, it makes it more difficult to request configuration changes to your server from topologically remote hosts.

While the reconfiguration facility works well with a server on the local host, and may work adequately between time-synchronized hosts on the same LAN, it works very poorly for more distant hosts. If reasonable passwords are chosen, care is taken in the distribution and protection of keys and appropriate source address restrictions are applied, the run time reconfiguration facility should provide an adequate level of security.

The following commands make authenticated requests.

addpeer *peer_address* [*keyid*] [*version*] [*prefer*]
Add a configured peer association at the given address and operating in symmetric active mode. An existing association with the same peer may be deleted when this command is executed, or may simply be converted to conform to the new configuration, as appropriate.

If the optional *keyid* is a non-zero integer, all outgoing packets to the remote server will have an authentication field attached encrypted with this key. If the *keyid* is `0` or omitted, no authentication is done.

Specify *version* as `1`, `2` or `3`. The default is `3`.

The *prefer* keyword indicates a preferred peer. This keyword is used primarily for clock synchronisation if possible. The preferred peer also determines the validity of the PPS signal - if the preferred peer is suitable for synchronisation so is the PPS signal.

addserver *peer_address* [*keyid*] [*version*] [*prefer*]
Identical to the `addpeer` command, except that the operating mode is client.

addtrap [*address* [*port*] [*interface*]
Set a trap for asynchronous messages.

authinfo
Return information concerning the authentication module, including known keys and counts of encryptions and decryptions which have been done.

broadcast *peer_address* [*keyid*] [*version*] [*prefer*]
Identical to the `addpeer` command, except that the operating mode is `broadcast`. In this case a valid key identifier and key are required. The `peer_address` parameter can be the broadcast address of the local network or a multicast group address assigned to NTP. If a multicast address, a multicast-capable kernel is required.

clrtrap [*address* [*port*] [*interface*]
Clear a trap for asynchronous messages.

delrestrict *address mask* [*ntpport*]
Delete the matching entry from the restrict list.

fudge *peer_address* [*time1*] [*time2*] [*stratum*] [*refid*]
Provide a way to set certain data for a reference clock.

readkeys
Cause the current set of authentication keys to be purged and a new set to be obtained by re-reading the `keys` file. The `keys` file must have been specified in the `xntpd` configuration file. This enables encryption keys to be changed without restarting the server.

restrict *address mask flag* [*flag*]
This command operates in the same way as the restrict configuration file commands of `xntpd`.

reset
Clear the statistics counters in various modules of the server.

traps
Display the traps set in the server.

trustkey *keyid* [...]
untrustkey *keyid* [...]
These commands operate in the same way as the `trustedkey` and `untrustkey` configuration file commands of `xntpd`.

unconfig *peer_address* [...]
Cause the configured bit to be removed from the specified peers. In many cases this causes the peer association to be deleted. When appropriate, however, the association may persist in an unconfigured mode if the remote peer is willing to continue on in this fashion.

unrestrict *address mask flag* [*flag*]
Unrestrict the matching entry from the restrict list.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWntpu

SEE ALSO

`ntpdate`(1M), `ntpq`(1M), `ntptrace`(1M), `xntpd`(1M), `rename`(2), `attributes`(5)

NAME

ypbind – NIS binder process

SYNOPSIS

/usr/lib/netsvc/yp/ypbind [-broadcast | -ypset | -ypsetme]

DESCRIPTION

NIS provides a simple network lookup service consisting of databases and processes. The databases are stored at the machine that runs an NIS server process. The programmatic interface to NIS is described in `ypclnt`(3NSL). Administrative tools are described in `ypinit`(1M), `ypwhich`(1), and `ypset`(1M). Tools to see the contents of NIS maps are described in `ypcat`(1), and `ypmatch`(1).

`ypbind` is a daemon process that is activated at system startup time from the startup script `/etc/init.d/rpc`. By default, it is invoked as `ypbind -broadcast`. `ypbind` runs on all client machines that are set up to use NIS. See `sysidtool`(1M). The function of `ypbind` is to remember information that lets all NIS client processes on a node communicate with some NIS server process. `ypbind` must run on every machine which has NIS client processes. The NIS server may or may not be running on the same node, but must be running somewhere on the network. If the NIS server is a NIS+ server in NIS (YP) compatibility mode, see the `NOTES` section of the `ypfiles`(4)man page for more information.

The information `ypbind` remembers is called a *binding* — the association of a domain name with a NIS server. The process of binding is driven by client requests. As a request for an unbound domain comes in, if started with the `-broadcast` option, the `ypbind` process broadcasts on the net trying to find an NIS server, either a `ypserv` process serving the domain or an `rpc.nisd` process in "YP-compatibility mode" serving NIS+ directory with name the same as (case sensitive) the domain in the client request. Since the binding is established by broadcasting, there must be at least one NIS server on the net. If started without the `-broadcast` option, `ypbind` process steps through the list of NIS servers that was created by `ypinit -c` for the requested domain. There must be an NIS server process on at least one of the hosts in the NIS servers file. All the hosts in the NIS servers file must be listed in either the `/etc/hosts` or `/etc/inet/ipnodes` files along with their IP addresses. Once a domain is bound by `ypbind`, that same binding is given to every client process on the node. The `ypbind` process on the local node or a remote node may be queried for the binding of a particular domain by using the `ypwhich`(1) command.

If `ypbind` is unable to speak to the NIS server process it is bound to, it marks the domain as unbound, tells the client process that the domain is unbound, and tries to bind the domain once again. Requests received for an unbound domain will wait until the requested domain is bound. In general, a bound domain is marked as unbound when the node running the NIS server crashes or gets overloaded. In such a case, `ypbind` will try to bind to another NIS server using the process described above.`ypbind` also accepts requests to set its binding for a particular domain. The request is usually generated by the `ypset`(1M) command. In order for `ypset` to work, `ypbind` must have been invoked with flags `-ypset` or `-ypsetme`.

OPTIONS

`-broadcast` Send a broadcast datagram using UDP/IP that requests the information needed to bind to a specific NIS server.

This option is analogous to `ypbind` with no options in earlier Sun releases and is recommended for ease of use.

`-ypset` Allow users from any remote machine to change the binding by means of the `ypset` command. By default, no one can change the binding. This option is insecure.

`-ypsetme` Only allow `root` on the local machine to change the binding to a desired server by means of the `ypset` command. `ypbind` can verify the caller is indeed a `root` user by accepting such requests only on the loopback transport. By default, no external process can change the binding.

FILES

`/var/yp/binding/`*ypdomain*`/ypservers`

`/etc/inet/hosts`

`/etc/inet/ipnodes`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`ypcat`(1), `ypmatch`(1), `ypwhich`(1), `ifconfig`(1M), `rpc.nisd`(1M), `ypinit`(1M), `ypset`(1M), `ypclnt`(3NSL), `hosts`(4), `ipnodes`(4), `ypfiles`(4), `attributes`(5)

NOTES

`ypbind` supports multiple domains. The `ypbind` process can maintain bindings to several domains and their servers, the default domain is the one specified by the `domainname`(1M) command at startup time.

The `-broadcast` option works only on the UDP transport. It is insecure since it trusts "any" machine on the net that responds to the broadcast request and poses itself as an NIS server.

NAME

ypinit – set up NIS client

SYNOPSIS

/usr/sbin/ypinit [-c] [-m] [-s *master_server*]

DESCRIPTION

ypinit can be used to set up an NIS client system. You must be the superuser to run this command. This script need not be used at all if ypbind(1M) is started with the -broadcast option (it is invoked with this option from the start up script /etc/init.d/rpc).

Normally, ypinit is run only once after installing the system. It may be run whenever a new NIS server is added to the network or an existing one is decommissioned.

ypinit prompts for a list of NIS servers to bind the client to; this list should be ordered from the closest to the furthest server. Each of these NIS servers must be listed in either the /etc/hosts or the /etc/inet/ipnodes file along with its IP address. ypinit stores the list in file /var/yp/binding/*domain*/ypservers. This file is used by ypbind when run without the -broadcast option.

OPTIONS

-c	Set up a ypclient system.
-m	Build a master ypserver data base.
-s *master_server*	Slave data base. *master_server* must be the same master configured in the YP maps and returned by the ypwhich -m command.

FILES

/etc/hosts

/etc/inet/ipnodes

/var/yp/binding/*domain*/ypservers

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

ypbind(1M), sysinfo(2), hosts(4), ipnodes(4), attributes(5)

BUGS

ypinit sets up the list of NIS servers only for the current domain on the system when it is run, that is, the domain returned by the SI_SRPC_DOMAIN command to sysinfo(2). Care should be taken to ensure that this is the same as the desired domain for NIS client processes.

NAME

ypmake – rebuild NIS database

SYNOPSIS

```
cd /var/yp ; make [map]
```

DESCRIPTION

The file called `Makefile` in `/var/yp` is used by `make`(1) to build the Network Information Service (NIS) database. With no arguments, `make` creates `dbm` databases for any NIS maps that are out-of-date, and then executes `yppush`(1M) to notify slave databases that there has been a change.

If you supply a *map* on the command line, `make` will update that map only. Typing `make passwd` will create and `yppush` the password database (assuming it is out of date). Likewise, `make ipnodes` and `make networks` will create and `yppush` the ipnodes and network files, `$(INETDIR)/ipnodes` and `$(DIR)/networks`.

There are four special variables used by `make`: `DIR`, which gives the directory of the source files; `NOPUSH`, which when non-null inhibits doing a `yppush` of the new database files; `INETDIR`, which gives the directory of the ipnodes source file; and `DOM`, which is used to construct a domain other than the master's default domain. The default for `DIR` is `/etc`, and the default for `INETDIR` is `/etc/inet`. The default for `NOPUSH` is the null string.

Refer to `ypfiles`(4) and `ypserv`(1M) for an overview of the NIS service.

FILES

`/var/yp`	Directory containing NIS configuration files.
`/etc/inet/hosts`	System hosts file.
`/etc/inet`	Default directory for `ipnodes` source file.
`/etc`	Default directory for source files other than `ipnodes`.

SEE ALSO

`make`(1), `nis+`(1), `makedbm`(1M), `rpc.nisd`(1M), `ypbind`(1M), `yppush`(1M), `ypserv`(1M), `ypclnt`(3NSL), `ypfiles`(4), `ipnodes`(4)

NOTES

The NIS makefile is only used when running the `ypserv`(1M) server to provide NIS services. If these are being provided by the NIS+ server running in NIS compatibility mode, see `rpc.nisd`(1M); this makefile is not relevant. See `ypfiles`(4) for more details.

The Network Information Service (NIS) was formerly known as Sun Yellow Pages (YP). The functionality of the two remains the same; only the name has changed. The name Yellow Pages is a registered trademark in the United Kingdom of British Telecommunications plc, and may not be used without permission.

NAME

yppoll – return current version of a NIS map at a NIS server host

SYNOPSIS

/usr/sbin/yppoll [-d *ypdomain*] [-h *host*] *mapname*

DESCRIPTION

The `yppoll` command asks a `ypserv`( ) process what the order number is, and which host is the master NIS server for the named map.

OPTIONS

`-d` *ypdomain*	Use *ypdomain* instead of the default domain.
`-h` *host*	Ask the `ypserv` process at *host* about the map parameters. If *host* is not specified, the NIS server for the local host is used. That is, the default host is the one returned by `ypwhich`(1).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO

`ypwhich`(1), `ypfiles`(4), `attributes`(5)

NAME

yppush – force propagation of changed NIS map

SYNOPSIS

/usr/lib/netsvc/yp/yppush [-v] [-h *host*] [-d *domain*] [-p #*parallel-xfrs*] *mapname*

DESCRIPTION

yppush copies a new version of a Network Information Service (NIS) map from the master NIS server to the slave NIS servers. It is normally run only on the master NIS server by the Makefile in /var/yp after the master databases are changed. It first constructs a list of NIS server hosts by reading the NIS ypservers map within the *domain*. Keys within the ypservers map are the ASCII names of the machines on which the NIS servers run.

