Hardware Information

Virtual I/O Server commands

ESCALA POWER5

REFERENCE
86 A1 45EW 00

ESCALA POWER5

Hardware Information

Virtual I/O Server commands

Hardware

July 2006

BULL CEDOC
357 AVENUE PATTON
B.P.20845
49008 ANGERS CEDEX 01
FRANCE

REFERENCE
86 A1 45EW 00

Suggestions and criticisms concerning the form, content, and presentation of this book are invited. A form is provided at the end of this book for this purpose.

To order additional copies of this book or other Bull Technical Publications, you are invited to use the Ordering Form also provided at the end of this book.

Trademarks and Acknowledgements

We acknowledge the right of proprietors of trademarks mentioned in this book.

AIX® is a registered trademark of International Business Machines Corporation, and is being used under licence.

UNIX® is a registered trademark in the United States of America and other countries licensed exclusively through the Open Group.

Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries

The information in this document is subject to change without notice. Bull will not be liable for errors contained herein, or for incidental or consequential damages in connection with the use of this material.

Table of Contents

Table of Contents

Virtual I/O Server commands

Table of Contents

Virtual I/O Server commands

Virtual I/O Server commands

This section lists all Virtual I/O Server commands. The first list has them categorized into high-level administrative categories. The next list has every command listed in alphabetical order.

Command categories

Installation commands

The following commands are the Virtual I/O Server installation sub-commands:

- ioslevel Command
- license Command
- lssw Command
- oem_platform_level Command
- oem_setup_env Command
- remote_management Command
- updateios Command

Volume group commands

The following commands are the Virtual I/O Server volume group sub-commands:

- activatevg Command
- chvg Command
- deactivatevg Command
- exportvg Command
- extendvg Command
- importvg Command
- lsvg Command
- mirrorios Command
- mkvg Command
- redefvg Command
- syncvg Command
- unmirrorios Command

Logical volume commands

- chlv Command
- cplv Command
- extendlv Command
- lslv Command
- mklv Command
- mklvcopy Command
- rmlv Command
- rmlvcopy Command

Physical volume commands

- lspv Command
- migratepv Command

Network commands

- cfglnagg Command
- cfgnamesrv Command
- chtcpip Command
- entstat Command
- hostmap Command

- hostname Command
- lsnetsvc Command
- lstcpip Command
- mktcpip Command
- netstat Command
- optimizenet Command
- ping Command
- rmtcpip Command
- startnetsvc Command
- stopnetsvc Command
- traceroute Command

Device commands

- cfgdev Command
- chdev Command
- chpath Command
- lsdev Command
- lsmap Command
- lspath Command
- mkpath Command
- mkvdev Command
- rmdev Command
- rmpath Command
- rmvdev Command

User ID commands

- chuser Command
- lsuser Command
- mkuser Command
- rmuser Command
- passwd Command

Security commands

- lsfailedlogin Command
- lsgcl Command
- viosecure Command

Maintenance commands

- backupios Command
- bootlist Command
- cattracerpt Command
- chdate Command
- chlang Command
- diagmenu Command
- errlog Command
- fsck Command
- mount Command
- pdump Command
- restorevgstruct Command
- savevgstruct Command
- shutdown Command
- showmount Command
- snap Command
- startsysdump Command
- starttrace Command
- stoptrace Command
- topas Command
- unmount Command

Storage Pool Commands

- chsp Command
- lssp Command
- mksp Command
- mkbdsp Command
- rmbdsp Command

Workload Manager Commands

- wkldagent Command
- wkldmgr Command
- wkldout Command

Standard shell commands

- awk Command
- cat Command
- chmod Command
- clear Command
- cp Command
- crontab Command
- date Command
- ftp Command
- grep Command
- head Command
- ls Command
- man Command
- mkdir Command
- more Command
- mv Command
- rm Command
- sed Command
- stty Command
- tail Command
- tee Command
- vi Command
- wall Command
- wc Command
- who Command

Integrated Virtualization Manager

The following commands are functional only in an Integrated Virtualization Manager environment:

- IVM bkprofdata Command
- IVM chled Command
- IVM chlparutil Command
- IVM chsvcevent Command
- IVM chsyscfg Command
- IVM chsysstate Command
- IVM lpcfgop Command
- IVM lshwres Command
- IVM lsled Command
- IVM lslparutil Command
- IVM lsrefcode Command
- IVM lssvcevents Command
- IVM lssyscfg Command
- IVM lssysconn Command
- IVM mkgencfg Command
- IVM mksvcevent Command
- IVM mksyscfg Command
- IVM mkvt Command
- IVM rmsyscfg Command

- IVM rmvt Command
- IVM rstprofdata Command

Virtual I/O Server command exit status

The following table defines the standard return codes returned by all of the Virtual I/O Server commands. Additional return codes unique to a specific commands are defined with in the individual command description page.

Return Code	Description
0	Success
1	Syntax Error
2	Not Found
1	Command requires an option
1	Command requires the specified option
1	Command requires an attribute
3	Invalid access to execute command
1	Invalid command
1	Invalid flag or argument
1	Invalid option flag
1	Invalid attribute
1	Invalid option combination
1	Specified option requires an argument
1	Specified option requires an attribute
1	Specified option also requires another option
1	Specified option is repeated
1	Attributes cannot be repeated
1	Contains an invalid argument
1	Is invalid
1	Too many arguments
1	Too few arguments
1	Unable to acquire permission to execute command
4	Execution of this command did not complete
10	No device found with physical location
11	Too many matches for physical location
12	Too many matches for physical location code
18	Insufficient memory

activatevg Command

Purpose

Activates a volume group.

Syntax

activatevg **-f** *VolumeGroup*

Description

The **activatevg** command activates the volume group specified by the *VolumeGroup* parameter and all associated logical volumes. When a volume group is activated, physical partitions are synchronized if they are not current.

Flags

-f Allows a volume group to be made active that does not currently have a quorum of available disks.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To activate volume group **vg03**, type:

```
activatevg vg03
```

Related Information

The **mkvg** command, the **chvg** command, the **extendvg** command, the **reducevg** command, the **mirrorios** command, the **unmirrorios** command, the **lsvg** command, the **deactivatevg** command, the **importvg** command, the **exportvg** command, and the **syncvg** command.

awk Command

Purpose

Finds lines in files that match a pattern and performs specified actions on those lines.

Syntax

awk **-F** *Ere* **-v** *Assignment* ... { **-f** *ProgramFile* | **'***Program***'** } *File* ... | *Assignment* ... ...

Description

The **awk** command utilizes a set of user-supplied instructions to compare a set of files, one line at a time, to extended regular expressions supplied by the user. Then actions are performed upon any line that matches the extended regular expressions.

The pattern searching of the **awk** command is more general than that of the **grep** command, and it allows the user to perform multiple actions on input text lines. The **awk** command programming language requires no compiling, and allows the user to use variables, numeric functions, string functions, and logical operators.

The **awk** command is affected by the **LANG**, **LC_ALL**, **LC_COLLATE**, **LC_CTYPE**, **LC_MESSAGES**, **LC_NUMERIC**, **NLSPATH**, and **PATH** environment variables.

The following topics are covered in this article:

- Input for the awk Command
- Output for the awk Command
- File Processing with Records and Fields
- The awk Command Programming Language
    - Patterns
    - Actions
    - Variables
    - Special Variables
- Flags
- Examples

Input for the awk Command

The **awk** command takes two types of input: input text files and program instructions.

Input Text Files

Searching and actions are performed on input text files. The files are specified by:

- Specifying the *File* variable on the command line.
- Modifying the special variables **ARGV** and **ARGC**.
- Providing standard input in the absence of the *File* variable.

If multiple files are specified with the *File* variable, the files are processed in the order specified.

Program Instructions

Instructions provided by the user control the actions of the **awk** command. These instructions come from either the `*Program*' variable on the command line or from a file specified by the **-f** flag together with the *ProgramFile* variable. If multiple program files are specified, the files are concatenated in the order specified and the resultant order of instructions is used.

Output for the awk Command

The **awk** command produces three types of output from the data within the input text file:

- Selected data can be printed to standard output, without alteration to the input file.
- Selected portions of the input file can be altered.
- Selected data can be altered and printed to standard output, with or without altering the contents of the input file.

All of these types of output can be performed on the same file. The programming language recognized by the **awk** command allows the user to redirect output.

File Processing with Records and Fields

Files are processed in the following way:

1. The **awk** command scans its instructions and executes any actions specified to occur before the input file is read.

   The **BEGIN** statement in the **awk** programming language allows the user to specify a set of instructions to be done before the first record is read. This is particularly useful for initializing special variables.
2. One record is read from the input file.

   A record is a set of data separated by a record separator. The default value for the record separator is the new-line character, which makes each line in the file a separate record. The record separator can be changed by setting the **RS** special variable.
3. The record is compared against each pattern specified by the **awk** command's instructions.

   The command instructions can specify that a specific field within the record be compared. By default, fields are separated by white space (blanks or tabs). Each field is referred to by a field variable. The first field in a record is assigned the **$1** variable, the second field is assigned the **$2** variable, and so forth. The entire record is assigned to the **$0** variable. The field separator can be changed by using the **-F** flag on the command line or by setting the **FS** special variable. The **FS** special variable can be set to the values of: blank, single character, or extended regular expression.
4. If the record matches a pattern, any actions associated with that pattern are performed on the record.
5. After the record is compared to each pattern, and all specified actions are performed, the next record is read from input; the process is repeated until all records are read from the input file.
6. If multiple input files have been specified, the next file is then opened and the process repeated until all input files have been read.
7. After the last record in the last file is read, the **awk** command executes any instructions specified to occur after the input processing.

   The **END** statement in the **awk** programming language allows the user to specify actions to be performed after the last record is read. This is particularly useful for sending messages about what work was accomplished by the **awk** command.

The awk Command Programming Language

The **awk** command programming language consists of statements in the form:

Pattern { *Action* }

If a record matches the specified pattern, or contains a field which matches the pattern, the associated action is then performed. A pattern can be specified without an action, in which case the entire line containing the pattern is written to standard output. An action specified without a pattern is performed for every input record.

Patterns

There are four types of patterns used in the **awk** command language syntax:

- Regular Expressions
- Relational Expressions
- Combinations of Patterns

- .BEGIN and END Patterns

Regular Expressions

The extended regular expressions used by the **awk** command are similar to those used by the **grep** command. The simplest form of an extended regular expression is a string of characters enclosed in slashes. For an example, suppose a file named `testfile` had the following contents:

```
smawley, andy
smiley, allen
smith, alan
smithern, harry
smithhern, anne
smitters, alexis
```

Entering the following command line:

```
awk '/smi/' testfile
```

would print to standard output of all records that contained an occurrence of the string `smi`. In this example, the program `'/smi/'` for the **awk** command is a pattern with no action. The output is:

```
smiley, allen
smith, alan
smithern, harry
smithhern, anne
smitters, alexis
```

The following special characters are used to form extended regular expressions:

Character	Function

+	Specifies that a string matches if one or more occurrences of the character or extended regular expression that precedes the **+** (plus) are within the string. The command line:
	`awk '/smith+ern/' testfile`
	prints to standard output any record that contained a string with the characters `smit`, followed by one or more `h` characters, and then ending with the characters `ern`. The output in this example is:
	`smithern, harry`
	`smithhern, anne`

?	Specifies that a string matches if zero or one occurrences of the character or extended regular expression that precedes the ? (question mark) are within the string. The command line:
	`awk '/smith?/' testfile`
	prints to standard output of all records that contain the characters `smit,` followed by zero or one instance of the `h` character. The output in this example is:
	`smith, alan`
	`smithern, harry`
	`smithhern, anne`
	`smitters, alexis`

|	Specifies that a string matches if either of the strings separated by the | (vertical line) are within the string. The command line:
	`awk '/allen`
	`|`
	`alan /' testfile`
	prints to standard output of all records that contained the string `allen` or `alan`. The output in this example is:
	`smiley, allen`
	`smith, alan`

( )

Groups strings together in regular expressions. The command line:

```
awk '/a(ll)?(nn)?e/' testfile
```

prints to standard output of all records with the string `ae` or `alle` or `anne` or `allnne`. The output in this example is:

```
smiley, allen
smithhern, anne
```

{*m*}

Specifies that a string matches if exactly *m* occurrences of the pattern are within the string. The command line:

```
awk '/l{2}/' testfile
```

prints to standard output

```
smiley, allen
```

{*m,*}

Specifies that a string matches if at least *m* occurrences of the pattern are within the string. The command line:

```
awk '/t{2,}/' testfile
```

prints to standard output:

```
smitters, alexis
```

{*m, n*}

Specifies that a string matches if between *m* and *n*, inclusive, occurrences of the pattern are within the string (where *m* <= *n*). The command line:

```
awk '/er{1, 2}/' testfile
```

prints to standard output:

```
smithern, harry
smithern, anne
smitters, alexis
```

String

Signifies that the regular expression matches any characters specified by the *String* variable within the square brackets. The command line:

```
awk '/sma-h/' testfile
```

prints to standard output of all records with the characters `sm` followed by any character in alphabetical order from `a` to `h`. The output in this example is:

```
smawley, andy
```

^ *String*

A ^ (caret) within the (square brackets) and at the beginning of the specified string indicates that the regular expression *does not* match any characters within the square brackets. Thus, the command line:

```
awk '/sm^a-h/' testfile
```

prints to standard output:

```
smiley, allen
smith, alan
smithern, harry
smithhern, anne
smitters, alexis
```

~,!~

Signifies a conditional statement that a specified variable matches (tilde) or does not match (tilde, exclamation point) the regular expression. The command line:

```
awk '$1 ~ /n/' testfile
```

prints to standard output of all records whose first field contained the character n. The output in this example is:

```
smithern, harry
smithhern, anne
```

^

Signifies the beginning of a field or record. The command line:

```
awk '$2 ~ /^h/' testfile
```

prints to standard output of all records with the character h as the first character of the second field. The output in this example is:

```
smithern, harry
```

$

Signifies the end of a field or record. The command line:

```
awk '$2 ~ /y$/' testfile
```

prints to standard output of all records with the character y as the last character of the second field. The output in this example is:

```
smawley, andy
smithern, harry
```

. (period)

Signifies any one character except the terminal new-line character at the end of a space. The command line:

```
awk '/a..e/' testfile
```

prints to standard output of all records with the characters a and e separated by two characters. The output in this example is:

```
smawley, andy
smiley, allen
smithhern, anne
```

*(asterisk)

Signifies zero or more of any characters. The command line:

```
awk '/a.*e/' testfile
```

prints to standard output of all records with the characters a and e separated by zero or more characters. The output in this example is:

```
smawley, andy
smiley, allen
smithhern, anne
smitters, alexis
```

\ (backslash)

The escape character. When preceding any of the characters that have special meaning in extended regular expressions, the escape character removes any special meaning for the character. For example, the command line:

```
/a\/\//
```

would match the pattern a //, since the backslashes negate the usual meaning of the slash as a delimiter of the regular expression. To specify the backslash itself as a character, use a double backslash. See the following item on escape sequences for more information on the backslash and its uses.

Recognized Escape Sequences

The **awk** command recognizes most of the escape sequences used in C language conventions, as well as several that are used as special characters by the **awk** command itself. The escape sequences are:

Escape Sequence	**Character Represented**
\"	\" (double-quotation) mark
\/	/ (slash) character
\ddd	Character whose encoding is represented by a one-, two- or three-digit octal integer, where *d* represents an octal digit
\\\\	\ (backslash) character
\a	Alert character
\b	Backspace character
\f	Form-feed character
\n	New-line character (see following note)
\r	Carriage-return character
\t	Tab character
\v	Vertical tab.

Note: Except in the **gsub**, **match**, **split**, and **sub** built-in functions, the matching of extended regular expressions is based on input records. Record-separator characters (the new-line character by default) cannot be embedded in the expression, and no expression matches the record-separator character. If the record separator is not the new-line character, then the new-line character can be matched. In the four built-in functions specified, matching is based on text strings, and any character (including the record separator) can be embedded in the pattern so that the pattern matches the appropriate character. However, in all regular-expression matching with the **awk** command, the use of one or more NULL characters in the pattern produces undefined results.