A "transfer map" request is sent to the NIS server at each host, along with the information needed by the transfer agent (the program that actually moves the map) to call back the yppush. When the attempt has completed (successfully or not), and the transfer agent has sent yppush a status message, the results can be printed to stdout. Messages are also printed when a transfer is not possible, for instance, when the request message is undeliverable, or when the timeout period on responses has expired.

Refer to ypfiles(4) and ypserv(1M) for an overview of the NIS service.

OPTIONS

The following options are supported:

-d *domain*	Specifies a *domain*.
-h *host*	Propagates only to the named *host*.
-p #*parallel-xfrs*	Allows the specified number of map transfers to occur in parallel.
-v	Verbose. This prints messages when each server is called, and for each response. If this flag is omitted, only error messages are printed.

FILES

/var/yp
Directory where NIS configuration files reside.

/var/yp/*domain*/ypservers. {*dir*, *pag* }
Map containing list of NIS servers to bind to when running in server mode.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWypu

SEE ALSO

ypserv(1M), ypxfr(1M), ypfiles(4), attributes(5)

NOTES The Network Information Service (NIS) was formerly known as Sun Yellow Pages (YP). The functionality of the two remains the same; only the name has changed. The name Yellow Pages is a registered trademark in the United Kingdom of British Telecommunications PLC, and must not be used without permission.

BUGS In the current implementation (version 2 NIS protocol), the transfer agent is `ypxfr`(1M), which is started by the `ypserv` program. If `yppush` detects that it is speaking to a version 1 NIS protocol server, it uses the older protocol, sending a version 1 `YPPROC_GET` request and issues a message to that effect. Unfortunately, there is no way of knowing if or when the map transfer is performed for version 1 servers. `yppush` prints a message saying that an "old-style" message has been sent. The system administrator should later check to see that the transfer has actually taken place.

NAME

ypserv, ypxfrd – NIS server and binder processes

SYNOPSIS

/usr/lib/netsvc/yp/ypserv [-dv]

/usr/lib/netsvc/yp/ypxfrd

DESCRIPTION

The Network Information Service (NIS) provides a simple network lookup service consisting of databases and processes. The databases are `ndbm` files in a directory tree rooted at `/var/yp`. See `ndbm`(3C). These files are described in `ypfiles`(4). The processes are `/usr/lib/netsvc/yp/ypserv`, the NIS database lookup server, and `/usr/lib/netsvc/yp/ypbind`, the NIS binder. The programmatic interface to the NIS service is described in `ypclnt`(3NSL). Administrative tools are described in `yppoll`(1M), `yppush`(1M), `ypset`(1M), `ypxfr`(1M), and `ypwhich`(1). Tools to see the contents of NIS maps are described in `ypcat`(1), and `ypmatch`(1). Database generation and maintenance tools are described in `ypinit`(1M), `ypmake`(1M), and `makedbm`(1M).

The `ypserv` utility is a daemon process typically activated at system startup from `/etc/init.d/rpc`. Alternatively, you can, as the root user, startNIS services using `ypstart`(1M) from the command-liner. `ypserv` runs only on NIS server machines with a complete NIS database. You can halt all NIS services using the `ypstop`(1M) command.

The `ypxfrd` utility transfers entire NIS maps in an efficient manner. For systems that use this daemon, map transfers are 10 to 100 times faster, depending on the map. To use this daemon, be sure `ypxfrd` is running on the master server. See `/usr/lib/netsvc/yp/ypstart`. `ypxfr` attempts to use `ypxfrd` first. If that fails, it prints a warning, then uses the older transfer method.

The `ypserv` daemon's primary function is to look up information in its local database of NIS maps.

The operations performed by `ypserv` are defined for the implementor by the *YP Protocol Specification*, and for the programmer by the header file `rpcsvc/yp_prot.h`.

Communication to and from `ypserv` is by means of RPC calls. Lookup functions are described in `ypclnt`(3NSL), and are supplied as C-callable functions in the `libnsl`(3LIB) library. There are four lookup functions, all of which are performed on a specified map within some NIS domain: `yp_match`(3NSL), `yp_first`(3NSL), `yp_next`(3NSL), and `yp_all`(3NSL). The `yp_match` operation takes a key, and returns the associated value. The `yp_first` operation returns the first key-value pair from the map, and `yp_next` can be used to enumerate the remainder. `yp_all` ships the entire map to the requester as the response to a single RPC request.

A number of special keys in the DBM files can alter the way in which `ypserv` operates. The keys of interest are:

`YP_INTERDOMAIN` — The presence of this key causes `ypserv` to forward to a DNS server host lookups that cannot be satisfied by the DBM files.

`YP_SECURE`	This key causes `ypserv` to answer only questions coming from clients on reserved ports.
`YP_MULTI_hostname`	This is a special key in the form, `YP_MULTI_`*hostname addr1,...,addrN*. A client looking for `hostname` receives the "closest" address.

Two other functions supply information about the map, rather than map entries: `yp_order`(3NSL), and `yp_master`(3NSL). In fact, both order number and master name exist in the map as key-value pairs, but the server will not return either through the normal lookup functions. If you examine the map with `makedbm`(1M), however, they are visible. Other functions are used within the NIS service subsystem itself, and are not of general interest to NIS clients. They include `do_you_serve_this_domain?`, `transfer_map`, and `reinitialize_internal_state`.

ypserv

`-d`	The NIS service should go to the DNS for more host information. This requires the existence of a correct `/etc/resolv.conf` file pointing at a machine running `in.named`(1M). This option turns on DNS forwarding regardless of whether or not the `YP_INTERDOMAIN` flag is set in the `hosts` maps. See `makedbm`(1M). In the absence of an `/etc/resolv.conf` file, `ypserv` complains, but ignores the `-d` option.
`-v`	Operate in the verbose mode, printing diagnostic messages to stderr.

FILES

`/var/yp/securenets`	Defines the hosts and networks that are granted access to information in the served domain; it is read at startup time by both `ypserv` and `ypxfrd`.
`/etc/init.d/rpc`	Startup file that starts up basic RPC services and NIS by calling `ypstart`(1M). If the `/var/yp/ypserv.log` file exists when `ypserv` starts up, log information is written to it when error conditions arise. The file `/var/yp/binding/domainname/ypservers` is used to list the NIS server hosts that `ypbind` can bind to.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWypu

SEE ALSO

`ypcat`(1), `ypmatch`(1), `ypwhich`(1), `domainname`(1M), `in.named`(1M), `makedbm`(1M), `ypbind`(1M), `ypinit`(1M), `ypmake`(1M), `yppoll`(1M), `yppush`(1M), `ypset`(1M), `ypstart`(1M), `ypstop`(1M), `ypxfr`(1M), `ndbm`(3C), `ypclnt`(3NSL), `libnsl`(3LIB), `securenets`(4), `ypfiles`(4), `attributes`(5)

Network Interfaces Programmer's Guide

System Administration Guide: Basic Administration

NOTES

`ypserv` supports multiple domains. The `ypserv` process determines the domains it serves by looking for directories of the same name in the directory `/var/yp`. It replies to all broadcasts requesting yp service for that domain.

The Network Information Service (NIS) was formerly known as Sun Yellow Pages (YP). The functionality of the two remains the same; only the name has changed. The name Yellow Pages is a registered trademark in the United Kingdom of British Telecommunications PLC, and must not be used without permission.

NAME

ypset – point ypbind at a particular server

SYNOPSIS

/usr/sbin/ypset [-d *ypdomain*] [-h *host*] *server*

DESCRIPTION

In order to run ypset, ypbind must be initiated with the –ypset or –ypsetme options. See ypbind(1M). ypset tells ypbind to get NIS services for the specified *ypdomain* from the ypserv process running on *server*. If *server* is down, or is not running ypserv, this might not be discovered until an NIS client process tries to obtain a binding for the domain. At this point, the binding set by ypset is tested by ypbind. If the binding is invalid, ypbind attempts to rebind for the same domain.

ypset is useful for binding a client node that is not on a broadcast net, or is on a broadcast net that is not running an NIS server host. It is also useful for debugging NIS client applications, for instance, where an NIS map exists only at a single NIS server host.

Where several hosts on the local net are supplying NIS services, ypbind can rebind to another host, even while you attempt to find out if the ypset operation succeeded. For example, if you enter the ypset command below, you might get the subsequent response from ypwhich:

```
example% ypset host1
example% ypwhich
host2
```

The sequence shown above is a function of the NIS subsystem's attempt to load-balance among the available NIS servers, and occurs when host1 does not respond to ypbind because it is not running ypserv (or is overloaded), and host2, running ypserv, obtains the binding.

server indicates which NIS server to bind to, and must be specified as a name or an IP address. This works only if the node has a current valid binding for the domain in question and ypbind has been set to allow use of ypset. In most cases, *server* should be specified as an IP address.

ypset tries to bind over a connectionless transport. The NIS library call, yp_all(), uses connection-oriented transport and derives the NIS server's address based on the connectionless address supplied by ypset.

Refer to ypfiles(4) for an overview of the NIS name service.

OPTIONS

-d *ypdomain*	Use *ypdomain*, instead of the default domain.
-h *host*	Set ypbind's binding on *host*, instead of locally. Specify *host* as a name.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

SEE ALSO `ypwhich`(1), `ypfiles`(4), `attributes`(5)

NAME

ypstart, ypstop – Start and stop NIS services

SYNOPSIS

```
/usr/lib/netsvc/yp/ypstart

/usr/lib/netsvc/yp/ypstop
```

DESCRIPTION

The `ypstart` command is used to start the Network Information Service (NIS). After the host has been configured using the `ypinit`(1M) command, `ypstart` automatically determines the NIS status of the machine and starts the appropriate daemons.

The `ypstop` command is used to stop the Network Information Service (NIS).

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWypu

SEE ALSO

`ypinit`(1M), `attributes`(5)

System Administration Guide: Basic Administration

NOTES

The Network Information Service (NIS) was formerly known as Sun Yellow Pages (YP). The functionality of the two services remains the same; only the name has changed. The name Yellow Pages is a registered trademark in the United Kingdom of British Telecommunications PLC, and must not be used without permission.

NAME

ypxfr, ypxfr_1perday, ypxfr_2perday, ypxfr_1perhour – transfer NIS map from a NIS server to host

SYNOPSIS

/usr/lib/netsvc/yp/ypxfr [-c] [-f] [-C *tid prog server*] [-d *ypdomain*] [-h *host*] [-s *ypdomain*] *mapname*

DESCRIPTION

The ypxfr command moves an NIS map in the default domain for the local host to the local host by making use of normal NIS services. It creates a temporary map in the directory /var/yp/*ypdomain* (this directory must already exist; *ypdomain* is the default domain for the local host), fills it by enumerating the map's entries, fetches the map parameters (master and order number), and loads them. It then deletes any old versions of the map and moves the temporary map to the real *name*.

If run interactively, ypxfr writes its output to the terminal. However, if it is started without a controlling terminal, and if the log file /var/yp/ypxfr.log exists, it appends all its output to that file. Since ypxfr is most often run from the privileged user's crontab file, or by ypserv, the log file can retain a record of what was attempted, and what the results were.

For consistency between servers, ypxfr should be run periodically for every map in the NIS data base. Different maps change at different rates: a map might not change for months at a time, for instance, and can therefore be checked only once a day. Some maps might change several times per day. In such a case, you might want to check hourly for updates. A crontab(1) entry can be used to automatically perform periodic updates. Rather than having a separate crontab entry for each map, you can group commands to update several maps in a shell script. Examples (mnemonically named) are in /usr/sbin/yp: ypxfr_1perday, ypxfr_2perday, and ypxfr_1perhour.

Refer to ypfiles(4) for an overview of the NIS name service.

OPTIONS

-c	Do not send a "Clear current map" request to the local ypserv process. Use this flag if ypserv is not running locally at the time you are running ypxfr. Otherwise, ypxfr complains that it cannot communicate with the local ypserv, and the transfer fails.
-f	Force the transfer to occur even if the version at the master is not more recent than the local version.
-C *tid prog server*	This option is for use *only* by ypserv. When ypserv starts ypxfr, it specifies that ypxfr should call back a yppush process at the host *server*, registered as program number *prog*, and waiting for a response to transaction *tid*.
-d *ypdomain*	Specify a domain other than the default domain.
-h *host*	Get the map from *host*, regardless of the master. If *host* is not specified, ypxfr asks the NIS service for the name of the master, and tries to get the map from there. *host* must be a valid host name.

`-s` *ypdomain*
: Specify a source domain from which to transfer a map that should be the same across domains.

FILES

`/var/yp/ypxfr.log`
: Log file

`/usr/lib/netsvc/yp/ypxfr_1perday`
: Script to run one transfer per day, for use with `cron`(1M)

`/usr/lib/netsvc/yp/ypxfr_2perday`
: Script to run two transfer per day, for use with `cron`(1M)

`/usr/lib/netsvc/yp/ypxfr_1perhour`
: Script for hourly transfers of volatile maps

`/var/yp/`*ypdomain*
: NIS domain

`/usr/spool/cron/crontabs/root`
: Privileged user's `crontab` file

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ypxfr Only

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnisu

ypxfr_1perday, ypxfr_1perhour, and ypxfr_2perday

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWypu

SEE ALSO

`crontab`(1), `cron`(1M), `ypinit`(1M)`yppush`(1M), `ypserv`(1M), `ypfiles`(4), `attributes`(5)

NAME

zdump – time zone dumper

SYNOPSIS

zdump [-v] [-c *cutoffyear*] [*zonename*...]

DESCRIPTION

The zdump command prints the current time for each time zone (*zonename*) listed on the command line. Specify *zonename* as the name of the time zone database file relative to /usr/share/lib/zoneinfo.

Specifying an invalid time zone (*zonename*) to zdump does not return an error, rather zdump uses GMT. This is consistent with the behavior of the library calls; zdump reflects the same behavior of the time routines in libc. See ctime(3C) and mktime(3C).

OPTIONS

The following options are supported:

-v Displays the entire contents of the time zone database file for *zonename*. Prints the time at the lowest possible time value; the time one day after the lowest possible time value; the times both one second before and exactly at each time at which the rules for computing local time change; the time at the highest possible time value; and the time at one day less than the highest possible time value. See mktime(3C) and ctime(3C) for information regarding time value (time_t). Each line of output ends with isdst=1 if the given time is Daylight Saving Time, or isdst=0 otherwise.

-c *cutoffyear* Cuts off the verbose output near the start of the year *cutoffyear*.

EXIT STATUS

The following exit values are returned:

0 Successful completion.

1 An error occurred.

FILES

/usr/share/lib/zoneinfo Standard zone information directory

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO

zic(1M), ctime(3C), mktime(3C), attributes(5), environ(5)

NAME

zic – time zone compiler

SYNOPSIS

zic [-s] [-v] [-l *localtime*] [-p *posixrules*] [-d *directory*] [-y *yearistype*] [*filename*...]

DESCRIPTION

zic reads text from the file(s) named on the command line and creates the time conversion information files specified in this input. If a *filename* is '–', the standard input is read.

Input lines are made up of fields. Fields are separated by any number of white space characters. Leading and trailing white space on input lines is ignored. A pound sign (#) indicates a comment that extends to the end of the line. White space characters and pound signs can be enclosed within double quotes (" ") if they are to be used as part of a field. Any line that is blank (after comment stripping) is ignored. Non-blank lines are expected to be of one of three types: rule lines, zone lines, or link lines.

Rule

A rule line has the form:

For example:

```
Rule   NAME  FROM  TO  TYPE  IN   ON     AT   SAVE  LETTER/S
```

The fields that make up a rule line are:

```
Rule   USA   1969  1973   -  Apr lastSun 2:00  1:00   D
```

NAME — Gives the (arbitrary) name of the set of rules this rule is part of.

FROM — Gives the first year in which the rule applies. The word minimum (or an abbreviation) means the minimum year with a representable time value. The word maximum (or an abbreviation) means the maximum year with a representable time value.

TO — Gives the final year in which the rule applies. In addition to minimum and maximum (as above), the word only (or an abbreviation) can be used to repeat the value of the FROM field.

TYPE — Gives the type of year in which the rule applies. If TYPE is:

'–'	The rule applies in all years between FROM and TO, inclusive.
uspres	The rule applies in U.S. Presidential election years.
nonpres	The rule applies in years other than U.S. Presidential election years.
even	The rule applies to even-numbered years.
odd	The rule applies to odd-numbered years.

If `TYPE` is something else, then `zic` will attempt to execute the command

```
yearistype year type
```

to check the type of a year: an exit status of `0` means that the year is of the given type; an exit status of `1` means that the year is not of the given type. The `yearistype` command is not currently provided in the Solaris environment.

`IN` Names the month in which the rule takes effect. Month names can be abbreviated.

`ON` Gives the day on which the rule takes effect. Recognized forms include:

`5`	the fifth day of the month
`lastSun`	The last Sunday in the month
`lastMon`	The last Monday in the month
`Sun>=8`	First Sunday on or after the eighth
`Sun<=25`	Last Sunday on or before the 25th

Names of days of the week can be abbreviated or spelled out in full. Note: There cannot be spaces within the `ON` field.

`AT` Gives the time of day at which the rule takes effect. Recognized forms include:

`2`	Time in hours
`2:00`	Time in hours and minutes
`15:00`	24-hour format time (for times after noon)
`1:28:14`	Time in hours, minutes, and seconds, where hour 0 is midnight at the start of the day and hour 24 is midnight at the end of the day.

Any of these forms can be followed by the letter `w` if the given time is local "wall clock" time; `s` if the given time is local "standard" time; or `u` (or `g` or `z`) if the given time is universal time. In the absence of an indicator, wall clock time is assumed.

`SAVE` Gives the amount of time to be added to local standard time when the rule is in effect. This field has the same format as the `AT` field (without the `w` and `s` suffixes).

`LETTER/S` Gives the "variable part" (for example, the "S" or "D" in "EST" or "EDT" of time zone abbreviations to be used when this rule is in effect. If this field is '–', the variable part is null.

Zone A zone line has the form:

```
Zone  NAME                GMTOFF  RULES/SAVE   FORMAT  [UNTIL]
```

For example:

```
Zone Australia/SouthWest  9:30       -         CST    1992 Mar 15 12:00
                          8:30     Aus         CST
```

The fields that make up a zone line are:

`NAME` The name of the time zone. This is the name used in creating the time conversion information file for the zone.

`GMTOFF` The amount of time to add to UTC to get standard time in this zone. This field has the same format as the `AT` and `SAVE` fields of rule lines; begin the field with a minus sign to subtract time from UTC.

`RULES/SAVE` The name of the rule(s) that apply in the time zone or, alternately, an amount of time to add to local standard time. If this field is '–', then standard time always applies in the time zone.

`FORMAT` The format for time zone abbreviations in this time zone. The pair of characters `%s` is used to show where the "variable part" of the time zone abbreviation goes. Alternately, a slash (/) separates standard and daylight abbreviations.

`UNTIL` The time at which the UTC offset or the rule(s) change for a location. It is specified as a year, a month, a day, and a time of day. The time of day has the same format as the `AT` field of rule lines. If this is specified, the time zone information is generated from the given UTC offset and rule change until the time specified.

The month, day, and time of day have the same format as the IN, ON, and AT columns of a rule; trailing columns can be omitted, and default to the earliest possible value for the missing columns.

The next line must be a "continuation" line. This line has the same form as a zone line except that the string "Zone" and the name are omitted. The continuation line places information starting at the time specified as the `UNTIL` field in the previous line in the file used by the previous line. Continuation lines can contain an `UNTIL` field, just as zone lines do, indicating that the next line is a further continuation.

Link A link line has the form:

```
Link   LINK-FROM   LINK-TO
```

For example:

```
Link   Europe/Istanbul Asia/Istanbul
```

The LINK-FROM field should appear as the NAME field in some zone line; the LINK-TO field is used as an alternate name for that zone.

Except for continuation lines, lines can appear in any order in the input.

OPTIONS

`-d` *directory* — Creates time conversion information files in the directory *directory* rather than in the standard directory `/usr/share/lib/zoneinfo`.

`-l` *localtime* — Uses the given time zone as local time *localtime*. `zic` acts as if the file contained a link line of the form:

```
Link localtime  localtime
```

`-p` *posixrules* — Uses the rules of the given time zone *posixrules* when handling POSIX-format time zone environment variables. `zic` acts as if the input contained a link line of the form:

```
Link     posixrules posixrules
```

This option is not used by `ctime`(3C) and `mktime`(3C) in the Solaris environment.

`-s` — Limits time values stored in output files to values that are the same whether they are taken to be signed or unsigned. You can use this option to generate SVVS-compatible files.

`-v` — Complains if a year that appears in a data file is outside the range of years representable by system time values (`0:00:00 a.m. UTC, January 1, 1970`, to `3:14:07 a.m. UTC, January 19, 2038`).

`-y` *yearistype* — Uses the given command *yearistype* rather than `yearistype` when checking year types (see `Rules` under `DESCRIPTION`).

OPERANDS

filename — A file containing input lines that specify the time conversion information files to be created. If a *filename* is '–', the standard input is read.

FILES

`/usr/share/lib/zoneinfo`	Standard directory used for created files
`/usr/share/lib/zoneinfo/src`	Directory containing source files

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu

SEE ALSO `time`(1), `zdump`(1M), `ctime`(3C), `mktime`(3C), `attributes`(5)

NOTES For areas with more than two types of local time, you might need to use local standard time in the `AT` field of the earliest transition time's rule to ensure that the earliest transition time recorded in the compiled file is correct.