Relational Expressions

The relational operators < (less than), > (greater than), <= (less than or equal to), >= (greater than or equal to), = = (equal to), and ! = (not equal to) can be used to form patterns. For example, the pattern:

```
$1 < $4
```

matches records where the first field is less than the fourth field. The relational operators also work with string values. For example:

```
$1 =! "q"
```

matches all records where the first field is not a `q`. String values can also be matched on collation values. For example:

```
$1 >= "d"
```

matches all records where the first field starts with a character that is `a`, `b`, `c`, or `d`. If no other information is given, field variables are compared as string values.

Combinations of Patterns

Patterns can be combined using three options:

- Ranges are specified by two patterns separated with a , (comma). Actions are performed on every record starting with the record that matches the first pattern, and continuing through and including the record that matches the second pattern. For example:

  ```
  /begin/,/end/
  ```

  matches the record containing the string `begin`, and every record between it and the record containing the string `end`, including the record containing the string `end`.
- Parentheses ( ) group patterns together.
- The boolean operators || (or), && (and), and ! (not) combine patterns into expressions that match if they evaluate true, otherwise they do not match. For example, the pattern:

  ```
  $1 == "al" && $2 == "123"
  ```

  matches records where the first field is `al` and the second field is `123`.

BEGIN and END Patterns

Actions specified with the **BEGIN** pattern are performed before any input is read. Actions specified with the **END** pattern are performed after all input has been read. Multiple **BEGIN** and **END** patterns are allowed and processed in the order specified. An **END** pattern can precede a **BEGIN** pattern within the program statements. If a program consists only of **BEGIN** statements, the actions are performed and no input is read. If a program consists only of **END** statements, all the input is read prior to any actions being taken.

Actions

There are several types of action statements:

- Action Statements
- Built-In Functions
- User-Defined Functions
- Conditional Statements
- Output Statements

Action Statements

Action statements are enclosed in { } (braces). If the statements are specified without a pattern, they are performed on every record. Multiple actions can be specified within the braces, but must be separated by new-line characters or ; (semicolons), and the statements are processed in the order they appear. Action statements include:

Arithmetical Statements
: The mathematical operators + (plus), - (minus), / (division), ^ (exponentiation), * (multiplication), % (modulus) are used in the form:

  ```
  Expression Operator Expression
  ```

  Thus, the statement:

  ```
  $2 = $1 ^ 3
  ```

  assigns the value of the first field raised to the third power to the second field.

Unary Statements
: The unary **-** (minus) and unary **+** (plus) operate as in the C programming language:

  ```
  +Expression or -Expression
  ```

Increment and Decrement Statements
: The pre-increment and pre-decrement statements operate as in the C programming language:

```
++Variable or --Variable
```

The post-increment and post-decrement statements operate as in the C programming language:

```
Variable++ or Variable--
```

Assignment Statements

The assignment operators += (addition), -= (subtraction), /= (division), and *= (multiplication) operate as in the C programming language, with the form:

```
Variable += Expression

Variable -= Expression

Variable /= Expression

Variable *= Expression
```

For example, the statement:

```
$1 *= $2
```

multiplies the field variable **$1** by the field variable **$2** and then assigns the new value to **$1**.

The assignment operators ^= (exponentiation) and %= (modulus) have the form:

```
Variable1^=Expression1
```

AND

```
Variable2%=Expression2
```

and they are equivalent to the C programming language statements:

```
Variable1=pow(Variable1, Expression1)
```

AND

```
Variable2=fmod(Variable2, Expression2)
```

where `pow` is the **pow** subroutine and `fmod` is the **fmod** subroutine.

String Concatenation Statements

String values can be concatenated by stating them side by side. For example:

```
$3 = $1 $2
```

assigns the concatenation of the strings in the field variables **$1** and **$2** to the field variable **$3**.

Built-In Functions

The **awk** command language uses arithmetic functions, string functions, and general functions. The close Subroutine statement is necessary if you intend to write a file, then read it later in the same program.

Arithmetic Functions

The following arithmetic functions perform the same actions as the C language subroutines by the same name:

Function	**Action**
atan2(*y*, *x* **)**	Returns arctangent of *y*/*x*.
cos(*x* **)**	Returns cosine of *x*; *x* is in radians.
sin(*x* **)**	Returns sin of *x*; *x* is in radians.
exp(*x* **)**	Returns the exponential function of *x*.
log(*x* **)**	Returns the natural logarithm of *x*.
sqrt(*x* **)**	Returns the square root of *x*.
int(*x* **)**	Returns the value of *x* truncated to an integer.

rand( )	Returns a random number *n*, with 0 <= *n* < 1.
srand(*Expr* **)**	Sets the seed value for the **rand** function to the value of the *Expr* parameter, or use the time of day if the *Expr* parameter is omitted. The previous seed value is returned.

String Functions

The string functions are:

Function	Action
gsub(*Ere*, *Repl*, *In* **)**	Performs exactly as the **sub** function, except that all occurrences of the regular expression are replaced.
sub(*Ere*, *Repl*, *In* **)**	Replaces the first occurrence of the extended regular expression specified by the *Ere* parameter in the string specified by the *In* parameter with the string specified by the *Repl* parameter. The **sub** function returns the number of substitutions. An & (ampersand) appearing in the string specified by the *Repl* parameter is replaced by the string in the *In* parameter that matches the extended regular expression specified by the *Ere* parameter. If no *In* parameter is specified, the default value is the entire record (the **$0** record variable).
index(*String1*, *String2* **)**	Returns the position, numbering from 1, within the string specified by the *String1* parameter where the string specified by the *String2* parameter occurs. If the *String2* parameter does not occur in the *String1* parameter, a 0 (zero) is returned.
length (*String***)**	Returns the length, in characters, of the string specified by the *String* parameter. If no *String* parameter is given, the length of the entire record (the **$0** record variable) is returned.
blength (*String***)**	Returns the length, in bytes, of the string specified by the *String* parameter. If no *String* parameter is given, the length of the entire record (the **$0** record variable) is returned.
substr(*String*, *M*, *N* **)**	Returns a substring with the number of characters specified by the *N* parameter. The substring is taken from the string specified by the *String* parameter, starting with the character in the position specified by the *M* parameter. The *M* parameter is specified with the first character in the *String* parameter as number 1. If the *N* parameter is not specified, the length of the substring will be from the position specified by the *M* parameter until the end of the *String* parameter.
match(*String*, *Ere* **)**	Returns the position, in characters, numbering from 1, in the string specified by the *String* parameter where the extended regular expression specified by the *Ere* parameter occurs, or else returns a 0 (zero) if the *Ere* parameter does not occur. The **RSTART** special variable is set to the return value. The **RLENGTH** special variable is set to the length of the matched string, or to -1 (negative one) if no match is found.

split(*String*, *A*, *Ere* **)**	Splits the string specified by the *String* parameter into array elements *A*1, *A*2, . . ., *An*, and returns the value of the *n* variable. The separation is done with the extended regular expression specified by the *Ere* parameter or with the current field separator (the **FS** special variable) if the *Ere* parameter is not given. The elements in the *A* array are created with string values, unless context indicates a particular element should also have a numeric value.
tolower(*String* **)**	Returns the string specified by the *String* parameter, with each uppercase character in the string changed to lowercase. The uppercase and lowercase mapping is defined by the **LC_CTYPE** category of the current locale.
toupper(*String* **)**	Returns the string specified by the *String* parameter, with each lowercase character in the string changed to uppercase. The uppercase and lowercase mapping is defined by the **LC_CTYPE** category of the current locale.
sprintf(*Format***,** *Expr***,** *Expr***, . . .)**	Formats the expressions specified by the *Expr* parameters according to the **printf** subroutine format string specified by the *Format* parameter and returns the resulting string.

General Functions

The general functions are:

Function	Action
close(*Expression* **)**	Close the file or pipe opened by a **print** or **printf** statement or a call to the **getline** function with the same string-valued *Expression* parameter. If the file or pipe is successfully closed, a 0 is returned; otherwise a non-zero value is returned. The **close** statement is necessary if you intend to write a file, then read the file later in the same program.
system(*Command* **)**	Executes the command specified by the *Command* parameter and returns its exit status. Equivalent to the **system**subroutine.
Expression | **getline** *Variable*	Reads a record of input from a stream piped from the output of a command specified by the *Expression* parameter and assigns the value of the record to the variable specified by the *Variable* parameter. The stream is created if no stream is currently open with the value of the *Expression* parameter as its command name. The stream created is equivalent to one created by a call to the **popen** subroutine with the *Command* parameter taking the value of the *Expression* parameter and the *Mode* parameter set to a value of **r**. Each subsequent call to the **getline** function reads another record, as long as the stream remains open and the *Expression* parameter evaluates to the same string. If a *Variable* parameter is not

specified, the **$0** record variable and the **NF** special variable are set to the record read from the stream.

getline *Variable* **<** *Expression*

Reads the next record of input from the file named by the *Expression* parameter and sets the variable specified by the *Variable* parameter to the value of the record. Each subsequent call to the **getline** function reads another record, as long as the stream remains open and the *Expression* parameter evaluates to the same string. If a *Variable* parameter is not specified, the **$0** record variable and the **NF** special variable are set to the record read from the stream.

getline *Variable*

Sets the variable specified by the *Variable* parameter to the next record of input from the current input file. If no *Variable* parameter is specified, **$0** record variable is set to the value of the record, and the **NF**, **NR**, and **FNR** special variables are also set.

Note: All forms of the **getline** function return 1 for successful input, zero for end of file, and -1 for an error.

User-Defined Functions

User-defined functions are declared in the following form:

```
function Name (Parameter, Parameter,...)  { Statements }
```

A function can be referred to anywhere in an **awk** command program, and its use can precede its definition. The scope of the function is global.

Function parameters can be either scalars or arrays. Parameter names are local to the function; all other variable names are global. The same name should not be used for different entities; for example, a parameter name should not be duplicated as a function name, or special variable. Variables with global scope should not share the name of a function. Scalars and arrays should not have the same name in the same scope.

The number of parameters in the function definition does not have to match the number of parameters used when the function is called. Excess formal parameters can be used as local variables. Extra scalar parameters are initialized with a string value equivalent to the empty string and a numeric value of 0 (zero); extra array parameters are initialized as empty arrays.

When invoking a function, no white space is placed between the function name and the opening parenthesis. Function calls can be nested and recursive. Upon return from any nested or recursive function call, the values of all the calling function's parameters shall be unchanged, except for array parameters passed by reference. The **return** statement can be used to return a value.

Within a function definition, the new-line characters are optional before the opening { (brace) and after the closing } (brace).

An example of a function definition is:

```
function average ( g,n)
  {
        for (i in g)
           sum=sum+gi
        avg=sum/n
        return avg
  }
```

The function average is passed an array, `g`, and a variable, `n`, with the number of elements in the array. The function then obtains an average and returns it.

Conditional Statements

Most conditional statements in the **awk** command programming language have the same syntax and function as conditional statements in the C programming language. All of the conditional statements allow the use of { }

(braces) to group together statements. An optional new-line can be used between the expression portion and the statement portion of the conditional statement, and new-lines or ; (semicolon) are used to separate multiple statements in { } (braces). Six conditional statements in C language are:

Conditional statement	Required syntax or description
if	**if** (*Expression*) { *Statement }* **else** *Action*
while	**while** (*Expression*) { *Statement* }
for	**for** (*Expression* ; *Expression* ; *Expression*) { *Statement* }
break	Causes the program loop to be exited when the **break** statement is used in either a **while** or **for** statement.
continue	Causes the program loop to move to the next iteration when the **continue** statement is used in either a **while** or **for** statement.

Five conditional statements in the **awk** command programming language that do not follow C-language rules are:

Conditional statement	Required syntax or description
for...in	**for** (*Variable* **in** *Array*) { *Statement* } The **for**...**in** statement sets the *Variable* parameter to each index value of the *Array* variable, one index at a time and in no particular order, and performs the action specified by the *Statement* parameter with each iteration. See the **delete** statement for an example of a **for**...**in** statement.
if...in	**if** (*Variable* **in** *Array*) { *Statement* } The **if**...**in** statement searches for the existence of the *Array* element. The statement is performed if the *Array* element is found.

delete	**delete** *Array Expression*
	The **delete** statement deletes both the array element specified by the *Array* parameter and the index specified by the *Expression* parameter. For example, the statements:
	`for (i in g)`
	`   delete gi;`
	would delete every element of the `g` array.

exit	**exit** *Expression* The **exit** statement first invokes all **END** actions in the order they occur, then terminates the **awk** command with an exit status specified by the *Expression* parameter. No subsequent **END** actions are invoked if the **exit** statement occurs within an **END** action.
#	**#** *Comment* The **#** statement places comments. Comments should always end with a new-line but can begin anywhere on a line.
next	Stops the processing of the current input record and proceeds with the next input record.

Output Statements

Two output statements in the **awk** command programming language are:

Output statement	Syntax and description
print	**print** *ExpressionList Redirection Expression*

The **print** statement writes the value of each expression specified by the *ExpressionList* parameter to standard output. Each expression is separated by the current value of the **OFS** special variable, and each record is terminated by the current value of the **ORS** special variable.

The output can be redirected using the *Redirection* parameter, which can specify the three output redirections with the > (greater than), >> (double greater than), and the | (pipe). The *Redirection* parameter specifies how the output is redirected, and the *Expression* parameter is either a path name to a file (when *Redirection* parameter is > or >>) or the name of a command (when the *Redirection* parameter is a |).

printf

printf *Format* , *ExpressionList Redirection Expression*

The **printf** statement writes to standard output the expressions specified by the *ExpressionList* parameter in the format specified by the *Format* parameter. The **printf** statement functions exactly like the **printf** command, except for the `c` conversion specification (%c). The *Redirection* and *Expression* parameters function the same as in the **print** statement.

For the `c` conversion specification: if the argument has a numeric value, the character whose encoding is that value will be output. If the value is zero or is not the encoding of any character in the character set, the behavior is undefined. If the argument does not have a numeric value, the first character of the string value will be output; if the string does not contain any characters the bahaviour is undefined.

Note: If the *Expression* parameter specifies a path name for the *Redirection* parameter, the *Expression* parameter should be enclosed in double quotes to insure that it is treated as a string.

Variables

Variables can be scalars, field variables, arrays, or special variables. Variable names cannot begin with a digit.

Variables can be used just by referencing them. With the exception of function parameters, they are not explicitly declared. Uninitialized scalar variables and array elements have both a numeric value of 0 (zero) and a string value of the null string (" ").

Variables take on numeric or string values according to context. Each variable can have a numeric value, a string value, or both. For example:

```
x = "4" + "8"
```

assigns the value of `12` to the variable `x`. For string constants, expressions should be enclosed in " " (double quotation) marks.

There are no explicit conversions between numbers and strings. To force an expression to be treated as a number, add 0 (zero) to it. To force an expression to be treated as a string, append a null string (" ").

Field Variables

Field variables are designated by a **$** (dollar sign) followed by a number or numerical expression. The first field in a record is assigned the **$1** variable , the second field is assigned to the **$2** variable, and so forth. The **$0** field variable is assigned to the entire record. New field variables can be created by assigning a value to them. Assigning a value to a non-existent field, that is, any field larger than the current value of **$NF** field variable, forces the creation of any intervening fields (set to the null string), increases the value of the **NF** special variable, and forces the value of **$0** record variable to be recalculated. The new fields are separated by the current field separator (which is the value of the **FS** special variable). Blanks and tabs are the default field separators. To change the field separator, use the **-F** flag, or assign the **FS** special variable a different value in the **awk** command program.

Arrays

Arrays are initially empty and their sizes change dynamically. Arrays are represented by a variable with subscripts in (square brackets). The subscripts, or element identifiers, can be numbers of strings, which provide a type of associative array capability. For example, the program:

```
/red/  { x"red"++ }
/green/ { y"green"++ }
```

increments counts for both the `red` counter and the `green` counter.