Index

Numbers and Symbols

— ikecert, 522
— logadm, 731
— mount_xmemfs, 1007
— projadd, 1261
rctladm, 1301
sftp-server, 1429
sshd, 1589
construct a udfs file system — mkfs_udfs, 951
24-bit TrueColor Visual for OpenWindows with an S24 frame buffer system (TCX) — tcxconfig
    configure the default linearity, 1667

A

accept — accept print requests, 28
access system administration tools with a graphical user interface — solstice, 1579
accounting
    run daily — runacct, 1369
accounting commands
    — acctcms, 36
    — acctcon, 38
    — acctdisk, 30
    — acctdusg, 30
    — acctmerg, 40
    — accton, 30
    — acctprc, 41
    — acctwtmp, 30
    — closewtmp, 30
    — fwtmp, 462
accounting commands *(continued)*
    — utmp2wtmp, 30
    — wtmpfix, 462
accounting shell procedures
    — chargefee, 43
    — ckpacct, 43
    — dodisk, 43
    — lastlogin, 43
    — monacct, 43
    — nulladm, 43
    — prctmp, 43
    — prdaily, 43
    — prtacct, 43
    — runacct, 43
    — shutacct, 43
    — startup, 43
    — turnacct, 43
acct — miscellaneous accounting commands, 30
acctadm — configure extended accounting facility, 33
acctcms — command summary from process accounting records, 36
acctcon — create summary of connect records, 38
acctcon1 — create summary of connect records, 38
acctcon2 — create summary of connect records, 38
acctdisk — create disk usage records, 30
acctdusg — compute disk usage by login, 30
acctmerg — merge or add total disk accounting files, 40

accton — turn on process accounting, 30
acctprc — summary of process accounting, 41
acctprc1 — summary of process accounting, 41
acctprc2 — summary of process accounting, 41
acctsh — shell procedures for accounting, 43
acctwtmp — create connect accounting record in /var/adm/wtmp, 30
adb scripts
generate — adbgen, 46
adbgen — generate adb script, 46
add (create) a new group definition on the system — groupadd, 476
add a new device driver to the system — add_drv, 51
add_drv — add a new device driver to the system, 51
addbadsec — map out defective disk blocks, 49
address resolution display and control — arp, 70
adds an application to the Solstice application registry — soladdapp, 1577
administer a new role account on the system — roleadd, 1318
administer disk space used for caching file systems with the Cache File-System (CacheFS) — cfsadmin, 204
administer files available as block devices through lofi — lofiadm, 726
administer flash archives — flar, 377
administer FTP Server enhanced group access file — privatepw, 1258
administer NIS+ aliases — aliasadm, 65
administration program for the Sun Enterprise Network Array (SENA), RSM, SPARCstorage Array (SSA) subsystems, Sun Fire 880 internal storage subsystem, and individual Fiber Channel Arbitrated Loop (FC_AL) devices — luxadm, 823
admintool — system administration tool with a graphical interface, 55
afbconfig — configure the AFB Graphics Accelerator, 57
aliasadm — administer NIS+ aliases, 65
ancillary agent for inbound zone transfers — named-xfer, 1014
answerbook2_admin — bring up AnswerBook2 administration tool GUI, 67
AnswerBook2 administration tool GUI — answerbook2_admin, 67
apache — Apache hypertext transfer protocol server overview, 68
Apache hypertext transfer protocol server overview — apache, 68
apply a patch package to a Solaris 2 system — patchadd, 1146
arp — address resolution display and control, 70
ASET
monitors or restricts accesses to system files and directories — aset, 72
prints tasks status — taskstat, 1666
aset.restore — restore files and directories to the state before ASET was installed, 78
audit — maintain audit trail, 79
audit records
select or merge from audit trail files — auditreduce, 88
audit_startup shell script, 96
audit statistics report — auditstat, 97
audit trail file
select records from — auditreduce, 88
audit_warn — audit daemon warning script, 99
auditconfig — get and set kernel audit parameters, 80
auditd — audit daemon, 86
auditreduce — select or merge audit records from audit trail files, 88
auditstat — display kernel audit statistics, 97
autoboot procedures — boot, 593
autofs
automatically mount file systems — automount, 101
Autofs
mount/unmount request server — automountd, 108
Automated conversational exchange tool — chat, 210
Automated Security Enhancement Tool, *See* ASET
automount — automatically mount file systems, 101

automountd — Autofs mount/unmount request server, 108
autopush — configures lists of automatically pushed STREAMS modules, 109

B

backup NIS+ directories — nisbackup, 1062
Basic Security Module
  enable — bsmconv, bsmunconv, 127
Basic Security Module commands
  — audit, 79
  — audit_startup, 96
  — audit_warn, 99
  — auditconfig, 80
  — auditd, 86
  — auditreduce, 88
  — auditstat, 97
bdconfig — configures the bd (buttons and dials) stream, 111
snmpXwbemd — SNMP Adapter Subagent for WBEM, 1562
boot — bootstrap procedures, 113
boot parameter server — rpc.bootparamd, 1334
boot — system startup procedures, 593
BOOTP
  DHCP network table management utility — pntadm, 1199
  administration — dhtadm, 285
  server — in.dhcpd, 528
  service — dhcpconfig, 276
bootstrap procedures — boot, 593
bootstrap PROM monitor program — monitor, 966
Bootstrap Protocol, *See* 'DHCP'
Bourne shell
  — rsh, 1366
broadcast message
  network wall server — rpc.rwalld, 1354
  write to all users — wall, 1753
  write to all users of a group— wall, 1753
  write to all users over a network — rwall, 1372
bsmconv — eable Solaris system to use Basic Security Module, 127
bsmrecord — verify Basic Security Module (BSM) configuration files, 129
bsmunconv — eable Solaris system to use Basic Security Module, 127
build class list and compile Solaris Management Console service beans for remote use — smccompile, 1464
busstat — report bus-related performance statistics, 132

C

cachefs
  cachefslog — Cache File System logging, 137
  cachefsstat — Cache File System statistics, 141
  cachefswssize — Determine working set size for cachefs, 143
  cfsadmin — administer disk space used for caching file systems with the Cache File-System (CacheFS), 204
  fsck_cachefs — check integrity of data cached with CacheFS, 416
  mount_cachefs — mount CacheFS file systems, 983
CacheFS daemon — cachefsd, 136
cachefsd — CacheFS daemon, 136
cachefspack — pack files and file systems in the cache, 139
captoinfo — convert termcap description to terminfo description, 145
catman — create formatted files for manual pages, 146
CD-ROM
  packages on — pkgadd, 1181
CD-ROM, managing
  — rmmount, 1313
CD-ROM, mounting
  — rmmount, 1313
cfgadm — configuration administration, 150
cfgadm_ac — EXX00 memory system administration, 161
cfgadm_scsi — SCSI hardware specific commands for cfgadm, 184
cfgadm_sysctrl — EXX00 system board administration, 190

cfs, *See* cachefs
cfsadmin — administer disk space used for caching file systems with the Cache File-System (CacheFS), 204
CacheFS Resource Parameters, 205
cg14config — configure the cgfourteen device, 208
change processor operational status — psradm, 1278
chargefee — accounting shell procedure, 43
chat — Automated conversational exchange tool, 210
check-hostname — check if sendmail can determine the system's fully-qualified host name, 218
check if sendmail can determine the system's fully-qualified host name — check-hostname, 218
check integrity of data cached with CacheFS — fsck_cachefs, 416
check-permissions — check permissions on mail rerouting files, 219
check permissions on mail rerouting files — check-permissions, 219
chown — change owner of file, 220
chroot — change root directory for a command, 221
CIM Boot Manager
starting, 555
CIM Object Manager
stopping, 555
ckpacct — accounting shell procedure, 43
clear inode
— clri, 226
— dcopy, 226
clear_locks — clear locks held on behalf of an NFS client, 224
clear locks held on behalf of an NFS client — clear_locks, 224
client configuration daemon — dhcpagent, 272
clinfo — display cluster information, 225
close down the FTP Servers at a given time — ftpshut, 458
closewtmp — puts a false DEAD_PROCESS record in /var/adm/wtmpx file, 30
clri — clear inode, 226
compile MOF files into CIM classes — mofcomp, 960
configuration applications
execute or define — sysidconfig, 1651
configure the AFB Graphics Accelerator — afbconfig, 57
configure the FFB Graphics Accelerator — ffbconfig, 368
configure the M64 Graphics Accelerator — m64config, 838
configure the PGX32 Graphics Accelerator — pgxconfig, 1169
configure the Raptor GFX Graphics Accelerator — GFXconfig, 1169
configure
cgfourteen device — cg14config, 208
contiguous memory for sx video — sxconfig, 1630
undo system configuration — sys-unconfig, 1660
configure and administer a smartcard — smartcard, 1447
configure extended accounting facility — acctadm, 33
configure Mobility IP Agent — mipagentconfig, 933
configure operating system crash dump — dumpadm, 339
configure or unconfigure mouse, keyboard, display — kdmconfig, 667
configure system wide IPsec policy — ipsecconf, 620
Configure the power management system — pmconfig, 1197
configure the Solaris Management Console — smcconf, 1468
configure the XVR-1000 Graphics Accelerator — SUNWgfb_config, 1610
configure transport providers for use by sockets — soconfig, 1575
configures the bd (buttons and dials) stream — bdconfig, 111
connect accounting
— acctcon, 38
— fwtmp, 462
— utmp2wtmp, 30
— wtmpfix, 462
consadm — select or display devices used as auxiliary console devices, 227
construct a FAT file system — mkfs_pcfs, 947

control and query bindings of processes to processors — pbind, 1162
conv_lp — convert LP configuration, 229
conv_lpd — convert LPD configuration, 230
convert a JavaSpaces datastore to the newer Reliable Log datastore format — wbemconfig, 1758
convert LP configuration — conv_lp, 229
convert LPD configuration — conv_lpd, 230
convert name server configuration files — named-bootconf, 1013
copy FNS contexts — fncopy, 387
coradm — core file administration, 232
core file administration — coradm, 232
CPU PROM monitor
program — monitor, 966
cpustat — monitor system behavior using CPU performance counters, 236
create
NIS dbm file — makedbm, 845
create a flash archive from a master system — flarcreate, 380
create a temporary snapshot of a UFS file system — fssnap_ufs, 448
create an FNS context — fncreate, 389
create FNS file system contexts — fncreate_fs, 396
create LDAP entries from corresponding /etc files — ldapaddent, 691
create new printers in the FNS namespace — fncreate_printer, 401
create temporary snapshots of a file system — fssnap, 446
creation and management of processor sets — psrset, 1283
cron — clock daemon, 239
cvcd — virtual console daemon, 241

D

daemon, distributed system administration — sadmind, 1380
daemon for DHCP client configuration — dhcpagent, 272
daemon for the Internet Key Exchange (IKE) — in.iked, 548
daemon to monitor metadevices — mdmonitord, 851
daemons
clock daemon — cron, 239
Internet Trivial File Transfer Protocol — in.tftpd, 611
kernel statistics — rpc.rstatd, 1352
network router discovery daemon — in.rdisc, 572
network username server — rpc.rusersd, 1353
network wall server — rpc.rwall, 1354
network status monitor — statd, 1597
NFS — nfsd, 1039
NIS+ service — rpc.nisd, 1342
remote quota — rquotad, 1365
RPC-based remote execution — rpc.rexd, 1350
server which records packets sent by spray — rpc.sprayd, 1356
date
set system date from a remote host — rdate, 1303
dcopy — clear inode, 226
dcs — domain configuration server, 242
dd — convert and copy a file, 243
debug tools
generate adb script — adbgen, 46
delete a project from the system — projdel, 1263
delete a user's login from the system — roledel, 1322
destroy an FNS context — fndestroy, 404
devattr — displays the values for a device's attributes, 249
devfsadm — administration command for /dev and /devices, 251
devfsadmd — administration daemon for /dev and /devices, 251
device access
verifies accessibility — getvol, 472
device_maps
display entries — dminfo, 330
devices
add, modify remove device entry from table — putdev, 1287
add, remove and change device group table — putdgrp, 1290