Arrays can be indexed with more than one subscript, similar to multidimensional arrays in some programming languages. Because programming arrays for the **awk** command are really one dimensional, the comma-separated subscripts are converted to a single string by concatenating the string values of the separate expressions, with each expression separated by the value of the **SUBSEP** environmental variable. Therefore, the following two index operations are equivalent:

```
xexpr1, expr2,...exprn
```

AND

```
xexpr1SUBSEPexpr2SUBSEP...SUBSEPexprn
```

When using the **in** operator, a multidimensional *Index* value should be contained within parentheses. Except for the **in** operator, any reference to a nonexistent array element automatically creates that element.

Special Variables

The following variables have special meaning for the **awk** command:

Special variable	Description
ARGC	The number of elements in the **ARGV** array. This value can be altered.
ARGV	The array with each member containing one of the *File* variables or *Assignment* variables, taken in order from the command line, and numbered from 0 (zero) to **ARGC** -1. As each input file is finished, the next member of the **ARGV** array provides the name of the next input file, unless: • The next member is an *Assignment* statement, in which case the assignment is evaluated. • The next member has a null value, in which case the member is skipped. Programs can skip selected input files by setting the member of the **ARGV** array that contains that input file to a null value. • The next member is the current value of **ARGV ARGC** -1, which the **awk** command interprets as the end of the input files.
CONVFMT	The **printf** format for converting numbers to strings (except for output statements, where the **OFMT** special variable is used). The default is "%.6g".
ENVIRON	An array representing the environment under which the **awk** command operates. Each element of the array is of the form: **ENVIRON "***Environment VariableName***"** = *EnvironmentVariableValue* The values are set when the **awk** command begins execution, and that environment is used until the end of execution, regardless of any modification of the **ENVIRON** special variable.
FILENAME	The path name of the current input file. During the execution of a **BEGIN** action, the value of **FILENAME** is undefined. During the execution of an **END** action, the value is the name of the last input file processed.
FNR	The number of the current input record in the current file.
FS	The input field separator. The default value is a blank. If the input field separator is a blank, any number of locale-defined spaces can separate fields. The **FS** special variable can take two additional values: • With **FS** set to a single character, fields are separated by each single occurrence of the character. • With **FS** set to an extended regular expression, each occurrence of a sequence matching the extended regular expression separates fields.
NF	The number of fields in the current record, with a limit of 99. Inside a **BEGIN** action, the **NF** special variable is undefined unless a **getline** function without a *Variable* parameter has been issued previously. Inside an **END** action, the **NF** special variable retains the value it had for

the last record read, unless a subsequent, redirected, **getline** function without a *Variable* parameter is issued prior to entering the **END** action.

NR The number of the current input record. Inside a **BEGIN** action the value of the **NR** special variable is 0 (zero). Inside an **END** action, the value is the number of the last record processed.

OFMT The **printf** format for converting numbers to strings in output statements. The default is "% .6g".

OFS The output field separator (default is a space).

ORS The output record separator (default is a new-line character).

RLENGTH The length of the string matched by the **match** function.

RS Input record separator (default is a new-line character). If the **RS** special variable is null, records are separated by sequences of one or more blank lines; leading or trailing blank lines do not result in empty records at the beginning or end of input; and the new-line character is always a field separator, regardless of the value of the **FS** special variable.

RSTART The starting position of the string matched by the **match** function, numbering from 1. Equivalent to the return value of the **match** function.

SUBSEP Separates multiple subscripts. The default is \031.

Flags

-f *ProgramFile* Obtains instructions for the **awk** command from the file specified by the *ProgramFile* variable. If the **-f** flag is specified multiple times, the concatenation of the files, in the order specified, will be used as the set of instructions.

-F *Ere* Uses the extended regular expression specified by the *Ere* variable as the field separator. The default field separator is a blank.

-v *Assignment* Assigns a value to a variable for the **awk** command's programming language. The *Assignment* parameter is in the form of *Name = Value*. The *Name* portion specifies the name of the variable and can be any combination of underscores, digits, and alphabetic characters, but it must start with either an alphabetic character or an underscore. The *Value* portion is also composed of underscores, digits, and alphabetic characters, and is treated as if it were preceded and followed by a " (double-quotation character, similar to a string value). If the *Value* portion is numeric, the variable will also be assigned the numeric value.

The assignment specified by the **-v** flag occurs before any portion of the **awk** command's program is executed, including the **BEGIN** section.

Assignment Assigns a value to a variable for the **awk** command's programming language. It has the same form and function as the *Assignment* variable with the **-v** flag, except for the time each is processed. The *Assignment* parameter is processed just prior to the input file (specified by the *File* variable) that follows it on the command line. If the *Assignment* parameter is specified just prior to the first of multiple input files, the assignments are processed just after the **BEGIN** sections (if any). If an *Assignment* parameter occurs after the last file, the assignment is processed before the **END** sections (if any). If no input files are specified, the assignments are processed the standard input is read.

File Specifies the name of the file that contains the input for processing. If no *File* variable is specified, or if a **-** (minus) sign is specified, standard input is processed.

'*Program***'** Contains the instructions for the **awk** command. If the **-f** flag is not specified, the *Program* variable should be the first item on the command line. It should be bracketed by ' ' (single quotes).

Exit Status

This command returns the following exit values:

0	Successful completion.
>0	An error occurred.

Examples

1. To display the lines of a file that are longer than 72 characters, enter:

```
awk  'length  >72'  chapter1
```

This selects each line of the `chapter1` file that is longer than 72 characters and writes these lines to standard output, because no *Action* is specified. A tab character is counted as 1 byte.

2. To display all lines between the words `start` and `stop,` including `"start"` and `"stop",` enter:

```
awk  '/start/,/stop/'  chapter1
```

3. To run an **awk** command program, `sum2.awk,` that processes the file, `chapter1`, enter:

```
awk  -f  sum2.awk  chapter1
```

The following program, `sum2.awk`, computes the sum and average of the numbers in the second column of the input file, `chapter1`:

```
    {
       sum += $2
    }
END {
       print "Sum: ", sum;
       print "Average:", sum/NR;
    }
```

The first action adds the value of the second field of each line to the variable `sum`. All variables are initialized to the numeric value of 0 (zero) when first referenced. The pattern **END** before the second action causes those actions to be performed after all of the input file has been read. The **NR** special variable, which is used to calculate the average, is a special variable specifying the number of records that have been read.

4. To print the first two fields in opposite order, enter:

```
awk '{ print $2, $1 }' chapter1
```

5. The following **awk** program

```
awk -f sum3.awk chapter2
```

prints the first two fields of the file `chapter2` with input fields separated by comma and/or blanks and tabs, and then adds up the first column, and prints the sum and average:

```
BEGIN  {FS = ",| \t+"}
       {print $1, $2}
       {s += $1}
END    {print "sum is",s,"average is", s/NR }
```

Related Information

The **grep** command and the **sed** command.

backupios Command

Purpose

Creates an installable image of the root volume group either onto a bootable tape, file system, or DVD.

Syntax

backupios **-file** *DirectoryName* **-mksysb** **-nosvg**

backupios **-tape** *Device* **-nopak** **-verify** **-nosvg**

backupios **-cd** *Device* **-udf** | **-cdformat** **-accept** **-nosvg**

Description

The **backupios** command creates a backup of the Virtual I/O Server and places it onto a filesystem, bootable tape, or DVD. You can use this backup to reinstall a system to its original state after it has been corrupted. If you create the backup on tape , the tape is bootable and includes the installation programs needed to install from the backup.

If the **-cd** flag is specified, the **backupios** command creates a system backup image to DVD-RAM media. If you need to create multi-volume discs because the image does not fit on one disc, the **backupios** command gives instructions for disk replacement and removal until all the volumes have been created.

Note: Vendor disc drives may support burning to additional disc types, such as CD-RW and DVD-R. Refer to the documentation for your drive to determine which disc types are supported.

If the **-file** flag is specified, the **backupios** command creates a system backup image to the path specified. The file system must be mounted and writable by the Virtual I/O Server root user prior to running the **backupios** command (see mount Command command for details). This backup can be reinstalled from the HMC using the **installios** command.

Prior to backing up the root volume group, the **backupios** command saves the structure of all user-defined volume groups by calling the **savevgstruct** command for each volume group defined. To avoid having all user-defined volume groups backed up, use the **-nosvg** flag.

Note: The **backupios** command backs up only the volume group structures that are activated. The structures of volumes groups that are deactivated are not backed up.

Flags

-accept	Accepts licenses.
-cd	The Virtual I/O Server backup is placed onto DVD-RAM media.
-cdformat	Creates final CD images that are DVD sized (up to 4.38 GB).
-file	The Virtual I/O Server backup is placed in a file.
-mksysb	When the **-mksyb** flag is used, the resources used by the **installios** command are not saved in the image.
-nopak	Disables software packing of the files as they are backed up. Some tape drives use their own packing or compression algorithms.
-nosvg	Prevents the volume groups structure of user defined volume groups from being saved as part of the **backupios** process.
-tape	The Virtual I/O Server backup is placed onto a tape.
-udf	Creates a UDF (Universal Disk Format) file system on DVD-RAM media. The default format is Rock Ridge (ISO9660).
-verify	Verifies a tape backup. This flag causes the **backupios** command to verify the file header of each file on the backup tape and report any read errors as they occur.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To generate a backup to the optical device named **/dev/cd1**, type:

   ```
   backupios -cd /dev/cd1 -cdformat
   ```
2. To generate a system backup to a tape device named **/dev/rmt0**, type:

   ```
   backupios -tape /dev/rmt0
   ```

Related Information

The **mount** command and the **savevgstruct** command.

IVM bkprofdata Command

Purpose

Backs up profile data. This command is operable only in an Integrated Virtualization Manager environment.

Syntax

bkprofdata **-o** backup **-f** *BackupFile* **-m** *ManagedSystem*

Description

The **bkprofdata** command performs a backup of logical partition configuration information to a file. The file can later be used to restore the partition configuration with the **rstprofdata** command.

Flags

-f *BackupFile*
The name of the file to write the dump to in the current working directory. If not specified, the default file is **/var/adm/lpm/profile.bak**.

-o *Operation*
The type of operation:

backup
Backs logical partition configuration data to a file.

-m *ManagedSystem*
The name of the managed system. This attribute is optional because there is only one system to manage. The name may either be the user-defined name for the managed system, or be in the form tttt-mmm*sssssss, where tttt is the machine type, mmm is the model, and sssssss is the serial number of the managed system.

Exit Status

This command has a return code of zero on success.

Security

This command is available to all users.

Examples

1. To backup the partition configuration data to **/var/adm/lpm/profile.bak**, type:

   ```
   bkprofdata -o backup
   ```

2. To backup the partition configuration data to **lparData.bak**, type:

   ```
   bkprofdata -o backup -f lparData.bak
   ```

Related Information

The **rstprofdata** command.

bootlist Command

Purpose

Displays and alters the list of boot devices available to the system.

Syntax

bootlist **-mode** *mode* { **-ls** **-rm** | *Device* **-attr** *Attribute=Value* ... ... }

Description

The **bootlist** command allows the user to display and alter the list of possible boot devices from which the system may be booted. When the system is booted, it will scan the devices in the list and attempt to boot from the first device it finds containing a boot image. This command supports the updating of the following:

- Service boot list. The service list designates possible boot devices for when the system is booted in service mode. How a system is booted in service mode is hardware-platform dependent. It may require a key switch to be turned to the Service position, a particular function key to be pressed during the boot process, or some other mechanism, as defined for the particular hardware platform.
- Previous boot device entry. This entry designates the last device from which the system booted. Some hardware platforms may attempt to boot from the previous boot device before looking for a boot device in one of the other lists.

The **bootlist** command supports the specification of generic device types as well as specific devices for boot candidates. Devices in the boot device list occur in the same order as devices listed on the invocation of this command.

The selection of the boot list to display or alter is made with the **-mode** *mode* option, where the *mode* variable is one of the keywords: **service**, **normal**, **both**, or **prevboot**. If the **both** keyword is specified, then both the normal boot list and the service boot list will be displayed, or if being altered, will be set to the same list of devices. If the **prevboot** keyword is specified, the only alteration allowed is with the **-rm** flag . The **-rm** flag invalidates the boot list specified by the **-mode** flag.

The devices currently in the boot list may be displayed by using the **-ls** flag. The list of devices that make up the specified boot list will be displayed, one device per line. If a device specified in the boot list is no longer present on the system, a '-' is displayed instead of a name.

Note: When you add a hot plug adapter to the system, that adapter and its child devices might not be available for specification as a boot device when you use the **bootlist** command. You may be required to reboot your system to make all potential boot devices known to the operating system.

Device Choices

The device name specified on the command line can occur in one of two different forms:

- It can indicate a specific device by its device logical name.
- It can indicate a generic or special device type by keyword. The following generic device keywords are supported:

fd	Any standard I/O-attached diskette drive
scdisk	Any SCSI-attached disk (including serial-link disk drives)
badisk	Any direct bus-attached disk
cd	Any SCSI-attached CD-ROM
rmt	Any SCSI-attached tape device
ent	Any Ethernet adapter
fddi	Any Fiber Distributed Data Interface adapter

When a specific device is to be included in the device list, the device's logical name (used with system management commands) must be specified. This logical name is made up of a prefix and a suffix. The suffix is generally a number and designates the specific device. The specified device must be in the Available state. If it is not, the update to the device list is rejected and this command fails. The following devices and their associated logical names are supported (where the bold type is the prefix and the *xx* variable is the device-specific suffix):

fd*xx*	Diskette-drive device logical names
hdisk*xx*	Physical-volume device logical names
cd*xx*	SCSI CD-ROM device logical names
rmt*xx*	Magnetic-tape device logical names
ent*xx*	Ethernet-adapter logical names
fddi*xx*	Fiber Distributed Data Interface adapter logical names

Attribute Choices

Attributes are extra pieces of information about a device that the user supplies on the command line. Since this information is specific to a particular device, generic devices do not have attributes. Attributes apply to the device that immediately precedes them on the command line, which allows attributes to be interspersed among devices on the command line. Currently, only network devices have attributes. These are:

bserver	IP address of the BOOTP server
gateway	IP address of the gateway
client	IP address of the client
speed	Network adapter speed
duplex	The mode of the network adapter

Error Handling

If this command returns with an error, the device lists are not altered. The following device list errors are possible:

- If the user attempts to display or alter a boot list that is not supported by the hardware platform, the command fails, indicating the mode is not supported.
- If the user attempts to add too many devices to the boot list, the command fails, indicating that too many devices were requested. The number of devices supported varies depending on the device selection and the hardware platform .
- If an invalid keyword, invalid flag, or unknown device is specified, the command fails with the appropriate error message.
- If a specified device is not in the `Available` state, the command fails with the appropriate error message.

Flags

Device	Provides the names of the specific or generic devices to include in the boot list.
-attr *Attribute=Value*	Specifies the device attribute value pairs to be used instead of the defaults. The *Attribute=Value* variable can be used to specify one attribute value pair or multiple attribute value pairs for one **-attr** flag. If you use an **-attr** flag with multiple attribute value pairs, the list of pairs must be enclosed in quotation marks with a blank space between the pairs. For example, entering `-attr Attribute=Value` lists one attribute value pair per flag, while entering `-attr 'Attribute1=Value1 Attribute2=Value2'` lists more than one attribute value pair.
-mode *Mode*	Specifies which boot list to display or alter. Possible values for the *mode* variable are **normal**, **service**, **both**, or **prevboot**.
-ls	Indicates that the specified boot list is to be displayed after any specified alteration is performed. The output is a list of device names.
-rm	Indicates that the device list specified by the **-mode** flag should be invalidated.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To invalidate the Service mode boot list, type:

   ```
   bootlist -mode service -rm
   ```
2. To make a boot list for Normal mode with devices listed on the command line, type:

   ```
   bootlist -mode normal hdisk0 hdisk1 rmt0 fd
   ```
3. To attempt to boot through a gateway using Ethernet, and then try other devices, type a command similar to the following. Substitute the IP addresses specified in the example with your IP addresses.