devices *(continued)*
adds /dev entries for miscellaneous devices and pseudo-devices — devlinks, 254
adds /dev entries for tape drives attached to the system — tapes, 1662
cgfourteen — configure the cgfourteen device, 208
display access control entries from device_maps, 330
display attributes — devattr, 249
group list — getdgrp, 465
list — getdev, 463
lists members of a group — listdgrp, 707
name — devnm, 258
print information about disk devices — devinfo, 253
print VTOC of a block device — prtvtoc, 1276
release from exclusive use — devfree, 250
remove a device driver from the system — rm_drv, 1306
reserve for exclusive use — devfree, 259
/devices directory
configure — drvconfig, 336
devinfo — print device specific information, 253
df — display amount of used and available disk space, 261
df_ufs — dispay amount of space occupied by ufs, 271
dfmounts — displays information on resources shared through DFS, 265
dfmounts_nfs — displays information on resources shared through NFS, 267
DFS
display information on resources shared — dfmounts, 265
list available resources from remote or local systems — dfshares, 268
dfshares — list available resources from remote or local systems, 268
dfshares_nfs — list available resources from remote systems, 269
DHCP
network table management utility — pntadm, 1199
administration — dhtadm, 285
server — in.dhcpd, 528
DHCP *(continued)*
service — dhcpconfig, 276
DHCP client configuration daemon — dhcpagent, 272
DHCP service lock daemon — dsvclockd, 338
dhcpagent — daemon for DHCP client configuration, 272
dhcpconfig — DHCP service configuration utility, 276
dhcpmgr — graphical interface for managing DHCP service, 283
dhtadm — DHCP configuration table management utility, 285
Diffie-Hellman key pair, 1037
directories
move — mvdir, 1012
directoryserver — front end for the Directory Server (DS), 298
disk accounting
— acctdisk, 30
— acctdusg, 30
— acctmerg, 40
disk blocks
display free ones — df, 261
disk quotas
remote quota daemon — rquotad, 1365
disk quotas and usage
display for a user's ufs file system — quota, 1297
diskettes
packages on — pkgadd, 1181
disks — adds /dev entries for hard disks attached to the system, 317
disks
partitioning and maintenance utility — format, 408
diskscan — perform surface analysis, 321
display
system configuration information — prtconf, 1271
system diagnostic information — prtdiag, 1273
display cluster information — clinfo, 225
SNMP Adapter Subagent for WBEM — snmpXwbemd, 1562
display kernel statistics — kstat, 680
display or set boot environment description — ludesc, 801

displays information about processors — psrinfo, 1281
Distributed File System, *See* DFS
dmesg — create error log from system diagnostic messages, 325
dmi_cmd — DMI command line interface utility, 326
DMI command line interface utility — dmi_cmd, 326
DMI command line retrieval utility — dmiget, 329
dmiget — DMI command line retrieval utility, 329
dminfo — display device_maps entries, 330
dmispd — Sun Solstice Enterprise DMI Service Provider, 332
DoD Internet
host table — gettable, 469
DoD Internet format host table, convert — htable, 490
dodisk — accounting shell procedure, 43
domain configuration server — dcs, 242
domainname — display name of current domain, 335
domainname — set name of current domain, 335
driver statistics from a synchronous serial link, report — syncstat, 1640
drvconfig — configure /devices, 336
dsvclockd — DHCP service lock daemon, 338
dumpadm — configure operating system crash dump, 339
Dynamic Host Configuration Protocol, *See* 'DHCP'

E

edquota — edit user quotas for ufs file system, 346
EEPROM display and load program — eeprom, 348
efdaemon — embedded FCode interpreter daemon, 355
embedded FCode interpreter daemon — efdaemon, 355
encryption keys
server for storing private keys — keyserv, 673
environment variables
convert termcap description into terminfo description — captoinfo, 145
error log
create from system diagnostic messages — dmesg, 325
etrn – send ETRN commands to start mail queue, 356
EXX00 memory system administration — cfgadm_ac, 161
EXX00 system board administration — cfgadm_sysctrl, 190

F

fbconfig — Frame Buffer configuration utility, 358
fdetach — detach a name from STREAMS-based file descriptor, 360
fdisk — create or modify fixed disk partition table
Menu Options, 361
ffbconfig — configure the FFB Graphics Accelerator, 368
FIFO
make FIFO special file — mkfifo, 943
FIFO, make — mknod, 956
File Transfer Protocol Server — ftpd, 541
File Transfer Protocol Server — in.ftpd, 541
file system
cache file systems
See cachefs
change the dynamic parameters — tunefs, 1703
check and repair — fsck, 412
check and repair ufs — fsck_ufs, 422
construct — mkfs, 945
construct ufs — mkfs_ufs, 953
create new — newfs, 1033
debugger — fsdb, 426
dispay amount of used disk space occupied by ufs — df_ufs, 271
display amount of used and available disk space — df, 261

file system *(continued)*
grow — growfs, 481
link and unlink files and directories — link, 706
list file names and statistics — ff, 366
list file names and statistics for ufs — ff_ufs, 376
loopback — mount, 978
make literal copy — volcopy, 1748
mount — mount, 978
mount — mountall, 981
mount ufs — mount_ufs, 1003
move directory — mvdir, 1012
provide labels — labelit, 685
provide labels for ufs — labelit_ufs, 690
report processes using file or file structure — fuser, 460
share multiple resources — shareall, 1432
ufs, make image copy — volcopy_ufs, 1750
unmount — umount, 978
unmount — umountall, 981
unshare multiple resources — unshareall, 1432
file system consistency check and interactive — fsck_pcfs, 417
file system type, determine — fstyp, 453
files
change ownership — chown, 220
convert and copy — dd, 243
flar — administer flash archives, 377
flarcreate — create a flash archive from a master system, 380
floppy, managing
— rmmount, 1313
floppy, mounting
— rmmount, 1313
flush disk activity — sync, 1633
fmthard — populate Volume Table of Contents on hard disks, 383
fncopy — copy FNS contexts, 387
fncreate — create an FNS context, 389
fncreate_fs — create FNS file system contexts, 396
fncreate_printer — create new printers in the FNS namespace, 401
fndestroy — destroy an FNS context, 404
FNS
copy FNS contexts — fncopy, 387
FNS *(continued)*
create context — fncreate, 389
create FNS file system contexts — fncreate_fs, 396
select naming service for FNS Initial Context — fnselect, 405
fnselect — select naming service for FNS Initial Context, 405
format — disk partitioning and maintenance utility, 408
Frame Buffer configuration utility — fbconfig, 358
front end for the Directory Server (DS) — directoryserver, 298
fsck_cachefs — check integrity of data cached with CacheFS, 416
fsck — check and repair file systems, 412
fsck_pcfs — file system consistency check and interactive, 417
fsck_ufs — check and repair ufs, 422
fsdb — file system debugger, 426
fsdb_ufs — ufs file system debugger, 435
Commands, 438
Expressions, 436
Formatted Output, 441
Inode Commands, 440
fsirand — install random inode generation numbers, 445
fssnap — create temporary snapshots of a file system, 446
fssnap_ufs — create a temporary snapshot of a UFS file system, 448
fstyp — determine file system type, 453
ftpaddhost — set up a virtual FTP host, 454
ftpconfig — set up anonymous FTP, 456
ftpd — File Transfer Protocol Server, 541
ftprestart — restart previously shutdown FTP Servers, 457
ftpshut — close down the FTP Servers at a given time, 458
fuser — identify processes using file or file structure, 460
fwtmp — convert connect accounting records to ASCII, 462

G

generate message IDs — msgid, 1011
get printer configuration — lpget, 768
getdev — lists devices that match given criteria, 463
getdgrp — lists device groups which contain devices that match given criteria, 465
getent — gets a list of entries from administrative databases, 467
gettable — get DoD Internet host table, 469
getty — set terminal type, modes, speed, and line discipline, 470
getvol — verifies device accessibility, 472
GFXconfig — configure the Raptor GFX Graphics Accelerator, 1169
graphical interface for managing DHCP service — dhcpmgr, 283
group file
  check for inconsistencies — grpck, 1292
groupadd — add (create) a new group definition on the system, 476
groupdel — delete a group definition on the system, 478
groupmod — modify a group definition on the system, 479
growfs — grow file system, 481

H

halt — stop the processor, 487
hard disks
  populate Volume Table of Contents — fmthard, 383
hostconfig — configure a system's host parameters, 488
hsfs
  labels for hsfs file systems, provide and print — labelit_hsfs, 687
  mount — mount_hsfs, 987
htable — convert DoD Internet format host table, 490

I

I/O statistics report — iostat, 616
IA Network Booting RPL (Remote Program Load) Server — rpld, 1360
ICMP
  router discovery daemon — in.rdisc, 572
id — return user identity, 491
idsconfig — prepare an iPlanet Directory Server (iDS) to be populated with data and serve LDAP clients, 494
ifconfig — configure network interface parameters, 496
ikeadm — manipulate Internet Key Exchange (IKE) parameters and state, 515
ikecert —, 522
in.comsat — biff server, 527
in.dhcpd — DHCP server, 528
in.fingerd — remote user information server, 536
in.ftpd — File Transfer Protocol Server, 541
in.iked — daemon for the Internet Key Exchange (IKE), 548
in.lpd — print job listener, 557
in.rarpd — Reverse Address Resolution Protocol server, 570
in.rdisc — ICMP router discovery daemon, 572
in.rexecd — remote execution server, 574
in.ripngd — network routing daemon for IPv6, 576
in.rlogind — remote login server, 579
in.routed — network routing daemon, 582
in.rshd — remote shell server, 586
in.rwhod — system status server, 589
in.talkd — talk server, 607
in.telnetd — DARPA TELNET virtual terminal protocol server, 608
in.tftpd — Internet Trivial File Transfer Protocol server, 611
in.tnamed — DARPA trivial name server, 613
in.uucpd — UUCP server, 614
incremental file system dump — ufsdump, 1707
inetd — Internet services daemon, 533
init — process control initialization, 550
  /etc/defaults/init file, 551
  init and System Booting, 550
  inittab Additions, 551
  Run Level Changes, 551
  Run Level Defined, 550

init — process control initialization *(continued)*
telinit, 551
init.wbem — start CIM Boot Manager, 555
init.wbem — stop CIM Object Manager, 555
inode
clear — clri, 226
install — install commands, 591
install
SunOS/SVR4 — suninstall, 1609
installboot procedures — boot, 593
installer — Solaris Web Start installer utility, 595
installf — add a file to software installation database, 596
Internet
biff server — in.comsat, 527
DARPA TELNET virtual terminal protocol server — in.telnetd, 608
DARPA trivial name server — in.tnamed, 613
DHCP server — in.dhcpd, 528
DHCP service configuration — dhcpconfig, 276
ICMP router discovery daemon — in.rdisc, 572
network routing daemon — in.routed, 582
query domain name servers — nslookup, 1113
query domain name servers — nstest, 1121
RARP server — in.rarpd, 570
remote user information server — in.fingerd, 536
send ICMP ECHO_REQUEST packets to network hosts — ping, 1176
server for talk program — in.talkd, 607
services daemon — inetd, 533
system status server — in.rwhod, 589
Trivial File Transfer Protocol server — in.tftpd, 611
Internet Control Message Protocol, *See* ICMP
Internet Protocol
to Ethernet addresses — arp, 70
Interprocess Communication
make FIFO special file — mkfifo, 943
iostat — report I/O statistics, 616
ipsecconf — configure system wifde IPsec policy, 620
ipseckey — manually manipulate an IPsec Security Association Database (SABD), 636

K

kadb — a kernel debugger, 645
kdb5_util — Kerberos Database maintenance utility, 665
kdmconfig — configure or unconfigure keyboard, display, and mouse options, 667
Kerberos Database maintenance utility — kdb5_util, 665
Kerberos warning daemon — ktkt_warnd, 684
kernel — UNIX system executable file containing basic operating system services, 670
kernel
debugger — kadb, 645
display status of loaded modules — modload, 957
load a module — modload, 958
kernel — UNIX system executable file containing basic operating system services SPARC Only, 670, 672
kernel
unload a module — modunload, 959
kernel debugger — kadb, 645
Kernel Symbolic Debugging for OpenBoot Firmware — obpsym, 1140
kernel system event command invocation daemon — syseventconfd, 1648
kernel system event notification daemon — syseventd, 1649
keyserv — server for storing private encryption keys, 673
killall — kill all active processes, 675
kstat — display kernel statistics, 680
ktkt_warnd — Kerberos warning daemon, 684