   ```
   bootlist -mode normal ent0 -attr gateway=129.35.21.1 bserver=129.12.2.10 \
   client=129.35.9.23 hdisk0 rmt0 tok0 bserver=129.35.10.19 hdisk1
   ```

Related Information

The **backupios** command.

cat Command

Purpose

Concatenates or displays files.

Syntax

cat -q -r -s -S -u -n -b -v -e -t - | *File* ...

Description

The **cat** command reads each *File* parameter in sequence and writes it to standard output. If you do not specify a file name, the **cat** command reads from standard input. You can also specify a file name of **-** (dash) for standard input.

Attention: Do not redirect output to one of the input files using the redirection symbol, > (caret). If you do this, you lose the original data in the input file because the shell truncates the file before the **cat** command can read it.

Flags

-b Omits line numbers from blank lines, when specified with the **-n** flag.

-e Displays a $ (dollar sign) at the end of each line, when specified with the **-v** flag.

-n Displays output lines preceded by line numbers, numbered sequentially from 1.

-q Does not display a message if the **cat** command cannot find an input file. This flag is identical to the **-s** flag.

-r Replaces multiple consecutive empty lines with one empty line. This flag is identical to the **-S** flag.

-s Does not display a message if the **cat** command cannot find an input file. This flag is identical to the **-q** flag. **Note:** Previously, the **-s** flag handled tasks now assigned to the **-S** flag.

-S Replaces multiple consecutive empty lines with one empty line. This flag is identical to the **-r** flag.

-t Displays tab characters as `^I` if specified with the **-v** flag.

-u Does not buffer output. The default is buffered output.

-v Displays nonprinting characters as visible characters, with the exception of tabs, new-lines, and form-feeds. ASCII control characters (octal 000–037) are printed as `^`*n*, where *n* is the corresponding ASCII character in the octal range 100–137 (@, A, B, C,..., X, Y, Z, , \, , ^, and _); the DEL character (octal 0177) is printed as `^?`. Other non-printable characters are printed as `M-`*x*, where *x* is the ASCII character specified by the low-order seven bits.

When used with the -v option, the following options may be used:

-e

A $ character will be printed at the end of each line prior to a new line.

-t

Tabs will be printed as `^I` and form feeds will be printed as `^L`

The -e and -t options are ignored if the -v option is not specified.

- Allows standard input to the **cat** command.

Exit Status

This command returns the following exit values:

0	All input files were output successfully.
>0	An error occurred.

Examples

Attention: Do not redirect output to one of the input files using the redirection symbol, > (caret).

1. To display a file at the workstation, enter:

   ```
   cat notes
   ```

   This command displays the data in the `notes` file.
2. To concatenate several files, enter:

   ```
   cat section1.1 section1.2 section1.3 >section1
   ```

   This command creates a file named `section1` that is a copy of `section1.1` followed by `section1.2` and `section1.3`.
3. To suppress error messages about files that do not exist, enter: `cat -q section2.1 section2.2 section2.3 >section2`
   If `section2.1` does not exist, this command concatenates `section2.2` and `section2.3`. The result is the same if you do not use the **-q** flag, except that the **cat** command displays the error message:

   ```
   cat: cannot open section2.1
   ```

   You may want to suppress this message with the **-q** flag when you use the **cat** command in shell procedures.
4. To append one file to the end of another, enter:

   ```
   cat section1.4 >> section1
   ```

   The >> (two carets) appends a copy of `section1.4` to the end of `section1`. If you want to replace the file, use the > (caret).
5. To add text to the end of a file, enter:

   ```
   cat >>notes
   Get milk on the way home
   Ctrl-D
   ```

   This command adds `Get milk on the way home` to the end of the file called `notes`. The **cat** command does not prompt; it waits for you to enter text. Press the Ctrl-D key sequence to indicate you are finished.
6. To concatenate several files with text entered from the keyboard, enter:

   ```
   cat section3.1 - section3.3 >section3
   ```

   This command concatenates the file `section3.1` with text from the keyboard (indicated by the minus sign), and the file `section3.3`, then directs the output into the file called `section3`.

Files

/usr/bin/cat	Contains the **cat** command.

Related Information

The **cp** command.

cattracerpt Command

Purpose

Formats a report from the trace log.

Syntax

cattracerpt -hookid *List* | **-lshid -outfile** *FileName*

Description

The **cattracerpt** command reads the trace log, formats the trace entries, and writes a report to standard output.

Flags

-hookid *List* Limits report to hook IDs specified with the *List* variable. The *List* parameter items must be separated by commas.

-lshid Displays the list of hook IDs. The **cattracerpt -listid** command can be used with the **starttrace -event** command that includes IDs of trace events.

-outfile *File* Writes the report to a file instead of to standard output.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To send a trace report to the **newfile** file, enter:

   ```
   cattracerpt -outfile newfile
   ```
2. To display a list of hook IDs, enter:

   ```
   cattracerpt -lshid
   ```

Related Information

The **starttrace** command, and the **stoptrace** command.

cfgdev Command

Purpose

Configures devices in the Virtual I/O Server.

Syntax

cfgdev **-dev** *Name*

Description

The **cfgdev** command configures devices. If there are any devices detected that have no device software installed when configuring devices, the **cfgdev** command returns a warning message with the name or a list of possible names for the device package that must be installed. If the specific name of the device package is determined, it is displayed as the only package name, on a line below the warning message. If the specific name cannot be determined, a colon-separated list of possible package names is displayed on a single line. A package name or list of possible package names is displayed for each of the devices, if more than one device is detected without its device software.

The system displays the following warning message when devices without their device software are detected:

```
cfgdev: 0514-621 WARNING: The following device packages are
        required for device support but are not currently
        installed.
devices.pci.22100020
devices.pci.14101800
devices.pci.scsi:devices.pci.00100300:devices.pci.NCR.53C825
```

In this example, two devices were found that had missing software, and the **cfgdev** command displayed the names of the device packages that must be installed. A third device that also has missing software was found, but in this case, the **cfgdev** command displays several possible device package names.

Attention: To protect the Configuration database, the **cfgdev** command is not interruptible. Stopping this command before execution is complete could result in a corrupted database.

Flags

-dev *Name*	Specifies the named device to configure along with its children.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To configure detected devices attached to the **scsi0** adapter, type:

   ```
   cfgdev -dev scsi0
   ```

Related Information

The **chdev** command, the **chpath** command, the **lsdev** command, the **lsmap** command, the **lspath** command, the **mkpath** command, the **mkvdev** command, the **rmdev** command, and the **rmpath** command.

cfglnagg Command

Purpose

Add or remove adapters from a Link Aggregation or change a Link Aggregation attributes.

Syntax

cfglnagg { **-add** **-backup** | **-rm** } *LinkAggregation Adapter*

cfglnagg **-f** **-attr** *Attribute=NewValue LinkAggregation* ...

Description

This command adds adapters to a Link Aggregation or removes adapters from a Link Aggregation. This command can also be used to modify Link Aggregation attributes. It is not necessary to detach the Link Aggregation's interface to add or remove adapters or modify most Link Aggregation attributes.

Changing the **use_jumbo_frame** attribute requires the Link Aggregation to be detached. If this attribute is specified, the **cfglnagg** command detaches the Link Aggregation before modifying the **use_jumbo_frame** attributes, then brings the Link Aggregation back up. The user is prompted to continue unless the **-f** flags is specified.

Use the **lsdev** command with the **-attr** flag to view all attributes associated with a Link Aggregation device, including adapters.

Note: To create a Link Aggregation see the **mkvdev** command.

Flags

-add	Adds the specified Adapter to the specified Link Aggregation. If the adapter must be added as a backup adapter, the **-backup** flag must be specified.
-attr	Specifies an attribute of the specified Link Aggregation.
-backup	Specifies that the *Adapter* is being added as a backup adapter.
-f	Instructs the **cgflnagg** command to not prompt the user.
-rm	Deletes the specified Adapter from the specified Link Aggregation. The specified adapter can be either a primary or backup adapter.

Parameters

Adapter	Specifies the adapter to add or delete.
LinkAggregation	Specifies the Link Aggregation.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To add adapter **ent8** to Link Aggregation **ent3**, type:

   ```
   cfglnagg -add ent3 ent8
   ```

Related Information

The **mktcpip** command, the **mkvdev** command, the **hostname** command, the **startnetsvc** command, the **stopnetsvc** command, the **netstat** command, the **entstat** command, the **cfgnamesrv** command, the **hostmap** command, the **traceroute** command, the **ping** command, the **optimizenet** command.

cfgnamesrv Command

Purpose

Directly manipulates domain name server entries for local resolver routines in the system configuration database.

Syntax

To Add a Name Server Entry

cfgnamesrv **-add** { **-ipaddr** *IPAddress* | **-dname** *DomainName* | **-slist** *SearchList*}

To Delete a Name Server Entry

cfgnamesrv **-rm** { **-ipaddr** *IPAddress* | **-domain** | **-slist**}

To Change a Name Server Entry

cfgnamesrv **-ch** *DomainName*

To Display a Name Server Entry

cfgnamesrv **-ls** **-namesrv** | **-domain** | **-slist**

To Create the Configuration Database File

cfgnamesrv **-mk** **-ipaddr** *IPAddress* **-dname** *DomainName* **-slist** *SearchList*

To Change a Search List Entry

cfgnamesrv **-chslist** *SearchList*

Description

The **cfgnamesrv** command adds or deletes domain name server entries for local resolver routines in the system configuration database. To add a name server entry, specify an Internet Protocol address and, optionally, a domain name.

The **cfgnamesrv** command can show one or all domain name server entries in the system configuration database. There are three types of domain name server entries:

- A domain entry identifying the name of the local Internet domain.
- A name server entry that identifies the Internet address of a domain name server for the local domain. The address must be in dotted decimal format.
- A search list entry that lists all the domains to search when resolving hostnames. This is a space delimited list.

One domain entry and a maximum of three name server entries can exist in the system configuration database. One search entry can exist.

Flags

-add	Adds an entry to the system configuration database.
-ch *DomainName*	Changes the domain name in the system configuration database.
-chslist	Changes the search list.
-dname	Indicates that the command deals with the domain name entry.
-domain	Specifies that the operation is on the domain name. Use this flag with the **-rm** flag and the **-ls** flag.

-ipaddr *IPAddress* Indicates that the command deals with a name server entry. Use dotted decimal format for the given IP address.

-ls Shows all domain and name server entries in the configuration system database. If you use the **-ipaddr** flag, the **cfgnamesrv** command shows all name server entries. If you use the **-domain** flag, the **cfgnamesrv** command shows the domain name entry found in the database.

-mk Creates the system configuration database.

-namesrv Specifies that the **-ls** flag should print all name server entries.

-rm Deletes an entry in the system configuration database. It must be used with the **-ipaddr** *IPAddress* flag or the **-domain** flag. The **-ipaddr** flag deletes a name server entry. The **-domain** flag deletes the domain name entry.

-slist Specifies that the operation is on the search list. Use this flag with the **-rm** and **-ls** flag.

-slist *SearchList* Changes the search list in the system configuration database.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To add a domain entry with a domain name of **abc.aus.century.com**, type:

   ```
   cfgnamesrv -add -dname abc.aus.century.com
   ```
2. To add a name server entry with IP address 192.9.201.1, type:

   ```
   cfgnamesrv  -add  -ipaddr 192.9.201.1
   ```
3. To show all system configuration database entries related to domain name server information used by local resolver routines, type:

   ```
   cfgnamesrv  -ls
   ```

   The output is given in the following format:

   ```
   domain xyz.aus.century.com
   name server 192.9.201.1
   ```
4. To set the search list **abc.aus.century.com xyz.aus.century.com**, type:

   ```
   cfgnamesrv -chslist abc.aus.century.com xyz.aus.century.com
   ```

Related Information

The **cfglnagg** command, the **entstat** command, the **lsnetsvc** command, the **mktcpip** command, the **netstat** command, and the **optimizenet** command.

chdate Command

Purpose

Displays or changes the date, time or time zone.

Syntax

chdate *mmddHHMM YYyy* | *yy* **-timezone** *TZ*

chdate **-year** *YYyy* **-month** *mm* **-day** *dd* **-hour** *HH* **-minute** *MM* **-timezone** *TZ*

Description

Displays or changes the system date, time, or time zone. Changes made to the time zone will not take effect until the user logs out. For the time zone to take effect for the entire system, the system must be rebooted. All flags are optional, and the current system time information will be used if a flag was unspecified.

If no flags or arguments are specified, the **chdate** command displays the current date and time.

The *mmddHHMM YYyy* | *yy* parameters correspond to month, day, hour, minute, and optional 4 or 2 digit year.

Note: The user must have **padmin** authority to change the date and time.

Flags

-year	Sets the year to *YYyy*.
-month	Sets the month to *mm*.
-day	Sets the day to *dd*.
-hour	Sets the hour to *HH* in 24-hour format.
-minute	Sets the minute to *MM*.
-timezone	Sets the time zone (for example, CST6CDT).

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To display the current date and time, type:

   ```
   chdate
   ```
2. To change the date to Tue Oct 12 16:30:00 CDT 2004 for a system in the US Central time zone, type:

   ```
   chdate -day 12 -month 10 -year 2004 -hour 16 -minute 30
   ```

   or

   ```
   chdate 101216302004
   ```

Note: The user must have **padmin** authority to change the date and time.

chdev Command

Purpose

Changes the characteristics of a device.

Syntax

chdev **-dev** *Name* **-perm** **-attr** *Attribute=Value Attribute=Value*...

Description

The **chdev** command changes the characteristics of the device specified with the given device logical name (the **-dev** *Name* flag). Some changes may not be allowed when the device is in the **Available** state.

When the **-perm** flag is not specified, the **chdev** command applies the changes to the device and updates the database to reflect the changes. If the **-perm** flag is specified, only the database is updated to reflect the changes, and the device itself is left unchanged. This is useful in cases where a device cannot be changed because it is in use; in which case, the changes can be made to the database with the **-perm** flag, and the changes will be applied to the device when the system is restarted. Not all devices support the **-perm** flag.

Attention: To protect the Configuration database, the **chdev** command is not interruptible. To stop this command before execution is complete could result in a corrupted database.

Flags

-attr *Attribute=Value*	Specifies the device attribute value pairs used for changing specific attribute values. The *Attribute=Value* parameter can use one attribute value pair or multiple attribute value pairs for one **-attr** flag. If you use an **-attr** flag with multiple attribute value pairs, the list of pairs must be enclosed in quotes with spaces between the pairs. For example, entering **-attr** *Attribute=Value* lists one attribute value pair per flag, while entering **-attr** '*Attribute1=Value1 Attribute2=Value2*' lists more than one attribute value pair.
-dev *Name*	Specifies the device logical name, indicated by the *Name* parameter, whose characteristics are to be changed.
-perm	Changes the device's characteristics permanently without actually changing the device. This is useful for devices that cannot be made unavailable and cannot be changed while in the available state. By restarting the system, the changes will be applied to the device. Not all devices support the **-perm** flag.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To change the retention instructions of the 4mm SCSI tape drive **rmt0** so that the drive does not move the tape to the beginning, then to the end, and then back to the beginning each time a tape is inserted or the drive is powered on, type:

   ```
   chdev -dev rmt0 -attr ret=no
   ```
2. To change the SCSI ID of the available SCSI adapter **scsi0** that cannot be made unavailable or changed due to available disk drives connected to it, type:

   ```
   chdev  -dev scsi0 -attr id=6 -perm
   ```

To apply the change to the adapter, shutdown and restart the system.

Related Information

The **cfgdev** command, the **chpath** command, the **lsdev** command, the **lsmap** command, the **lspath** command, the **mkpath** command, the **mkvdev** command, the **rmdev** command, and the **rmpath** command.

chlang Command

Purpose

Changes the language settings for the system.

Syntax

chlang { **-msg** *msgtran* **-lang** *Name* **-dev** *Media* | **-ls**}

Description

The **chlang** command is a high-level command that changes the language and keyboard settings for the entire Virtual I/O Server. The user needs to log out for language changes to take effect. If the language file sets are not already installed on the system, the -**dev** flag is used to specify their location.

When **chlang** is run with the **-ls** option, all available languages are listed.

Flags

-msg *msgtran*	Modifies the NSLPATH environment variable. The *msgtran* parameter is a colon-separated list of message translations (locale names) that indicates the message translation hierarchy required for the system or user.
-dev *Media*	Specifies the device or directory containing the images to install.
-lang *Name*	Specifies the language-territory (locale name) that will become the locale setting for the **LANG** environment variable.
-ls	Lists available languages.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To change the language for the entire system to French Canadian, type:

   ```
   chlang -lang fr_CA
   ```
2. To display available languages:

   ```
   chlang -ls
   ```

Related Information

The **license** command.

IVM chled Command

Purpose

Change states for physical and virtual LEDs. This command is available only in an Integrated Virtualization Manager environment.

Syntax

To change virtual partition LEDs:

chled **-r** *sa* **-t** *virtuallpar* **-o** *Operation* { **-p** *PartitionName* | **--id** *PartitionID* } **-m** *ManagedSystem*

To change virtual system LEDs:

chled **-r** *sa* **-t** *virtualsys* **-o** *Operation* **-m** *ManagedSystem*

Description

The **chled** command changes the states of physical and virtual LEDs.

Flags

-r *ResourceType*
The type of LED resource to change. The only valid value is sa for System Attention (SA) LED.

-t *SystemAttentionType*
The type of System Attention (SA) LEDs to change.

-r sa -t virtuallpar

Change attributes for virtual partition system attention LEDs
Attributes: lpar_id, lpar_name, state
Filters: None

-r sa -t virtualsys

Change attributes for virtual system attention LEDs
Attributes: state
Filters: None

-o
The operation to perform on the LED. Valid values are:

off
 deactivate the LED
on
 activate the LED

-m *ManagedSystem*
The name of the managed system. This attribute is optional because there is only one system to manage. The name may either be the user-defined name for the managed system, or be in the form tttt-mmm*ssssssss, where tttt is the machine type, mmm is the model, and ssssssss is the serial number of the managed system.

-p *PartitionName*
The name of the partition for which the virtual partition system attention LED is to be changed.

--id *PartitionID*
The ID of the partition for with the virtual partition system attention LED is to be changed.

Exit Status

This command has a return code of zero on success.

Security

This command is not accessible by users with the ViewOnly role.

Examples

1. To deactivate the virtual system attention LED for the system, type:

   ```
   chled -r sa -t virtualsys -o off
   ```

2. To activate the virtual partition system attention LED for partition lpar3, type:

   ```
   chled -r sa -t virtuallpar -o on -p lpar3
   ```

3. To dectivate the virtual partition system attention LED for the partition with ID 3, type:

   ```
   chled -r sa -t virtuallpar -o off --id 3
   ```

Related Information

The **lsled** command.

IVM chlparutil Command

Purpose

Change settings for data collection. This command is available only in an Integrated Virtualization Manager environment.

Syntax

chlparutil **-r** *config* **-s** *SampleRate* **-m** *ManagedSystem*

Description

The **chlparutil** command changes the settings for data collection such as the sample rate at which data is collected.

Flags

-r *ResourceType*	The type of resources to change: config Change configuration settings.
-s *SampleRate*	The interval in seconds to sample utilization data. An interval of zero disables sampling. Sample rates of 0 and 30 are the only valid values.
-m *ManagedSystem*	The name of the managed system. This attribute is optional because there is only one system to manage. The name may either be the user-defined name for the managed system, or be in the form tttt-mmm*ssssssss, where tttt is the machine type, mmm is the model, and ssssssss is the serial number of the managed system.

Exit Status

This command has a return code of zero on success.

Security

This command is not accessible by users with the ViewOnly role.

Examples

1. Disable the collection of utilization data:

```
chlparutil -r config -s 0
```

Related Information

The **lslparutil** command.

chlv Command

Purpose

Changes the characteristics of a logical volume.

Syntax

To Change the Name of a logical volume:

chlv **-lv** *NewLogicalVolumeName LogicalVolume*

Description

The **chlv** command changes the characteristics of a logical volume according to the command flags. The *LogicalVolume* parameter can be a logical volume name or logical volume ID. Changing the name of a logical volume is the only supported option.

Note: A logical volume that has been assigned as a backing device can not be renamed.

Flags

-lv	Specifies logical volume

chmod Command

Purpose

Changes file modes.

Syntax

To Change File Modes Symbolically

chmod **-R** **-h** **-f** **u** **g** **o** | **a** { { **-** | **+** | **=** } **r** **w** **x** **X** **s** **t** } { *File* ... | *Directory* ... }

To Change File Modes Numerically

chmod **-R** **-h** **-f** *PermissionCode* { *File* ... | *Directory* ... }

Description

The **chmod** command modifies the mode bits and the extended access control lists (ACLs) of the specified files or directories. The mode can be defined symbolically or numerically (absolute mode).

When a symbolic link is encountered and you have not specified the **-h** flag, the **chmod** command changes the mode of the file or directory pointed to by the link and not the mode of the link itself. If you specify the **-h** flag, the **chmod** command prevents this mode change.

If you specify both the **-h** flag and the **-R** flag, the **chmod** command descends the specified directories recursively, and when a symbolic link is encountered, the mode of the file or directory pointed to by the link is not changed.

Flags

-f	Suppresses all error reporting except invalid permissions and usage statements.
-h	Suppresses a mode change for the file or directory pointed to by the encountered symbolic link. **Note:** This behavior is slightly different from the behavior of the **-h** flag on the **chgrp** and **chown** commands because mode bits cannot be set on symbolic links.
-R	Descends only directories recursively, as specified by the pattern *File...*|*Directory...*. The **-R** flag changes the file mode bits of each directory and of all files matching the specified pattern. See Example 6. When a symbolic link is encountered and the link points to a directory, the file mode bits of that directory are changed but the directory is not further traversed.

Symbolic Mode

To specify a mode in symbolic form, you must specify three sets of flags.

Note: Do not separate flags with spaces.

The first set of flags specifies who is granted or denied the specified permissions, as follows:

u	File owner.
g	Group and extended ACL entries pertaining to the file's group.
o	All others.
a	User, group, and all others. The **a** flag has the same effect as specifying the **ugo** flags together. If none of these flags are specified, the default is the **a** flag and the file creation mask (umask) is applied.

The second set of flags specifies whether the permissions are to be removed, applied, or set:

- Removes specified permissions.

+ Applies specified permissions.

= Clears the selected permission field and sets it to the permission specified. If you do not specify a permission following **=**, the **chmod** command removes all permissions from the selected field.

The third set of flags specifies the permissions that are to be removed, applied, or set:

r Read permission.

w Write permission.

x Execute permission for files; search permission for directories.

X Execute permission for files if the current (unmodified) mode bits have at least one of the user, group, or other execute bits set. The **X** flag is ignored if the *File* parameter is specified and none of the execute bits are set in the current mode bits.

Search permission for directories.

s Set-user-ID-on-execution permission if the **u** flag is specified or implied. Set-group-ID-on-execution permission if the **g** flag is specified or implied.

t For directories, indicates that only file owners can link or unlink files in the specified directory. For files, sets the **save-text** attribute.

Numeric or Absolute Mode

The **chmod** command also permits you to use octal notation for the mode. The numeric mode is the sum of one or more of the following values:

4000	Sets user ID on execution.
2000	Sets group ID on execution.
1000	Sets the link permission to directories or sets the **save-text** attribute for files.
0400	Permits read by owner.
0200	Permits write by owner.
0100	Permits execute or search by owner.
0040	Permits read by group.
0020	Permits write by group.
0010	Permits execute or search by group.
0004	Permits read by others.
0002	Permits write by others.
0001	Permits execute or search by others.

Notes:

1. Specifying the mode numerically disables any extended Access Control Lists (ACL).
2. Changing group access permissions symbolically also affects the extended ACL entries. The group entries in the ACL that are equal to the owning group of the file are denied any permission that is removed from the mode.
3. You can specify multiple symbolic modes separated with commas. Operations are performed in the order they appear from left to right.
4. You must specify the mode symbolically or use an explicit 4-character octal with a leading zero (for example, 0755) when removing the set-group-ID-on-execution permission from directories.

Security

Access Control: This program should be installed as a normal user program in the Trusted Computing Base.

Only the owner of the file or the root user can change the mode of a file.

Exit Status

This command returns the following exit values:

0 The command executed successfully and all requested changes were made.

>0 An error occurred.

Examples

1. To add a type of permission to several files:

   ```
   chmod  g+w  chap1  chap2
   ```

   This adds write permission for group members to the files `chap1` and `chap2`**.**
2. To make several permission changes at once:

   ```
   chmod  go-w+x  mydir
   ```

   This denies group members and others the permission to create or delete files in `mydir` (**go-w**) and allows group members and others to search `mydir` or use it in a path name (**go+x**). This is equivalent to the command sequence:

   ```
   chmod  g-w  mydir
   chmod  o-w  mydir
   chmod  g+x  mydir
   chmod  o+x  mydir
   ```
3. To permit only the owner to use a shell procedure as a command:

   ```
   chmod  u=rwx,go= cmd
   ```

   This gives read, write, and execute permission to the user who owns the file (**u=rwx**). It also denies the group and others the permission to access `cmd` in any way (**go=**).

   If you have permission to execute the `cmd` shell command file, then you can run it by entering:

   ```
   cmd
   ```

   Note: Depending on the **PATH** shell variable, you may need to specify the full path to the `cmd` file.
4. To use Set-ID Modes:

   ```
   chmod  ug+s  cmd
   ```

   When the `cmd` command is executed, the effective user and group IDs are set to those that own the `cmd` file. Only the effective IDs associated with the child process that runs the `cmd` command are changed. The effective IDs of the shell session remain unchanged.

   This feature allows you to permit access to restricted files. Suppose that the `cmd` program has the Set-User-ID Mode enabled and is owned by a user called `dbms`. The user `dbms` is not actually a person, but might be associated with a database management system. The user `betty` does not have permission to access any of `dbms`'s data files. However, she does have permission to execute the `cmd` command. When she does so, her effective user ID is temporarily changed to `dbms`, so that the `cmd` program can access the data files owned by the user `dbms`.

   This way the user `betty` can use the `cmd` command to access the data files, but she cannot accidentally damage them with the standard shell commands.
5. To use the absolute mode form of the **chmod** command:

   ```
   chmod  644  text
   ```

   This sets read and write permission for the owner, and it sets read-only mode for the group and others. This also removes all extended ACLs that might be associated with the file.
6. To recursively descend directories and change file and directory permissions given the tree structure:

   ./dir1/dir2/file1

   ./dir1/dir2/file2

   ./dir1/file1

   enter this command sequence:

   ```
   chmod -R 777 f*
   ```

which will change permissions on ./dir1/file1.

But given the tree structure of:

./dir1/fdir2/file1

./dir1/fdir2/file2

./dir1/file3

the command sequence:

```
chmod -R 777 f*
```

will change permissions on:

./dir1/fdir2

./dir1/fdir2/file1

./dir1/fdir2/file2

./dir1/file3

File

/usr/bin/chmod	Contains the **chmod** command .

chpath Command

Purpose

Changes the operational status of paths to a MultiPath I/O (MPIO) capable device, or changes an attribute associated with a path to an MPIO capable device.

Syntax

chpath **-dev** *Name* **-op** *OpStatus* **-pdev** *Parent* **-conn** *Connection*

chpath **-dev** *Name* **-pdev** *Parent* **-conn** *Connection* **-perm** **-attr** *Attribute=Value*...

Description

The **chpath** command either changes the operational status of paths to the specified device (the **-dev** *Name* flag) or it changes one, or more, attributes associated with a specific path to the specified device. The required syntax is slightly different depending upon the change being made.

The first syntax shown above changes the operational status of one or more paths to a specific device. The set of paths to change is obtained by taking the set of paths which match the following criteria:

- The target device matches the specified device.
- The parent device matches the specified parent (**-pdev** *Parent*), if a parent is specified.
- The connection matches the specified connection (**-conn** *Connection*), if a connection is specified.
- The path status is **PATH_AVAILABLE**

The operational status of a path refers to the usage of the path as part of MPIO path selection. The value of **enable** indicates that the path is to be used while disable indicates that the path is not to be used. It should be noted that setting a path to disable impacts future I/O, not I/O already in progress. As such, a path can be disabled, but still have outstanding I/O until such time that all of the I/O that was already in progress completes. As such, if **-op disable** is specified for a path and I/O is outstanding on the path, this fact will be displayed.

Disabling a path affects path selection at the device driver level. The **path_status** of the path is not changed in the device configuration database. The **lspath** command must be used to see current operational status of a path.

The second syntax shown above changes one or more path specific attributes associated with a particular path to a particular device. Note that multiple attributes can be changed in a single invocation of the **chpath** command; but all of the attributes must be associated with a single path. In other words, you cannot change attributes across multiple paths in a single invocation of the **chpath** command. To change attributes across multiple paths, separate invocations of **chpath** are required; one for each of the paths that are to be changed.

Flags

-attr *Attribute=Value*
Identifies the attribute to change as well as the new value for the attribute. The Attribute is the name of a path specific attribute. The Value is the value which is to replace the current value for the Attribute. The *Attribute=Value* parameter can use one attribute value pair or multiple attribute value pairs for one **-attr** flag. If you use an **-attr** flag with multiple attribute value pairs, the list of pairs must be enclosed in quotes with spaces between the pairs. For example, entering **-attr** *Attribute=Value* lists one attribute value pair per flag, while entering **-attr** '*Attribute1=Value1 Attribute2=Value2*' lists more than one attribute value pair.

-dev *Name*
Specifies the logical device name of the target device for the path(s) affected by the change. This flag is required in all cases.

-pdev *Parent*
Indicates the logical device name of the parent device to use in qualifying the paths to be changed. This flag is required when changing attributes, but is optional when change operational status.

-perm

Changes the path's characteristics without actually changing the path. The change takes affect on the path the next time the path is unconfigured and then configured (possibly on the next boot).

-conn *Connection*
Indicates the connection information to use in qualifying the paths to be changed. This flag is optional when changing operational status. When changing attributes, it is optional if the device has only one path to the indicated parent. If there are multiple paths from the parent to the device, then this flag is required to identify the specific path being changed.

-op *OpStatus*
Indicates the operational status to which the indicated paths should be changed. The operational status of a path is maintained at the device driver level. It determines if the path will be considered when performing path selection.The allowable values for this flag are:

enable
Mark the operational status as enabled for MPIO path selection. A path with this status will be considered for use when performing path selection. Note that enabling a path is the only way to recover a path from a failed condition.

disable
Mark the operational status as disabled for MPIO path selection. A path with this status will not be considered for use when performing path selection.

This flag is required when changing operational status. When used in conjunction with the **-attr** *Attribute=Value* flag, a usage error is generated.

Exit Status

SeeVirtual I/O Server command exit status.

Examples

1. To disable the paths between **scsi0** and the **hdisk1** disk device, enter:

   ```
   chpath -dev hdisk1 -pdev scsi0 -op disable
   ```

   The system displays a message similar to one of the following:

   ```
   paths disabled
   ```

   or

   ```
   some paths disabled
   ```

   The first message indicates that all **PATH_AVAILABLE** paths from **scsi0** to **hdisk1** have been successfully enabled. The second message indicates that only some of the **PATH_AVAILABLE** paths from **scsi0** to **hdisk1** have been successfully disabled.

Related Information

The **cfgdev** command, the **chdev** command, the **lsdev** command, the **lsmap** command, the **lspath** command, the **mkpath** command, the **mkvdev** command, the **rmdev** command, and the **rmpath** command.

chsp Command

Purpose

Change the characteristics of a storage pool.

Syntax

Add physical volume to a storage pool:

chsp **-add** **-f** **-sp** *StoragePool PhysicalVolume*...