L

labelit — provide labels for file systems, 685
labelit_hsfs — provide and print labels for hsfs file systems, 687
labelit_ufs — provide labels for ufs, 690
lastlogin — accounting shell procedure, 43

ldapaddent — create LDAP entries from corresponding /etc files, 691
link — link and unlink files and directories, 706
listdgrp — lists members of a device group, 707
listen — network listener daemon, 708
llc2_loop — loopback diagnostics to test the driver, adapter and network., 710
lock
— lockstat, 718
lockd — network lock daemon, 714
lockfs — lock file system, 715
lockstat — report kernel lock and profiling statistics, 718
lofiadm — administer files available as block devices through lofi, 726
log files and system log daemon — syslogd, 1657
logadm —, 731
logins — list user and system login information, 740
loopback diagnostics to test the driver, adapter and network. — llc2_loop, 710
loopback file system
mount — mount, 978
LP print services
administer filters — lpfilter, 755
administer forms — lpforms, 761
configure — lpadmin, 742
register remote systems — lpsystem, 777
set printing queue priorities — lpusers, 778
lpadmin — configure LP print service, 742
lpfilter — administer filters used with LP print service, 755
lpforms — administer forms used with LP print service, 761
lpforms — administer forms used with the LP print service
Adding or Changing a Form, 761
Allowing and Denying Access to a Form, 764
lpforms — administer forms used with the Deleting a Form, 763
lpforms — administer forms used with the LP print service
Listing Form Attributes, 764
Listing the Current Alert, 766
lpforms — administer forms used with the LP print service *(continued)*
Removing an Alert Definition, 766
Setting an Alert to Mount a Form, 764
Terminating an Active Alert, 766
lpget — get printer configuration, 768
lpmove — moves print requests that are queued, 770
lpsched — start the LP print service, 772
lpset — set printing configuration, 773
lpshut — stop the LP print service, 776
lpsystem — register remote systems with LP print service, 777
lpusers — set printing queue priorities, 778
lu, 780
luactivate, 783
lucancel, 785
lucompare, 786
lucreate, 789
lucurr, 797
ludelete, 799
ludesc — display or set boot environment description, 801
ludesc, 801
lufslist, 804
lumake, 806
lumount, 808
lurename, 811
lustatus, 813
luupgrade, 815
luxadm — administration program for the Sun Enterprise Network Array (SENA), RSM, SPARCstorage Array (SSA) subsystems, Sun Fire 880 internal storage subsystem, and individual Fiber Channel Arbitrated Loop (FC_AL) devices, 823
luxadm — administration program for the Sun Enterprise Network Array (SENA), RSM and SPARCstorage Array (SSA) subsystems
Enclosure, 825
Enclosure Services Card Subcommands, 833
Other Expert Mode Subcommands, 833
Pathname, 823
SENA Expert Mode Subcommands, 833
SPARCstorage Array Subcommands, 832
Subcommands, 825
usr/sbin, 837

M

m64config — configure the M64 Graphics Accelerator, 838
mail delivery server — sendmail, 1407
mail.local — store mail in a mailbox, 843
mail utilities
  biff server — in.comsat, 527
  create aliases database — newaliases, 1031
make device_allocate entries — mkdevalloc, 941
make device_maps entries — mkdevmaps, 942
make FIFO — mknod, 956
make FIFO special file — mkfifo, 943
make local NFS file systems available for mounting by remote systems — share_nfs, 1433
make named pipe — mkfifo, 943
make named pipe — mknod, 956
make special file — mknod, 956
makedbm — make NIS ndbm file, 845
manage and view WBEM log files — smlog, 1506
manage bulk operations on user accounts — smmultiuser, 1513
manage diskless client support for a server — smdiskless, 1492
manage email alias entries — smmaillist, 1509
manage entries in the exec_attr database — smexec, 1498
manage group entries — smgroup, 1502
manage jobs in the crontab database — smcron, 1485
manage OS services — smosservice, 1518
manage patches — smpatch, 1523
manage profiles in the prof_attr and exec_attr databases — smprofile, 1529
manage roles and users in role accounts — smrole, 1535
manage serial port — smserialport, 1543
manage user entries — smuser, 1549
manipulate Internet Key Exchange (IKE) parameters and state — ikeadm, 515
manual pages
  create formatted files for — catman, 146
manually manipulate an IPsec Security Association Database (SABD) — ipseckey, 636
map out defective disk blocks — addbadsec, 49
mdmonitord — daemon to monitor metadevices, 851
medstat command, 852
metaclear command, 854
metadb file, 856
metadetach command, 901
metadevadm command, 861
metahs command, 863
metainit command, 866
metaoffline command, 876
metaonline command, 876
metaparam command, 878
metarecover — recover soft partition information, 880
metarename command, 882
metareplace command, 886
metaroot command, 889
metaset command, 891
metastat command, 896
metasync command, 899
metattach command, 901
mibiisa — Sun SNMP Agent, 907
mipagent — Mobile IP agent, 930
mipagentconfig — configure Mobility IP Agent, 933
mipagentstat — show Mobile IP Mobility Agent status, 939
mkdevalloc — make device_allocate entries, 941
mkdevmaps — make device_maps entries, 942
mkfifo — make FIFO special file, 943
mkfile — create a file, 944
mkfs — make file system, 945
mkfs_pcfs — construct a FAT file system, 947
mkfs_udfs — construct a udfs file system, 951
mkfs_ufs — construct ufs, 953
mknod — make special file, 956
Mobile IP agent — mipagent, 930
modify a project's information on the system — projmod, 1264
modify a role's login information on the system — rolemod, 1324
modify device driver attributes — update_drv, 1723
modinfo — display status of loaded kernel modules, 957

modload — load a kernel module, 958
modunload — unload a kernel module, 959
mofcomp — compile MOF files into CIM classes, 960
monacct — accounting shell procedure, 43
monitor program — monitor, 966
monitor system behavior using CPU performance counters — cpustat, 236
mount — mount filesystems and remote resources, 978
mount
  show all remote mounts — showmount, 1440
mount_cachefs — mount CacheFS file systems, 983
mount_hsfs — mount hsfs file systems, 987
mount hsfs file systems — mount_hsfs, 987
mount_nfs — mount remote NFS resources, 989
mount_pcfs — mount pcfs file systems, 997
mount pcfs file systems — mount_pcfs, 997
mount_tmpfs — mount tmpfs, 999
mount_ufs — mount ufs, 1003
mount_xmemfs —, 1007
mountall — mount multiple filesystems, 981
mountd — NFS mount request server, 986
mouse, keyboard, display
  configure or unconfigure — kdmconfig, 667
mpstat — report per-processor statistics, 1009
msgid — generate message IDs, 1011
mvdir — move a directory, 1012

N

name service cache daemon — nscd, 1111
named-bootconf — convert name server configuration files, 1013
named pipe, make — mkfifo, 943
named pipe, make — mknod, 956
named-xfer — ancillary agent for inbound zone transfers, 1014
ncaconfd — Solaris Network Cache and Accelerator (NCA) configuration daemon, 1016
ncheck — generate a list of path names vs i-numbers of all files on special, 1017
ncheck_ufs — generate a list of pathnames vs i-numbers for ufs, 1019
ndd — get and set driver configuration parameters, 1022
netstat — show network status, 1024
netstat — display network status
  Active Sockets (First Form), 1027
  DHCP Interface Information (Sixth Form), 1029
  Interface Status (Third Form), 1028
  Multicast Routing Tables (Fifth Form), 1029
  Network Data Structures (Second Form), 1027
  Routing Table (Fourth Form), 1029
  TCP Sockets, 1027
network routing daemon — in.routed, 582
network
  listener daemon — listen, 708
  lock daemon — lockd, 714
  test connections to hosts using ICMP — ping, 1176
Network File System, *See* NFS
network interface parameters
  configure — ifconfig, 496
network listener service
  administration — nlsadmin, 1105
network packets capture and inspection — snoop, 1564
network routing daemon for IPv6 — in.ripngd, 576
Network Time Protocol daemon — xntpd, 1770
newaliases — make mail aliases database, 1031
newfs — make new file system, 1033
newkey — create a new key in publickey database, 1037
NFS
  crash and recovery functions for locking services — statd, 1597
  create files for use as NFS mounted swap areas — mkfile, 944
  daemon — nfsd, 1039
  display information on resources shared — dfmounts_nfs, 267
  display statistics — nfsstat, 1044
  list available resources from remote systems — dfshares_nfs, 269

NFS *(continued)*
make local NFS filesystem unavailable for mounting by remote systems — unshare_nfs, 1722
mount — mount_nfs, 989
mount request server — mountd, 986
NFS, secure
create a new key in publickey database — newkey, 1037
NFS security
server for storing private encryption keys — keyserv, 673
nfsstat — display NFS statistics, 1044
NIS
binder process — ypbind, 1794
make ndbm file — makedbm, 845
point binder process ypbind at a particular server — ypset, 1804
rebuild database — ypmake, 1797
return current version of a NIS map at a NIS server host — yppoll, 1798
transfer NIS map from a NIS server to host — ypxfr, 1807
NIS+
add /etc files and NIS maps into their corresponding NIS+ tables — nisaddent, 1055
client and root server initialization utility — nisinit, 1072
credential creation tool — nisaddcred, 1049
display the contents of the NIS+ transaction log — nislog, 1080
initialize a domain to store system administration information— nissetup, 1099
nisping — ping NIS+ servers, 1081
nissetup — initialize a NIS+ domain to serve clients, 1099
nisstat — report NIS+ server statistics, 1101
service daemon — rpc.nisd, 1342
update public keys in NIS+ directory object — nisupdkeys, 1103
utility to cache location information about NIS+ servers — nis_cachemgr, 1065
utility to print out the contents of the shared cache file — nisshowcache, 1100
NIS+ credentials for NIS+ principals
initialize — nisclient, 1067
NIS+ password update daemon
— nispasswdd, 1348
— rpc.nispasswdd, 1348
NIS+ servers
set up — nisserver, 1096
NIS+ utility to set server preferences for NIS+ clients — nisprefadm, 1089
NIS server and binder processes
— ypserv, 1801
— ypxfrd, 1801
NIS services, start
— ypstart, 1806
NIS services, stop
— ypstop, 1806
nisaddcred — create NIS+ credentials, 1049
nisaddent — add /etc files and NIS maps into their corresponding NIS+ tables, 1055
nisbackup — backup NIS+ directories, 1062
nisclient — initialize NIS+ credentials for NIS+ principals, 1067
nisldapmaptest — test NIS+ and LDAP mapping configuration files, 1076
nispasswdd — NIS+ password update daemon, 1348
nisping — ping NIS+ servers, 1081
nispopulate — populate the NIS+ tables in a NIS+ domain, 1084
nisprefadm — NIS+ utility to set server preferences for NIS+ clients, 1089
nisrestore — restore NIS+ directory backup, 1093
nisserver — set up NIS+ servers, 1096
nissetup — initialize a domain to serve clients, 1099
nisshowcache — NIS+ utility to print out the contents of the shared cache file, 1100
nisstat — report NIS+ server statistics, 1101
nscd — name service cache daemon, 1111
nslookup — query Internet domain name servers, 1113
nstest — query Internet domain name servers, 1121
nsupdate — update Internet name servers interactively, 1125
ntpdate — set the date and time by way of NTP, 1128
ntpq — standard Network Time Protocol query program, 1131

NIS server and binder processes *(continued)*
Control Message Commands, 1133
Interactive Commands, 1132
ntptrace — trace a chain of NTP hosts back to their master time source, 1138
nulladm — accounting shell procedure, 43