Remove a physical volume from a storage pool :

chsp **-rm** **-f** **-sp** *StoragePool PhysicalVolume*...

Set storage pool as the default :

chsp **-default** *StoragePool*

Description

The **chsp** command adds and removes physical volumes represented by the *PhysicalVolume* parameter from a storage pool. Before adding physical volume, the **chsp** command checks to verify that they are not already in another volume group or storage pool. If the system detects a description area from a volume group or storage pool the command will fail. If the **-f** flag is specified, the physical volume is added even if it contains a description area, unless it is a member of another storage pool or volume group.

Before removing physical volumes from the storage pool, the **chsp** command prompts the user to verify each logical volume should be removed. If the user specifies the **-f** flag, the logical volumes are removed with out prompting for input. If all physical volumes in a storage pool are removed, the storage pool is also removed.

If the **-sp** flag is not specified, the default storage pool is assumed.

If the **-default** flag is included, the storage pool specified will become the default storage pool for all users. If no default has been set, the rootvg storage pool will be the default. Only the padmin user can set the default storage pool.

Files

-add	Adds the given physical volume into specified storage pool. If a storage pool is not specified the physical volume is added to the default pool.
-default	Sets the specified storage pool as the system default pool. Only the padmin user can set the default storage pool.
-f	When combined with the **-add** flag, forces the physical volume to be added to the specified storage pool unless the physical volume is part of another storage pool or volume group in the Device Configuration Database or a volume group that is active. When combined with the **-rm** flag, forces the removal of all logical volumes on the physical volume before removal from the storage pool.
-rm	Remove the specified physical volume from specified storage pool. If a storage pool is not specified, the physical volume is removed from the default pool. The user will be prompted to confirm the removal of any logical volumes on the physical volume.
-sp *StoragePool*	Specifies the storage pool to be changed. If the specified name does not start with "`sp_`, the command will automatically prefix it with `sp_`.

Exit Status

23	Specified storage pool is not valid

Examples

1. To add physical volume hdisk3 to the default storage pool, type:

   ```
   chsp -add hdisk3
   ```
2. To remove physical volume hdisk2 from clstorage storage pool, type:

   ```
   chsp -rm -sp clstorage hdisk2
   ```

IVM chsvcevent Command

Purpose

Changes an existing serviceable event. This command is available only in an Integrated Virtualization Manager environment.

Syntax

To close an existing serviceable event:

chsvcevent -o *close* **-p** *ProblemNumber* **-n** *Name* **-c** *CommentText* **-m** *ManagedSystem*

Description

The **chsvcevent** command closes an existing serviceable event.

Flags

-o Operation	The operation to be performed. The only valid value is `close`.
-p *ProblemNumber*	The problem number (problem_number) for the event, as displayed by the lssvcevents command.
-n	A free form name string identifying the person who is closing the event.
-c	A free form text comment about why the event is being changed.
-m *ManagedSystem*	The name of the managed system. This attribute is optional because there is only one system to manage. The name may either be the user-defined name for the managed system, or be in the form tttt-mmm*ssssssss, where tttt is the machine type, mmm is the model, and ssssssss is the serial number of the managed system.

Exit Status

This command has a return code of zero on success.

Security

This command is not accessible by users with the ViewOnly role.

Examples

1. To close a serviceable event, type:

```
chsvcevent -o close -p 6013EFFF-205F3F22-4CC992E5-F8B6270-7540D8A3 -m 9111-520*XXXXXXX -n My Name -c Clos
```

Related Information

The **lssvcevents** command, and the **mksvcevent** command.

IVM chsyscfg Command

Purpose

Changes attributes of partitions, partition profiles, or the managed system. This command is available only in an Integrated Virtualization Manager environment.

Syntax

To change system attributes:

chsyscfg -r sys { **-f** *ConfigurationFile* | **-i** *ConfigurationData* } **-m** *ManagedSystem*

To change partition attributes

chsyscfg -r lpar { **-f** *ConfigurationFile* | **-i** *ConfigurationData* } **-m** *ManagedSystem*

To change partition profile attributes

chsyscfg -r prof { **-f** *ConfigurationFile* | **-i** *ConfigurationData* } **-m** *ManagedSystem*

Description

The **chsyscfg** command changes attributes of partitions, partition profiles, or the managed system.

Flags

-r *ResourceType*
: The type of resources to change:

sys
: Managed system resources

lpar
: Logical partition resources

prof
: Logical partition profile resources

-m *ManagedSystem*
: The name of the managed system. This attribute is optional because there is only one system to manage. The name may either be the user-defined name for the managed system, or be in the form tttt-mmm*ssssssss, where tttt is the machine type, mmm is the model, and ssssssss is the serial number of the managed system.

-f *ConfigurationFile*
: The name of the file containing the configuration data needed to change the resources. The configuration data consists of attribute name and value pairs, which are in comma separated value (CSV) format. These attribute name and value pairs form a configuration record. A line feed marks the end of a configuration record. The file must contain one configuration record for each resource to be changed, and each configuration record must be for the same resource type. If the resource type is the managed system, then the file must contain only one configuration record.

The format of a configuration record is as follows:

```
attribute-name=value,attribute-name=value,...<LF>
```

Note that certain attributes accept a comma separated list of values, as follows:

```
"attribute-name=value,value,...",...<LF>
```

When a list of values is specified, the attribute name/value pair must be enclosed in double quotes. Depending on the shell being used, nested double quote characters may need to be preceded by an escape character, which is usually a \ (back slash) character.

If '+=' is used in the attribute name/value pair instead of '=', then the specified value is added to the existing value for the attribute if the attribute is numerical. If the attribute is a list, then the specified values are added to the existing list.

If '-=' is used in the attribute name/value pair instead of '=', then the specified value is subtracted from the existing value for the attribute if the attribute is numerical. If the attribute is a list, then the specified values are deleted from the existing list.

Attribute names for partitions

name | lpar_id
  name or ID of the partition to change (required)
new_name
  new name for the partition.
shared_proc_pool_util_auth
  Valid values are:
  0 - do not allow authority
  1 - allow authority
work_group_id
  Valid values are:
  none - do not participate in the workload management group
  1 - participate in the workload management group

Attribute names for partition profiles

name | lpar_name | lpar_id
  Name or ID of the partition to change (required) **Note:** This command uses the profile name and lpar_name interchangeably because this environment does not support multiple profiles per partition. When using this command on the Hardware Management Console, you must specify the profile name and partition name or ID because it supports multiple profiles per partition.
new_name
  New name for the partition.
min_mem
  minimum memory in megabytes
desired_mem
  assigned memory in megabytes

max_mem
  maximum memory in megabytes
proc_mode
  Valid values are:
  ded: dedicated processor mode
  shared: shared processor mode
min_procs
  Minimum processors. In shared processing mode, this refers to virtual processors.
desired_procs
  Assigned processors. In shared processing mode, this refers to virtual processors.
max_procs
  Maximum processors. In shared processing mode, this refers to virtual processors.
min_proc_units
  Minimum shared processing units.
desired_proc_units
  Assigned shared processing units.
max_proc_units
  Maximum shared processing units.
sharing_mode
  Valid values are:
  keep_idle_procs: Valid with dedicated processor mode
  share_idle_procs: Valid with dedicated processor mode
  cap: Capped mode. Valid with shared processor mode
  uncap: Uncapped mode. Valid with shared processor mode
uncap_weight
  A weighted average of processing priority when in uncapped sharing mode. The smaller the value, the lower the weight. Valid values are: 0 - 255
auto_start
  Valid values are:
  0 - do not automatically start with system power on
  1 - automatically start with system power on

boot_mode

Partition power on mode. Valid values are:
norm - normal
dd - diagnostic with default boot list
ds - diagnostic with stored boot list
of - Open Firmware OK prompt
sms - System Management Services

max_virtual_slots

Maximum number of virtual I/O adapter slots

virtual_eth_adapters

Comma separated list of virtual ethernet adapters, with each adapter having the following format:
`slot_number/is_ieee/port_vlan_id/additional_vlan_ids/is_trunk/is_re`
All 5 '/' characters must be present, but optional values may be omitted. Optional values are `is_ieee`, `additional_vlan_ids`, `is_required` and `is_trunk`.

Valid values for `is_ieee`, `is_trunk`, and `is_required`"

0 - no
1 - yes

For example, `4/0/2//0/0` specifies a virtual Ethernet adapter with a virtual slot number of 4, is not IEEE 802.1Q enabled, has a port virtual LAN ID of 2, no additional virtual LAN IDs, it is not a trunk adapter, and is not required.

Attribute names for the managed system

new_name

new name for the managed system

pend_configured_max_lpars

The maximum number of logical partitions which can be created after restarting the managed system.

lpar_comm_ipaddr

The IP address through which client partitions will communicate with the management partition. This is used primarily for Dynamic Logical Partitioning (DLPAR). It is defaulted to the first IP address configured on your system, but can be manually set if desired. If you set this manually, and then change your systems IP address, you must update this value.

lpar_comm_default

Returns the lpar_comm_ipaddr to using the default IP address configured on the system as reported by lstcpip -interfaces. Valid values:
◊ 1: Default IP address will be used.

-i *ConfigurationData*

This option allows you to enter configuration data on the command line, instead of using a file. Data entered on the command line must follow the same format as data in a file, and must be enclosed in double quotes.

When this option is used, only a single resource can be changed.

The **-i** and the **-f** options are mutually exclusive.

Exit Status

This command has a return code of zero on success.

Security

This command is not accessible by users with the ViewOnly role.

Examples

1. To change the user defined name for the managed system, type:

```
chsyscfg -r sys -i "new_name=sys1"
```

2. To change partitions using the configuration data in the file **/tmp/lparfile**, type:

```
chsyscfg -r lpar -f /tmp/lparfile
```

3. To reduce a partition profile's assigned and minimum memory by 256 MB, type:

```
chsyscfg -r prof -i "lpar_name=partition3,min_mem-=256,desired_mem-=256"
```

Related Information

The **lssyscfg** command, the **mksyscfg** command, and the **rmsyscfg** command.

IVM chsysstate Command

Purpose

Changes the state of a partition. This command is operable only in the Integrated Virtualization Manager environment.

Syntax

To activate a partition:

chsysstate -r *lpar* **-o** `on` { **-n** *Name* | **--id** *PartitionID* } **-k** *KeylockPosition* **-b** *BootMode* **-m** *ManagedSystem*

To perform a partition shutdown using the shutdown command on the client operating system:

chsysstate -r *lpar* **-o** `osshutdown` { **-n** *Name* | **--id** *PartitionID* } **-m** *ManagedSystem*

To perform a delayed partition shut down (white button shut down):

chsysstate -r *lpar* **-o** `shutdown` { **-n** *Name* | **--id** *PartitionID* } **-m** *ManagedSystem*

To perform an immediate partition shutdown (operator panel function 8):

chsysstate -r *lpar* **-o** `shutdown` **--immed** { **-n** *Name* | **--id** *PartitionID* } **-m** *ManagedSystem*

To perform an immediate restart of a partition (operator panel function 3):

chsysstate -r *lpar* **-o** `shutdown` **--immed --restart** { **-n** *Name* | **--id** *PartitionID* } **-m** *ManagedSystem*

To perform a restart of a partition after initiating a dump (operator panel function 22):

chsysstate -r *lpar* **-o** `dumprestart` { **-n** *Name* | **--id** *PartitionID* } **-m** *ManagedSystem*

To change the keylock position for a partition:

chsysstate -r *lpar* **-o** `chkey` **-k** `KeylockPosition` { **-n** *Name* | **--id** *PartitionID* } **-m** *ManagedSystem*

Description

The **chsysstate** command changes the state of a partition. To avoid data loss, it is recommended to use the shutdown facilities provided by the operating system in the partition.

Flags

-r *ResourceType*

The type of resources to change: lpar Logical partition resources

-m *ManagedSystem*

The name of the managed system. This attribute is optional because there is only one system to manage. The name may either be the user-defined name for the managed system, or be in the form tttt-mmm*sssssss, where tttt is the machine type, mmm is the model, and sssssss is the serial number of the managed system.

-o *Operation*

The type of operation to perform:

- `chkey`: Change the keylock position
- `on`: Power on
- `dumprestart`: restart after initiating a dump
- `osshutdown`: shut down using the client partition's shutdown command. This is the safest shutdown option, so should be used if available (RMC state must be active).
- `shutdown`: shut down

-b *BootMode*

Override the current power on mode setting. Valid values are:

- `norm`: normal
- `dd`: diagnostic with default boot list
- `ds`: diagnostic with stored boot list
- `of`: Open Firmware OK prompt
- `sms`: System Management Services

-k *KeylockPosition*
Keylock position. Valid values are:

- `norm`: normal keylock
- `manual`: manual keylock

--immed
Force the state change immediately.

--restart
Restart the partition. This flag is valid only if the **--immed** flag is also specified.

Exit Status

This command has a return code of zero on success.

Security

This command is not accessible by users with the ViewOnly role.

Examples

1. To power on the partition with an ID of 2 and set the boot mode to System Management Services, type:

   ```
   chsysstate -r lpar -o on --id 2 -b sms
   ```
2. To shut down the partition with an ID of 3, type:

   ```
   chsysstate -r lpar -o shutdown --id 3
   ```
3. To immediately restart the partition with an ID of 3 using the client operating system's shutdown command, type:

   ```
   chsysstate -r lpar -o osshutdown --restart --immed --id 3
   ```

Related Information

The **lssyscfg** command, the **mksyscfg** command, the **rmsyscfg** command, and the **mkvt** command.

chtcpip Command

Purpose

Changes the Virtual I/O Server TCP/IP settings and parameters.

Syntax

chtcpip -interface *Interface* **-inetaddr** *Address* **-netmask** *SubnetMask*

chtcpip -interface *Interface* **-gateway -add** *New_gateway_address* **-remove** *Old_gateway_address*

Description

The chtcpip command changes the TCP/IP settings and configuration on the Virtual I/O Server.

Flags

-add *New_Gateway_Address* New Default Gateway address to be added.

-inetaddr *Address* Changes the IP address of the host. Specify the address in dotted decimal notation.

-interface *Interface* Specifies a particular network interface, for example `en0`.

-gateway *Gateway* Changes the gateway address for a static route. Specify both the current address and new address in dotted decimal notation

-netmask *SubnetMask* Specifies the subnet mask the gateway. Use to determine the appropriate subnetwork for routing.

-remove *Old_Gateway_Address* Old Default Gateway address to removed.

Exit Status

0	The command completed successfully
>0	An error occurred.

Examples

1. To changes the current network address and mask to the new settings, type:

   ```
   chtcpip -interface en0 -inetaddr 9.1.1.1 -netmask 255.255.255.0
   ```
2. To changes the default gateway from 9.1.2.3 to 9.2.3.4, type:

   ```
   chtcpip -interface en0 -gateway -add 9.2.3.4 -remove 9.1.2.3
   ```

   Note: If more than one default gateway is defined, an error will occur. Only one default gateway can be defined by the **mktcpip** command.

Related Information

The **topas** command, and the **mktcpip** command.

chuser Command

Purpose

Changes user attributes.

Syntax

chuser -attr *Attribute=Value* ... *Name*

Description

The **chuser** command changes attributes for the user identified by the Name parameter. To change an attribute, specify the attribute name and the new value with the **-attr** *Attribute=Value* parameter.

Attributes

If you have the proper authority, you can set the following user attributes:

account_locked
Indicates if the user account is locked. Possible values include:

true
The user's account is locked. The values **yes**, **true**, and **always** are equivalent. The user is denied access to the system.

false
The user's account is not locked. The values **no**, **false**, and **never** are equivalent. The user is allowed access to the system. This is the default value.

expires
Identifies the expiration date of the account. The Value parameter is a 10-character string in the *MMDDhhmmyy* form, where *MM* = month, *DD* = day, *hh* = hour, *mm* = minute, and *yy* = last 2 digits of the years 1939 through 2038. All characters are numeric. If the *Value* parameter is 0, the account does not expire. The default is 0.

histexpire
Defines the period of time (in weeks) that a user cannot reuse a password. The value is a decimal integer string. The default is 0, indicating that no time limit is set.

histsize
Defines the number of previous passwords a user cannot reuse. The value is a decimal integer string. The default is 0. Only an administrative user can change this attribute.

loginretries
Defines the number of unsuccessful login attempts allowed after the last successful login before the system locks the account. The value is a decimal integer string. A zero or negative value indicates that no limit exists. Once the user's account is locked, the user will not be able to log in until the prime administrator resets the user's **account_locked** attribute.

maxage
Defines the maximum age (in weeks) of a password. The password must be changed by this time. The value is a decimal integer string. The default is a value of 0, indicating no maximum age. Range: 0 to 52

maxexpired
Defines the maximum time (in weeks) beyond the **maxage** value that a user can change an expired password. The value is a decimal integer string. The default is -1, indicating restriction is set. If the **maxexpired** attribute is 0, the password expires when the **maxage** value is met. If the **maxage** attribute is 0, the **maxexpired** attribute is ignored. Range: 0 to 52

maxrepeats
Defines the maximum number of times a character can be repeated in a new password. Since a value of 0 is meaningless, the default value of 8 indicates that there is no maximum number. The value is a decimal integer string. Range: 0 to 8

minlen
Defines the minimum length of a password. The value is a decimal integer string. The default is a value of 0, indicating no minimum length. The maximum value allowed is 8. This attribute is determined by **minlen** and/or **'minalpha + minother'**, whichever is greater. **'minalpha + minother'** should never be greater than 8. If **'minalpha + minother'** is greater than 8, then the effective value for **minother** is reduced to **'8 - minalpha'**.

minother
Defines the minimum number of non-alphabetic characters that must be in a new password. The value is a decimal integer string. The default is a value of 0, indicating no minimum

number. Range: 0 to 8

pgrp
Defines the Primary Group and Groups membership. The only valid entries are `staff` and `view`. If this attribute is not defined, the default is `staff` is used.

pwdwarntime
Defines the number of days before the system issues a warning that a password change is required. The value is a decimal integer string. A zero or negative value indicates that no message is issued. The value must be less than the difference of the **maxage** and **minage** attributes. Values greater than this difference are ignored and a message is issued when the **minage** value is reached.

Exit Status

See Virtual I/O Server command exit status.

Security

This command can only be executed by the prime administrator (**padmin**) user.

Examples

1. To change the expiration date for the **davis** user account to 8 a.m., 1 May, 1995, type:

```
chuser -attr expires=0501080095 davis
```

Related Information

The **lsuser** command, the **mkuser** command, the **rmuser** command, and the **passwd** command.

chvg Command

Purpose

Sets the characteristics of a volume group.

Syntax

chvg **-unlock** **-suspend** | **-resume** *VolumeGroup*

Description

The **chvg** command changes the characteristics of a volume group.

Flags

-resume Resumes normal I/O operations for a volume group.

-suspend Drains I/O's for this volume group and suspends future I/O's.

-unlock Unlocks the volume group. This option is provided if the volume group is left in a locked state by abnormal termination of another LVM operation (such as the command core dumping, or the system crashing). **Note:** Before using the **-unlock** flag, make sure that the volume group is not being used by another LVM command.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To suspend the volume group **vg03**, type:

   ```
   chvg -suspend vg03
   ```
2. To resume the volume group **vg03**, type:

   ```
   chvg -resume vg03
   ```

Related Information

The **mkvg** command, the **lsvg** command, the **extendvg** command, the **reducevg** command, the **mirrorios** command, the **unmirrorios** command, the **activatevg** command, the **deactivatevg** command, the **importvg** command, the **exportvg** command, and the **syncvg** command.

clear Command

Purpose

Clears the terminal screen.

Syntax

clear

Description

The **clear** command clears your screen, if possible. The **clear** command first checks the **TERM** environment variable for the terminal type. Next, the **/usr/share/lib/terminfo** directory, which contains terminal definition files, is checked to determine how to clear the screen. If the **TERM** environment variable is not set, the **clear** command exits without taking any action.

Examples

To clear your screen, enter:

```
clear
```

Files

/usr/share/lib/terminfo	Contains terminal information database.

cp Command

Purpose

Copies files.

Syntax

To Copy a File to another File

cp -E{**force**|**ignore**|**warn**} **-f** **-h** **-i** **-p** **-I** **-U** **-** *SourceFile TargetFile*

To Copy a File to a Directory

cp -E{**force**|**ignore**|**warn**} **-f** **-h** **-i** **-p** **-r** | **-R** **-H** | **-L** | **-P** **-I** **-U** **-** *SourceFile ... TargetDirectory*

To Copy a Directory to a Directory

cp -E{**force**|**ignore**|**warn**} **-f** **-h** **-i** **-p** { **-r** | **-R** } **-H** | **-L** | **-P** **-I** **-U** **-** *SourceDirectory ... TargetDirectory*

Description

The **cp** command copies the source file specified by the *SourceFile* parameter to the destination file specified by the *TargetFile* parameter. If the target file exists, **cp** overwrites the contents, but the mode, owner, and group associated with it are not changed. The last access time of the *SourceFile* and the last modification time of the *TargetFile* are set to the time the copy was done. If the *TargetFile* does not exist, **cp** creates a new file named *TargetFile* that has the same mode as the source file except that the sticky bit is not set unless it was done by a superuser; the owner and group of the *TargetFile* is that of the user. When the *TargetFile* is a link to another file, **cp** overwrites the destination link with the content of the source file; the links from the *TargetFile* remains. Also, the **cp** command can copy the source files specified by the *SourceFile* parameter (or directories named by the *SourceDirectory* parameter) to the directory specified by the *TargetDirectory* parameter.

Note: If one of the source parameters is a directory, you need to specify one of the **-r** or **-R** flags.

If any directories are created by the **cp** command during the copying process, the newly created directory will have the same mode as the corresponding source directory.

You can also copy special device files. The preferred option for accomplishing this is the **-R** flag. Specifying **-R** causes the special files to be re-created under the new path name. Specifying the **-r** flag causes the **cp** command to attempt to copy the special file to a regular file.

Flags

-E
: The **-E** option requires one of the following arguments. If you omit the **-E** option, **warn** is the default behavior.

force
: Fails the **cp** operation on a file if the fixed extent size or space reservation of the file cannot be preserved.

ignore
: Ignores any errors in preserving extent attributes.

warn
: Issues a warning if the space reservation or the fixed extent size of the file cannot be preserved.

-f
: Specifies removal of the target file if it cannot be opened for write operations. The removal precedes any

copying performed by the **cp** command.

-h Forces the **cp** command to copy symbolic links. The default is to follow symbolic links, that is, to copy files to which symbolic links point.

-H Take actions based on the type and contents of the file referenced by any symbolic link specified as a *SourceFile* operand.

-i Prompts you with the name of a file to be overwritten. This occurs if the *TargetDirectory* or *TargetFile* parameter contains a file with the same name as a file specified in the *SourceFile* or *SourceDirectory* parameter. If you enter `y` or the locale's equivalent of `y`, the **cp** command continues. Any other answer prevents the **cp** command from overwriting the file.

-I Suppresses the warning message during ACL conversion.

-L Take actions based on the type and contents of the file referenced by any symbolic link specified as a *SourceFile* operand or any symbolic links encountered during traversal of a file hierarchy.

-p Duplicates the following characteristics of each SourceFile/SourceDirectory in the corresponding *TargetFile* and/or *TargetDirectory*:

- The time of the last data modification and the time of the last access. If this duplication fails for any reason, the **cp** command will write a diagnostic message to standard error.
- The user ID and group ID. If this duplication fails for any reason, the **cp** command may write a diagnostic message to standard error.
- The file permission bits and the S_ISUID and S_ISGID bits. If this duplication fails for any reason, the **cp** command will write a diagnostic message to standard error.

If the user ID or group ID cannot be duplicated, the file permission bits S_ISUID and S_ISGID are cleared.

In order to preserve the owner ID and group ID, permission modes, modification and access times, user must have the appropriate file access permissions (user should be a superuser or have the same owner ID as the destination file)

The target file will not be deleted if these characteristics cannot be preserved.

Access control lists (ACLs) associated with the *SourceFile* are preserved if the target filesystem supports the same. If the source file contains NFS4 ACL and the target filesystem does not support NFS4 ACL, the NFS4 ACL is converted to AIXC.

When ACL conversion succeeds, a warning message is printed out the stderr.

-P Take actions on any symbolic link specified as a *SourceFile* operand or any symbolic link encountered during traversal of a file hierarchy.

-r Copies file hierarchies under the file or directory specified by the *SourceFile* or *SourceDirectory* parameter (recursive copy). The **-r** flag processes special files in the same manner as regular files.

-R Copies file hierarchies under the regular files and directories from the directory specified by the *SourceFile* or *SourceDirectory* parameter to the directory specified by the *TargetDirectory* parameter. Special file types, such as first-in, first-out (FIFO) files and block and character device files, are re-created instead of copied. Symbolic links are followed unless the **-h** flag is specified. (The **-R** flag is preferred to the **-r** flag.)

If none of the **-H**, **-L**, or **-P** options were specified, it is unspecified which of those options will be used as the default. Consider the following:

- If the **-H** option was specified, the **cp** command will take action based on the type and contents of the file referenced by any symbolic link specified as a *SourceFile* operand.
- If the **-L** option was specified, the **cp** command will take action based on the type and contents of the file referenced by any symbolic link specified as a *SourceFile* operand or any symbolic links encountered during traversal of a file hierarchy.
- If the **-P** option was specified, the **cp** command will copy any symbolic link specified as a *SourceFile* operand and any symbolic links encountered during traversal of a file hierarchy and will not follow any symbolic links.

-U Copies Extended Attributes (EA), Access Control Lists (ACL) in the *SourceFile* to the *TargetFile*. If the EA is not supported on the target filesystem then it is ignored. If the source ACL type is not supported on the target filesystem then it is converted to the compatible ACL type supported by the target filesystem.

-- Indicates that parameters following the **--** (dash, dash) flag are to be interpreted as file names. This null flag allows the specification of file names that start with a **-** (minus sign).

Exit Status

This command returns the following exit values:

0	All files were copied successfully.
>0	An error occurred.

Examples

1. To make a copy of a file in the current directory, enter:

   ```
   cp prog.c prog.bak
   ```

   This copies `prog.c` to `prog.bak`. If the `prog.bak` file does not already exist, the **cp** command creates it. If it does exist, the **cp** command replaces it with a copy of the `prog.c` file.
2. To copy a file in your current directory into another directory, enter:

   ```
   cp jones /home/nick/clients
   ```

   This copies the `jones` file to `/home/nick/clients/jones`.
3. To copy a file to a new file and preserve the modification date, time, and access control list associated with the source file, enter: `cp  -p smith smith.jr`
   This copies the `smith` file to the `smith.jr` file. Instead of creating the file with the current date and time stamp, the system gives the `smith.jr` file the same date and time as the `smith` file. The `smith.jr` file also inherits the `smith` file's access control protection.
4. To copy all the files in a directory to a new directory, enter:

   ```
   cp /home/janet/clients/* /home/nick/customers
   ```

   This copies only the files in the `clients` directory to the `customers` directory.
5. To copy a directory, including all its files and subdirectories, to another directory, enter: `cp -R /home/nick/clients /home/nick/customers`
   Note: A directory cannot be copied into itself. This copies the `clients` directory, including all its files, subdirectories, and the files in those subdirectories, to the `customers/clients` directory.
6. To copy a specific set of files to another directory, enter:

   ```
   cp jones lewis smith /home/nick/clients
   ```

   This copies the `jones`, `lewis`, and `smith` files in your current working directory to the `/home/nick/clients` directory.
7. To use pattern-matching characters to copy files, enter:

   ```
   cp programs/*.c .
   ```

   This copies the files in the `programs` directory that end with `.c` to the current directory, signified by the single `.` (dot). You must type a space between the `c` and the final dot.
8. To copy a file to a new file and preserve the ACL and EA associated with the source file, enter:

   ```
   cp -U smith smith.jr
   ```

Files

/usr/bin/cp
Contains the **cp** command.

Related Information

The **mv** command.

cplv Command

Purpose

Copies the contents of a logical volume to a new logical volume.

Syntax

To Copy to a New Logical Volume

cplv **-vg** *VolumeGroup* **-lv** *NewLogicalVolume* | **-prefix** *Prefix* *SourceLogicalVolume*

To Copy to an Existing Logical Volume

cplv **-f** *SourceLogicalVolume* *DestinationLogicalVolume*

Description

Attention: Do not copy from a larger logical volume containing data to a smaller one. Doing so results in a corrupted file system because some data is not copied.

The **cplv** command copies the contents of *SourceLogicalVolume* to a new or existing logical volume. The *SourceLogicalVolume* parameter can be a logical volume name or a logical volume ID. The **cplv** command creates a new logical volume with a system-generated name by using the default syntax. The system-generated name is displayed.

Note: The **cplv** command can not copy logical volumes which are in the open state, including logical volumes that are being used as backing devices for virtual storage.

Flags

-f	Copies to an existing logical volume without requesting user confirmation.
-lv *NewLogicalVolume*	Specifies the name to use, in place of a system-generated name, for the new logical volume. Logical volume names must be unique systemwide names, and can range from 1 to 15 characters.
-prefix *Prefix*	Specifies a prefix to use in building a system-generated name for the new logical volume. The prefix must be less than or equal to 13 characters. A name cannot be a name already used by another device.
-vg *VolumeGroup*	Specifies the volume group where the new logical volume resides. If this is not specified, the new logical volume resides in the same volume group as the *SourceLogicalVolume*.

Examples

1. To copy the contents of logical volume fslv03 to a new logical volume, type:

   ```
   cplv fslv03
   ```

   The new logical volume is created, placed in the same volume group as fslv03, and named by the system.
2. To copy the contents of logical volume fslv03 to a new logical volume in volume group vg02, type:

   ```
   cplv  -vg vg02 fslv03
   ```

   The new logical volume is created, named, and added to volume group vg02.
3. To copy the contents of logical volume lv02 to a smaller, existing logical volume, lvtest, without requiring user confirmation, type:

```
cplv -f lv02 lvtest
```

crontab Command

Purpose

Submits, edits, lists, or removes cron jobs.

Syntax

crontab **-e** *UserName* | **-l** *UserName* | **-r** *UserName* | **-v** *UserName* | *File*

Description

The **crontab** command submits, edits, lists, or removes cron jobs. A cron job is a command run by the **cron** daemon at regularly scheduled intervals. To submit a cron job, specify the **crontab** command with the **-e** flag. The **crontab** command invokes an editing session that allows you to create a **crontab** file. You create entries for each cron job in this file. Each entry must be in a form acceptable to the **cron** daemon. For information on creating entries, see The crontab File Entry Format.

When you finish creating entries and exit the file, the **crontab** command copies it into the **/var/spool/cron/crontabs** directory and places it in a file named for your current user name. If a file with your name already exists in the **crontabs** directory, the **crontab** command overwrites it.

Alternatively, you can create a **crontab** file by specifying the *File* parameter. If the file exists, it must be in the format the **cron** daemon expects. If the file does not exist, the **crontab** command invokes the editor. If the **EDITOR** environment variable exists, the command invokes the editor it specifies. Otherwise, the **crontab** command uses the **vi** editor.

To list the contents of your **crontab** file, specify the **crontab** command with the **-l** flag. To remove an existing file, use the **-r** flag.

The optional *UserName* parameter can be used by the owner of the **crontab** file or by the root user to edit, list, remove, or verify the status of the cron jobs for the specified user. If the *UserName* is invalid, an error message is generated and the program exits.

If the optional *UserName* parameter is not specified, the **crontab** flags are available for the root user and the current user.

Security

Only the root user or the owner of the **crontab** file can use *UserName* following the **-e**, **-l**, **-r**, and **-v** flags to edit, list, remove, or verify the **crontab** file of the specified user.