O

obpsym — Kernel Symbolic Debugging for OpenBoot Firmware, 1140
Kernel Symbolic Name Syntax, 1140
OCF server — ocfserv, 1142
ocfserv — OCF server, 1142
operating system
install SunOS/SVR4 — suninstall, 1609
save core dump — savecore, 1403
operating system, bootable
load and execute — boot, 113
output system definition
display current — sysdef, 1642
owner of file
change — chown, 220

P

pack files and file systems in the cache — cachefspack, 139
packages
install, upgrade, and remove software — swmtool, 1629
parse clustertoc file based on dynamic entries — parse_dynamic_clustertoc, 1143
parse_dynamic_clustertoc — parse clustertoc file based on dynamic entries, 1143
passmgmt — password files management, 1144
password file
check for inconsistencies — pwck, 1292
passwords
installs and updates /etc/shadow with information from /etc/passwd — pwconv, 1293
management — passmgmt, 1144
patchadd — apply a patch package to a Solaris 2 system, 1146
patchrm — remove a Solaris patch package and restore previously saved files, 1156
pbind — control and query bindings of processes to processors, 1162
Binding processes, 1162
Querying Bindings, 1163
Unbinding a process, 1163
pcfs
mount — mount_pcfs, 997
PCMCIA user daemon — pcmciad, 1164
pcmciad — PCMCIA user daemon, 1164
per-processor statistics report — mpstat, 1009
perform surface analysis — diskscan, 321
pfinstall — tests installation profiles, 1165
pgxconfig — configure the PGX32 Graphics Accelerator, 1169
PICL daemon — picld, 1174
picld — PICL daemon, 1174
pkgadd — transfer software packages to system, 1181
pkgask — stores answers to an interactive software package, 1184
pkgchk — check package installation accuracy, 1186
pkgrm — remove package from system, 1189
pmconfig — Configure the power management system, 1197
pntadm — DHCP network table management utility, 1199
Point to point protocol daemon — pppd, 1219
pool administration utility — pooladm, 1206
pool binding command — poolbind, 1208
pooladm — pool administration utility, 1206
poolbind — pool binding command, 1208
populate security attribute databases in a name service — smattrpop, 1455
populate the NIS+ tables in a NIS+ domain — nispopulate, 1084
ports
add or remove a service — pmadm, 1192
create /dev entries and inittab entries for serial lines — ports, 1214
enable or disable a service — pmadm, 1192
install or replace a service configuration script — pmadm, 1192
print requested service information — pmadm, 1192
service administration — pmadm, 1192

power manager daemon — powerd, 1218
powerd — power manager daemon, 1218
poweroff — stop the processor, 487
PPP tunneling driver utility — sppptun, 1580
pppd —Point to point protocol daemon, 1219
PPPoE chat utility — pppoec, 1244
PPPoE server daemon — pppoed, 1247
pppoec — PPPoE chat utility, 1244
pppoed — PPPoE server daemon, 1247
pppstats — Print PPP statistics, 1252
praudit — display audit trail, 1255
prctmp — accounting shell procedure, 43
prdaily — accounting shell procedure, 43
prepare an iPlanet Directory Server (iDS) to be populated with data and serve LDAP clients — idsconfig, 494
print device specific information — devinfo, 253
print FRUID-specific information about the FRUs on a system or domain — prtfru, 1274
print PICL tree — prtpicl, 1275
Print PPP statistics, 1252
print queue
  accept or reject requests — accept, reject, 28
print requests
  accept or reject — accept, reject, 28
print service, LP
  — lpmove, 770
printer filters
  add and change — lpfilter, 755
  list attributes — lpfilter, 755
  remove — lpfilter, 755
printer forms
  add or change — lpforms, 761
  delete — lpforms, 763
  list attributes — lpforms, 764
  listing the current alert — lpforms, 766
  provide access — lpforms, 764
  removing an alert definition — lpforms, 766
  setting an alert to mount a form — lpforms, 764
  terminating an active alert — lpforms, 766
printers
  add and change printers — lpadmin, 742
  define alerts for printer faults — lpadmin, 742
  mount printer wheels — lpadmin, 742
  remove printers — lpadmin, 742
printers *(continued)*
  set or change system default destination — lpadmin, 742
  setting priorities — lpusers, 778
printmgr.1m — Solaris Print Manager is a graphical user interface for managing printers in a network, 1256
privatepw — administer FTP Server enhanced group access file, 1258
process accounting
  — acctcms, 36
  — accton, 30
  — acctprc, 41
process scheduler
  administration — dispadmin, 322
processes
  initialization — init, 550
  kill all active processes — killall, 675
  using file or file structure — fuser, 460
prodreg — Solaris Product Registry, 1260
program, standalone
  load and execute — boot, 113
programming tools
  install — install commands, 591
projadd —, 1261
projdel — delete a project from the system, 1263
projmod — modify a project's information on the system, 1264
PROM monitor program — monitor, 966
PROM monitor program
  display and load program — eeprom, 348
prstat — report active process statistics, 1266
prtacct — accounting shell procedure, 43
prtconf — print system configuration information, 1271
prtdiag — print system diagnostic information, 1273
prtfru — print FRUID-specific information about the FRUs on a system or domain, 1274
prtpicl — print PICL tree, 1275
prtvtoc — print the VTOC of a block device, 1276
psradm — change processor operational status, 1278
psrinfo — displays information about processors, 1281

psrset — creation and management of processor sets, 1283
putdev — add, modify remove device entry from table, 1287
putdgrep — add, remove and change device group table, 1290

Q

quick halt
— halt, 487
quota
remote quota daemon — rquotad, 1365
quotas
check file system quota for inconsistencies — quotacheck, 1298
display a user's disk quota and usage — quota, 1297
displays number of blocks in filesystem currently owned by each user — quot, 1295
summarize quotas for a file system — repquota, 1309
turn off file system quotas — quotaoff, 1299
turn on file system quotas — quotaon, 1299
quotas, user
edit — edquota, 346

R

RARP
server — in.rarpd, 570
re-preinstall — restore a system to its original factory condition, 1310
real-time clock and GMT-lag management, provide — rtc, 1368
reboot — restart the operating system, 1304
rebuild NIS database — ypmake, 1797
recover soft partition information — metarecover, 880
reject — reject print requests, 28
remote execution server — in.rexecd, 574
remote login server — in.rlogind, 579
remote shell server — in.rshd, 586
remote shell server — rshd, 586
remote system with debugging on, try to contact — Uutry, 1742
remote login server — in.rlogind
rlogind, 579
remote system with debugging on, try to contact — Uutry
uutry, 1742
remote magtape protocol server — rmt, 1316
Remote Program Load Server, *See* rpld
remote resources
mount or unmount — mount, 978
mount NFS — mount_nfs, 989
remote system
execute remote command requests — uuxqt, 1743
make local resource unavailable for mounting — unshare, 1721
register with LP print service — lpsystem, 777
set system date — rdate, 1303
user information server — in.fingerd, 536
removable media device server — rpc.smserverd, 1355
removable media mounter for CD-ROM and floppy — rmmount, 1313
remove a Solaris patch package and restore previously saved files — patchrm, 1156
removef — remove a file from software database, 1307
removes an application from the Solstice application registry — soldelapp, 1578
report active process statistics — prstat, 1266
report bus-related performance statistics — busstat, 132
report I/O statistics — iostat, 616
report trap statistics — trapstat, 1687
restart previously shutdown FTP Servers — ftprestart, 457
restore
incremental file system restore — ufsrestore, 1714
restore NIS+ directory backup — nisrestore, 1093
return user identity — id, 491
Reverse Address Resolution Protocol, *See* RARP
rlogind — remote login server, 579
rm_drv — remove a device driver from the system, 1306

rmmount — removable media mounter for CD-ROM and floppy, 1313
rmt — remote magtape protocol server, 1316
roleadd — administer a new role account on the system, 1318
roledel — delete a user's login from the system, 1322
rolemod — modify a role's login information on the system, 1324
root directory
change for a command — chroot, 221
route — manually manipulate routing tables, 1327
RPC
kernel statistics server — rpc.rstatd, 1352
network username server — rpc.rusersd, 1353
network wall server — rpc.rwall, 1354
NIS+ service daemon — rpc.nisd, 1342
program number to universal addresses mapping — rpcbind, 1332
remote execution server — rpc.rexd, 1350
remote quota daemon — rquotad, 1365
report information — rpcinfo, 1335
sends one-way stream of packets to host — spray, 1582
server which records packets sent by spray — rpc.sprayd, 1356
server, Autofs mount/unmount requests — automountd, 108
server, NFS mount requests — mountd, 986
RPC, secure
create a new key in publickey database — newkey, 1037
rpc.bootparamd — boot parameter server, 1334
rpc.metad command, 1339
rpc.metamedd command, 1340
rpc.metamhd command, 1341
rpc.nisd — NIS+ service daemon, 1342
rpc.nisd_resolv, 1342, 1347
rpc.nispasswdd — NIS+ password update daemon, 1348
rpc.rexd — remote execution daemon, 1350
rpc.rstatd — kernel statistics server, 1352
rpc.ruserd — network username server, 1353
rpc.rwalld — network wall server, 1354
rpc.smserverd — removable media device server, 1355
rpc.sprayd — records packets sent by spray, 1356
rpc.ypupdated — server for changing NIS information, 1359
rpcbind — converts RPC program numbers to universal addresses, 1332
rpcinfo — report RPC information, 1335
rpld — IA Network Booting RPL (Remote Program Load) Server, 1360
rquotad — remote quota server, 1365
rsh — the restricted shell command interpreter, 1366
rshd — remote shell server, 586
rtc — provide all real-time clock and GMT-lag management, 1368
runacct — run daily accounting, 1369
rwall — write to all users over a network, 1372
rwall daemon
network wall server — rpc.rwalld, 1354

S

sac — Service Access Controller (SAC), 1373
sacadm — Service Access Controller administration, 1376
sadmind — Distributed system administration daemon, 1380
saf — Service Access Facility (SAF), 1384
SAF
format and output port monitor-specific information — ttyadm, 1698
port monitor for terminal ports — ttymon, 1700
savecore — save OS core dump, 1403
scheduler, process
administration — dispadmin, 322
sckmd — Sun Fire 15K key management daemon, 1405
SCSI hardware specific commands for cfgadm — cfgadm_scsi, 184
select a time zone — tzselect, 1705
select naming service for FNS Initial Context — fnselect, 405
select or display devices used as auxiliary console devices — consadm, 227

sendmail — mail delivery system, 1407
set serial line interface
  set operating parameters — syncinit, 1634
server for changing NIS information
  — rpc.ypupdated, 1359
  — ypupdated, 1359
servers
  automountd — mount/unmount request server, 108
  biff server — in.comsat, 527
  DARPA TELNET virtual terminal protocol server — in.telnetd, 608
  DARPA trivial name server — in.tnamed, 613
  DHCP server — in.dhcpd, 528
  in.rexecd — remote execution server, 574
  in.uucpd — UUCP server, 614
  inetd — Internet services daemon, 533
  mountd — mount request server, 986
  RARP server — in.rarpd, 570
  system status server — in.rwhod, 589
  talk program server — in.talkd, 607
  yppasswdd — NIS password server, 1357
servers, NIS+
  location information — nis_cachemgr, 1065
Service Access Controller, *See* SAC
Service Access Facility, *See* SAF
Service Location Protocol Daemon — slpd, 1445
set printing configuration — lpset, 773
set the date and time by way of NTP — ntpdate, 1128
set up a virtual FTP host — ftpaddhost, 454
set up anonymous FTP — ftpconfig, 456
setuname — changes machine information, 1427
sf880drd — Sun Fire 880 Dynamic Reconfiguration daemon, 1428
share — make local resource available for mounting by remote systems, 1430
share_nfs — make local NFS file system available for mounting by remote systems, 1433
shareall — multiple resources, 1432
SHELL variable — rsh, 1366
shells
  the restricted shell command interpreter — rsh, 1366
show Mobile IP Mobility Agent status — mipagentstat, 939
show network status — netstat, 1024
showmount — display remote mounts, 1440
showrev — displays revision information for current hardware and software, 1441
shutacct — accounting shell procedure, 43
shutdown — shutdown system, change system state, 1443
slpd — Service Location Protocol Daemon, 1445
smartcard — configure and administer a smartcard, 1447
smattrpop — populate security attribute databases in a name service, 1455
smc — start the Solaris Management Console, 1460
smccompile — build class list and compile Solaris Management Console service beans for remote use, 1464
smcconf — configure the Solaris Management Console, 1468
smcron — manage jobs in the crontab database, 1485
smdiskless — manage diskless client support for a server, 1492
smexec — manage entries in the exec_attr database, 1498
smgroup — manage group entries, 1502
smlog — manage and view WBEM log files, 1506
smmaillist — manage email alias entries, 1509
smmultiuser — manage bulk operations on user accounts, 1513
smosservice — manage OS services, 1518
smpatch — manage patches, 1523
smprofile — manage profiles in the prof_attr and exec_attr databases, 1529
smrole — manage roles and users in role accounts, 1535
smserialport — manage serial port, 1543
smuser — manage user entries, 1549
snmpdx — Sun Solstice Enterprise Master Agent, 1557
snmpXdmid — Sun Solstice Enterprise SNMP-DMI mapper subagent, 1560
snoop — capture and inspec tnetwork packets, 1564

soconfig — configure transport providers for use by sockets, 1575
software package
add a file to the software installation database — installf, 596
check installation accuracy — pkgchk, 1186
remove a file from software database — removef, 1307
remove package from system — pkgrm, 1189
stores answers to an interactive package — pkgask, 1184
soladdapp — adds an application to the Solstice application registry, 1577
Solaris Network Cache and Accelerator (NCA) configuration daemon — ncaconfd, 1016
Solaris Print Manager is a graphical user interface for managing printers in a network — printmgr.1m, 1256
Solaris Product Registry, *See* ProdReg
Solaris Web Start installer utility — installer, 595
soldelapp — removes an application from the Solstice application registry, 1578
solstice — access system administration tools with a graphical user interface, 1579
SPARCstorage Array
administration program — ssaadm, 1583
special NTP query program — xntpdc, 1786
sppptun — PPP tunneling driver utility, 1580
spray — sends one-way stream of packets to host, 1582
ssaadm — administration program for the SPARCstorage Array, 1583
standard Network Time Protocol query program — ntpq, 1131
Start and stop NIS services — ypstart, 1806
Start NIS services — ypstart, 1806
start Sun WBEM User Manager — wbemadmin, 222, 1755
start the LP print service — lpsched, 772
start the Solaris Management Console — smc, 1460
start WBEM Log Viewer — wbemlogviewer, 1759
startup — accounting shell procedure, 43
startup procedures — boot, 593
statd — network status monitor, 1597
statistics
audit — auditstat, 97
for file system — ff, 366
for ufs file system — ff, 376
I/O — iostat, 616
NFS, display — nfsstat, 1044
per-processor — mpstat, 1009
stop the processor — halt, 487
stop the processor — poweroff poweroff, 487
Stop NIS services — ypstop, 1806
stop the LP print service — lpshut, 776
store mail in a mailbox — mail.local, 843
STREAMS
automatically pushed modules — autopush, 109
detach a name from file descriptor — fdetach, 360
error logger cleanup program — strclean, 1600
error logger daemon — strerr, 1601
port monitor for terminal ports — ttymon, 1700
print trace messages — strace, 1598
su — become super user or another user, 1605
sulogin — single-user mode automatically invoked when system is first started, 1608
Sun Fire 15K key management daemon — sckmd, 1405
Sun Fire 880 Dynamic Reconfiguration daemon — sf880drd, 1428
Sun SNMP Agent — mibiisa, 907
Sun Solstice Enterprise DMI Service Provider — dmispd, 332
Sun Solstice Enterprise Master Agent — snmpdx, 1557
Sun Solstice Enterprise SNMP-DMI mapper subagent — snmpXdmid, 1560
SUNWgfb_config — configure the XVR-1000 Graphics Accelerator, 1610
suninstall — install SunOS/SVR4, 1609
super user command — su, 1605
swap — administer the system swap areas, 1626

swmtool — install, upgrade and remove software packages, 1629
sxconfig — configure contiguous memory for sx video, 1630
sync — update super block, 1633
synchronous serial link
report driver statistics — syncstat, 1640
synchronous serial loopback test program — syncloop, 1637
syncinit — set serial line interface operating parameters., 1634
syncloop — synchronous serial loopback test program, 1637
syncstat — report driver statistics from a synchronous serial link., 1640
sys-unconfig — undo system configuration, 1660
sysdef — displays current system definition, 1642
sysevent event specification administration — syseventadm, 1644
syseventadm — sysevent event specification administration, 1644
syseventconfd — kernel system event command invocation daemon, 1648
syseventd — kernel system event notification daemon, 1649
sysidconfig — execute or define configuration applications, 1651
sysidnet — system configuration, 1654
sysidpm — system configuration, 1654
sysidroot — system configuration, 1654
sysidtool — system configuration, 1654
syslogd — system log message daemon, 1657
system activity reports — sar, 1401
system status server — in.rwhod, 589
system administration
access control lists — admintool, 55
add a file to the software installation database — installf, 596
control for basic administrative functions — uadmin, 1706
create formatted files for manual pages — catman, 146
get entries from administrative database — getent, 467
install commands — install, 591
NIS+ aliases — aliasadm, 65
system administration *(continued)*
system databases — admintool, 55
system configuration
print information — prtconf, 1271
— sysidnet, 1654
— sysidpm, 1654
— sysidroot, 1654
— sysidtool, 1654
undo — sys-unconfig, 1660
system definition
display current — sysdef, 1642
system diagnostic
print information — prtdiag, 1273
system diagnostic messages
error log — dmesg, 325
system information
displays revision information for current hardware and software — showrev, 1441
system log daemon — syslog, 1657
system parameters
change value — setuname, 1427
System Performance Analysis Utilities Tools
system activity report package — sar, 1401
system PROM monitor program — monitor, 966
system shutdown
— halt, 487

T

tapes — adds /dev entries for tape drives attached to the system, 1662
tcxconfig — configure the default linearity of the 24-bit TrueColor Visual for OpenWindows with an S24 frame buffer system (TCX), 1667
telinit — process control initialization, 550
terminal
compare or print terminfo descriptions — infocmp, 537
set type, modes, speed, and line discipline — getty, 470
terminfo
compiler — tic, 1679
test installation profiles — pfinstall, 1165
test NIS+ and LDAP mapping configuration files — nisldapmaptest, 1076

time zone
compiler — zic, 1810
print current time in each named on command line — zdump, 1809
timed event services
daemon for cron — cron, 239
tmpfs
mount — mount_tmpfs, 999
trace a chain of NTP hosts back to their master time source — ntptrace, 1138
trace messages
STREAMS — strace, 1598
transfer software packages to system — pkgadd, 1181
trapstat — report trap statistics, 1687
TTY ports
maintain line settings and hunt sequences — sttydefs, 1603
ttyadm — format and output port monitor-specific information, 1698
ttymon — port monitor for terminal ports, 1700
tunefs — tuneup an existing file system, 1703
turnacct — accounting shell procedure, 43
tzselect — select a time zone, 1705

U

ufs
check and repair — fsck_ufs, 422
construct file system — mkfs_ufs, 953
display amount of disk space — df_ufs, 271
generate a list of pathnames vs i-numbers — ncheck_ufs, 1019
incremental file system restore — ufsrestore, 1714
make image copy of file system— volcopy_ufs, 1750
mount — mount_ufs, 1003
provide labels for ufs — labelit_ufs, 690
ufs file system debugger — fsdb_ufs, 435
ufsdump — incremental file system dump, 1707
umount — unmount file systems and remote resources, 978
umountall — unmount multiple file systems, 981
unconfigure
undo system configuration — sys-unconfig, 1660
UNIX system executable file containing basic operating system services — kernel, 670
unshare — make local resource unavailable for mounting by remote systems, 1721
unshare_nfs — make local NFS filesystem unavailable for mounting by remote systems, 1722
unshareall — multiple resources, 1432
update_drv — modify device driver attributes, 1723
update Internet name servers interactively — nsupdate, 1125
update super block — sync, 1633
user IDs
administer a new user login on the system — useradd, 1724
become super user or another user — su, 1605
user information
server, remote — in.fingerd, 536
user quotas
remote quota daemon — rquotad, 1365
useradd — administer a new user login on the system, 1724
userdel — delete a user's login from the system, 1729
usermod — modify a user's login information on the system, 1731
users
list login information — logins, 740
who is doing what — whodo, 1761
utmp2wtmp — create an entry for each user logged on in file /var/adm/wtmpx, 30
utmpd — utmpx monitoring daemon, 1735
utmpx monitoring daemon — utmpd, 1735
uucheck — check the uucp directories and permissions file, 1736
uucico — file transport program for uucp system, 1737
uucleanup — uucp spool directory clean-up, 1739
uucp
check directories and permissions file — uucheck, 1736

uucp *(continued)*
clean-up of spool directory — uucleanup, 1739
file transport program — uucico, 1737
scheduler for file transport program — uusched, 1741
server — in.uucpd, 614
uusched — scheduler for uucp transport program, 1741
Uutry — try to contact remote system with debugging on, 1742
uutry — try to contact remote system with debugging on, 1742
uuxqt — execute remote command requests, 1743

V

verify Basic Security Module (BSM) configuration files — bsmrecord, 129
virtual console daemon — cvcd, 241
virtual memory
display statistics — vmstat, 1744
vmstat — display virtual memory statistics, 1744
vold — Volume Management daemon to manage CD-ROM and floppy devices, 1751
Volume Management
floppy and CD-ROM management daemon — vold, 1751
removable media mounter — rmmount, 1313
vtoc
print VTOC of a block device — prtvtoc, 1276

W

wall — write to all users, 1753
wbemadmin — start Sun WBEM User Manager, 222, 1755
wbemconfig — convert a JavaSpaces datastore to the newer Reliable Log datastore format, 1758
wbemlogviewer — start WBEM Log Viewer, 1759
whodo — who is doing what, 1761
wracct — write extended accounting records for active processes and tasks, 1763
write extended accounting records for active processes and tasks — wracct, 1763
wtmpfix — correct connect accounting records date/time stamp, 462

X

xntpd — Network Time Protocol daemon, 1770
Access Control Commands, 1774
Authentication Commands, 1773
Authentication Key File Format, 1782
Configuration Commands, 1771
Miscellaneous Commands, 1780
Monitoring Commands, 1777
Primary Clock Support, 1783
xntpdc — special NTP query program, 1786

Y

YP
make ndbm file — makedbm, 845
rebuild database — ypmake, 1797
ypbind — NIS binder process, 1794
ypinit — set up NIS client, 1796
ypmake — rebuild NIS database, 1797
yppoll — return current version of a NIS map at a NIS server host, 1798
yppush — force propagation of changed NIS map, 1799
ypserv — NIS server and binder processes, 1801
ypserv, 1802
ypset — point NIS binder process ypbind at a particular server, 1804
ypstart — Start and stop NIS services, 1806
ypstop — Start and stop NIS services, 1806
ypupdated — server for changing NIS information, 1359
ypxfr — transfer NIS map from a NIS server to host, 1807
ypxfrd — NIS server and binder processes, 1801

yppasswdd — NIS password server, 1357

Z

zdump — prints the current time in each zonename named on command line, 1809
zic — time zone compiler, 1810