The cron Daemon

The **cron** daemon runs commands according to the **crontab** file entries. Unless you redirect the output of a cron job to standard output or error, the **cron** daemon mails you any command output or errors. If you specify a cron job incorrectly in your **crontab** file, the **cron** daemon does not run the job.

The **cron** daemon examines **crontab** files only when the **cron** daemon is initialized. When you make changes to your **crontab** file using the **crontab** command, a message indicating the change is sent to the **cron** daemon. This eliminates the overhead of checking for new or changed files at regularly scheduled intervals.

Controls on Using the crontab Command

The **/var/adm/cron/cron.allow** and **/var/adm/cron/cron.deny** files control which users can use the **crontab** command. A root user can create, edit, or delete these files. Entries in these files are user login names with one name to a line. If your login ID is associated with more than one login name, the **crontab** command uses the first login name that is in the **/etc/passwd** file, regardless of which login name you might actually be using. Also, to allow users to start **cron** jobs, the daemon attribute in the **/etc/security/user** file should be set to TRUE , using the **chuser** command.

The following is an example of an **cron.allow** file:

```
root
nick
dee
sarah
```

If the **cron.allow** file exists, only users whose login names appear in it can use the **crontab** command. The root user's log name must appear in the **cron.allow** file if the file exists. A system administrator can explicitly stop a user from using the **crontab** command by listing the user's login name in the **cron.deny** file. If only the **cron.deny** file exists, any user whose name does not appear in the file can use the **crontab** command.

A user cannot use the **crontab** command if one of the following is true:

- The **cron.allow** file and the **cron.deny** file do not exist (allows root user only).
- The **cron.allow** file exists but the user's login name is not listed in it.
- The **cron.deny** file exists and the user's login name is listed in it.

If neither the **cron.allow** nor the **cron.deny** file exists, only someone with root user authority can submit a job with the **crontab** command.

The crontab File Entry Format

A **crontab** file contains entries for each cron job. Entries are separated by newline characters. Each **crontab** file entry contains six fields separated by spaces or tabs in the following form:

```
minute  hour  day_of_month  month  weekday  command
```

These fields accept the following values:

minute	0 through 59
hour	0 through 23
day_of_month	1 through 31
month	1 through 12
weekday	0 through 6 for Sunday through Saturday
command	a shell command

You must specify a value for each field. Except for the *command* field, these fields can contain the following:

- A number in the specified range. To run a command in May, specify 5 in the **month** field.
- Two numbers separated by a dash to indicate an inclusive range. To run a **cron** job on Tuesday through Friday, place 2-5 in the **weekday** field.
- A list of numbers separated by commas. To run a command on the first and last day of January, you would specify 1,31 in the **day_of_month** field.
- An * (asterisk), meaning all allowed values. To run a job every hour, specify an asterisk in the hour field.

  Note: Any character preceded by a backslash (including the %) causes that character to be treated literally. The specification of days may be made by two fields (day of the month and day of the week). If you specify both as a list of elements, both are adhered to. For example, the following entry:

```
0 0 1,15 * 1 command
```

would run command on the first and fifteenth days of each month, as well as every Monday. To specify days by only one field, the other field should contain an * .

Specifying Commands

The **cron** daemon runs the command named in the sixth field at the selected date and time. If you include a % (percent sign) in the sixth field, the **cron** daemon treats everything that precedes it as the command invocation and makes all that follows it available to standard input, unless you escape the percent sign (\%). Blank lines and lines whose first non-blank character is the number sign (#) will be ignored. If the arguments to the command have a backslash ('\'), the backslash should be preceded by another backslash.

Note: The shell runs only the first line of the command field. All other lines are made available to the command as standard input.

The **cron** daemon starts a subshell from your **HOME** directory. If you schedule a command to run when you are not logged in and you want commands in your **.profile** file to run, the command must explicitly read your **.profile** file.

The **cron** daemon supplies a default environment for every shell, defining **HOME, LOGNAME, SHELL** (=/**usr**/**bin**/**sh**), and **PATH** (=/**usr**/**bin**).

Flags

-e *UserName*	Edits a copy of the user's **crontab** file or creates an empty file to edit if the **crontab** file does not exist for a valid *UserName*. When editing is complete, the file is copied into the **crontab** directory as the user's **crontab** file.
-l *UserName*	Lists the user's **crontab** file.
-r *UserName*	Removes the user's **crontab** file from the **crontab** directory.
-v *UserName*	Lists the status of the user's cron jobs.

Security

Auditing Events: If the auditing subsystem has been properly configured and is enabled, the **crontab** command generates the following audit record (event) every time the command is run:

Event	**Information**
CRON_JobRemove	Lists which users removed a **cron** job and when.
CRON_JobAdd	Lists which users added a **cron** job and when.

Exit Status

This command returns the following exit values:

0	Successful completion.
>0	An error occurred.

Examples

1. To copy a file called `mycronjobs` into the **/var/spool/cron/crontabs** directory, enter the following:

   ```
   crontab mycronjobs
   ```

The file will be copied as:

```
/var/spool/cron/crontabs/<username>
```

where <*username*> is your current user name.

2. To write the time to the console every hour on the hour, enter:

   ```
   0 * * * * echo The hour is `date` .
   >/dev/console
   ```
3. To run the **calendar** command at 6:30 a.m. every Monday, Wednesday, and Friday, enter:

   ```
   30 6 * * 1,3,5 /usr/bin/calendar
   ```
4. To run the **calendar** command every day of the year at 6:30, enter the following:

   ```
   30 6 * * * /usr/bin/calendar
   ```
5. To run a script called `maintenance` every day at midnight in August, enter the following:

   ```
   0 0 * 8 * /u/harry/bin/maintenance
   ```
6. To define text for the standard input to a command, enter:

   ```
   0 16 * 12 5 /usr/sbin/wall%HAPPY HOLIDAY!%Remember to
   turn in your time card.
   ```

   The text following the **%** (percent sign) defines the standard input to the **wall** command as:

   ```
   HAPPY HOLIDAY!

   Remember to turn in your time card.
   ```

Files

/var/adm/cron/FIFO	A named pipe that sends messages to the **cron** daemon when new jobs are submitted with the **crontab** or **at** command.
/var/spool/cron/crontabs	Specifies the crontab spool area.
/var/adm/cron/cron.allow	Specifies a list of users allowed access to the **crontab** command.
/var/adm/cron/cron.deny	Specifies a list of users denied access to the **crontab** command.

date Command

Purpose

Displays or sets the date or time.

Syntax

To Set the Date and Time as Root User

/usr/bin/date -n -u *Date* **+***FieldDescriptor ...*

To Display the Date and Time

/usr/bin/date -u +*FieldDescriptor* ...

To adjust the Time in Seconds as root User

/usr/bin/date -a + | - *sss.fff*

Description

Attention: Do not change the date when the system is running with more than one user.

The **date** command writes the current date and time to standard output if called with no flags or with a flag list that begins with a **+** (plus sign). Otherwise, it sets the current date. Only a root user can change the date and time. The **date** command prints out the usage message on any unrecognized flags or input.

The following formats can be used when setting the date with the *Date* parameter:

- *mmddHHMMYYyy*
- *mmddHHMMyy*

The variables to the *Date* parameter are defined as follows:

mm	Specifies the month number.
dd	Specifies the number of the day in the month.
HH	Specifies the hour in the day (using a 24-hour clock).
MM	Specifies the minute number.
YY	Specifies the first two digits of the year. **Note:** If you do not specify the first two digits of the year, values in the range 69 to 99 refer to the twentieth century, 1969 to 1999 inclusive, and values in the range 00 to 68 refer to years in the twenty-first century, 2000 to 2068 inclusive.
yy	Specifies the last two digits of the year. **Note:** The **date** command accepts a 4 digit year as input. For example, if a four-digit year is specified, the **date** command tries to set the year to "YYyy" and fails for values which are out of range (less than 1970 and greater than 2037).

The current year is used as the default value when the year is not specified. The system operates in Coordinated Universal Time (CUT).

If you follow the **date** command with a **+** (plus sign) and a field descriptor, you can control the output of the command. You must precede each field descriptor with a **%** (percent sign). The system replaces the field descriptor with the specified value. Enter a literal % as %% (two percent signs). The **date** command copies any other characters to the output without change. The **date** command always ends the string with a new-line character.

Flags

-a + | - *sss.fff* Slowly adjusts the time by ***sss.fff*** seconds (*fff* represents fractions of a second). This adjustment can be positive or negative. The system's clock will be sped up or slowed down until it has drifted by the number of seconds specified.

-n Does not set the time globally on all machines in a local area network that have their clocks synchronized.

-u Displays or sets the time in Coordinated Universal Time (CUT).

Field Descriptors

%a Displays the locale's abbreviated weekday name.
%A Displays the locale's full weekday name.
%b Displays the locale's abbreviated month name.
%B Displays the locale's full month name.
%c Displays the locale's appropriate date and time representation. This is the default.
%C Displays the first two digits of the four-digit year as a decimal number (00-99). A year is divided by 100 and truncated to an integer.
%d Displays the day of the month as a decimal number (01-31). In a two-digit field, a 0 is used as leading space fill.
%D Displays the date in the format equivalent to **%m/%d/%y**.
%e Displays the day of the month as a decimal number (1-31). In a two-digit field, a blank space is used as leading space fill.
%h Displays the locale's abbreviated month name (a synonym for **%b**).
%H Displays the hour (24-hour clock) as a decimal number (00-23).
%I Displays the hour (12-hour clock) as a decimal number (01-12).
%j Displays the day of year as a decimal number (001-366).
%k Displays the 24-hour-clock hour clock as a right-justified, space-filled number (0 to 23).
%m Displays the month of year as a decimal number (01-12).
%M Displays the minutes as a decimal number (00-59).
%n Inserts a <new-line> character.
%p Displays the locale's equivalent of either AM or PM.
%r Displays 12-hour clock time (01-12) using the AM-PM notation; in the POSIX locale, this is equivalent to **%I:%M:%S %p**.
%S Displays the seconds as a decimal number (00- 59).
%s Displays the number of seconds since January 1, 1970, Coordinated Universal Time (CUT).
%t Inserts a <tab> character.
%T Displays the 24-hour clock (00-23) in the format equivalent to **HH:MM:SS** .
%u Displays the weekday as a decimal number from 1-7 (Sunday = 7). Refer to the **%w** field descriptor.
%U Displays week of the year(Sunday as the first day of the week) as a decimal number00 - 53 . All days in a new year preceding the first Sunday are considered to be in week 0.
%V Displays the week of the year as a decimal number from 01-53 (Monday is used as the first day of the week). If the week containing January 1 has four or more days in the new year, then it is considered week 01; otherwise, it is week 53 of the previous year.
%w Displays the weekday as a decimal number from 0-6 (Sunday = 0). Refer to the **%u** field descriptor.
%W Displays the week number of the year as a decimal number (00-53) counting Monday as the first day of the week.
%x Displays the locale's appropriate date representation.
%X Displays the locale's appropriate time representation.
%y Displays the last two numbers of the year (00-99).
%Y Displays the four-digit year as a decimal number.
%Z Displays the time-zone name, or no characters if no time zone is determinable.
%% Displays a % (percent sign) character.

Modified Field Descriptors

The **%E** and **%O** field descriptors can be modified to indicate a different format or specification. If the corresponding keyword (see the **era**, **era_year**, **era_d_fmt**, and **alt_digits** keywords) is not specified or not supported for the current locale, the unmodified field descriptor value is used.

%Ec Displays the locale's alternative appropriate date and time representation.

%EC Displays the name of the base year (or other time period) in the locale's alternative representation.

%Ex Displays the locale's alternative date representation.

%EX Displays the locale's alternative time representation.

%Ey Displays the offset from the **%EC** field descriptor (year only) in the locale's alternative representation.

%EY Displays the full alternative year representation.

%Od Displays the day of the month using the locale's alternative numeric symbols.

%Oe Displays the day of the month using the locale's alternative numeric symbols.

%OH Displays the hour (24-hour clock) using the locale's alternative numeric symbols.

%OI Displays the hour (12-hour clock) using the locale's alternative numeric symbols.

%Om Displays the month using the locale's alternative numeric symbols.

%OM Displays minutes using the locale's alternative numeric symbols.

%OS Displays seconds using the locale's alternative numeric symbols.

%Ou Displays the weekday as a number in the locale's alternative representation (Monday=1).

%OU Displays the week number of the year using the locale's alternative numeric symbols. Sunday is considered the first day of the week.

%OV Displays the week number of the year using the locale's alternative numeric symbols. Monday is considered the first day of the week.

%Ow Displays the weekday as a number in the locale's alternative representation (Sunday =0).

%OW Displays the week number of the year using the locale's alternative numeric symbols. Monday is considered the first day of the week.

%Oy Displays the year (offset from %C) in alternative representation.

Exit Status

This command returns the following exit values:

0 The date was written successfully.

>0 An error occurred.

Examples

1. To display current date and time, enter:

   ```
   date
   ```
2. To set the date and time, enter:

   ```
   date 0217142590
   ```

   For a system using CST as its time zone, this sets the date and time to Sat Feb 17 14:25:00 CST 1990.

   Note: You must have root authority to change the date and time.
3. To display the date and time in a specified format, enter:

   ```
   date +"%r %a %d %h %y (Julian Date: %j)"
   ```

   This displays the date shown in Example 2 as:

   ```
   02:25:03 PM Fri 17 Feb 90 (Julian Date: 048)
   ```

Environment Variables

The following environment variables affect the execution of the **date** command.

LANG Determines the locale to use when both **LC_ALL** and the corresponding environment variable (beginning with **LC_**) do not specify a locale.

LC_ALL

Determines the locale to be used to override any values for locale categories specified by the setting of **LANG** or any environment variable beginning with **LC_**.

LC_CTYPE
Determines the locale for the interpretation of sequences of bytes of text data as characters (for example, single versus multibyte character in an argument).

LC_MESSAGES
Determines the language in which messages should be written.

LC_TIME
Determines the contents of date and time strings written by **date**.

NLSPATH
Determines the location of message catalogues for the processing of **LC_MESSAGES**.

TZ
Specifies the time zone in which the time and date are written, unless the **-u** option is specified. If the **TZ** variable is not set and the **-u** flag is not specified, an unspecified system default time zone is used.

deactivatevg Command

Purpose

Deactivates a volume group.

Syntax

deactivatevg *VolumeGroup*

Description

The **deactivatevg** command deactivates the volume group specified by the *VolumeGroup* parameter along with its associated logical volumes. The logical volumes must first be closed. For example, if the logical volume contains a file system, it must be unmounted.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To deactivate volume group **vg03**, type:

```
deactivatevg vg03
```

Related Information

The **mkvg** command, the **chvg** command, the **extendvg** command, the **reducevg** command, the **mirrorios** command, the **unmirrorios** command, the **lsvg** command, the **activatevg** command, the **importvg** command, the **exportvg** command, and the **syncvg** command.

diagmenu Command

Purpose

Places the user into the diagnostic menus.

Syntax

diagmenu

Description

The **diagmenu** command performs hardware problem determination and maintenance. When the user suspects there is a problem, **diagmenu** assists in finding it. Through the diagnostic menus, users can perform tasks such as running diagnostics, hot plugging devices, formatting and certifying media, and managing RAID devices.

Once inside the diagnostic menus, task specific help can be obtained by pressing the **F1** key.

Related Information

The **errlog** command.

entstat Command

Purpose

Shows Ethernet device driver and device statistics.

Syntax

entstat **-all** **-reset** *Device_Name*

Description

The **entstat** command displays the statistics gathered by the specified Ethernet device driver. The user can optionally specify that the device-specific statistics be displayed in addition to the device generic statistics. If no flags are specified, only the device generic statistics are displayed.

Flags

-all	Displays all the statistics, including the device-specific statistics.
-reset	Resets all the statistics back to their initial values.

Parameters

Device_name	The name of the Ethernet device, for example, `ent0`.

Exit Status

The statistic fields displayed in the output of the **entstat** command and their descriptions are as follows.

Note: Some adapters may not support a specific statistic. The value of non-supported statistic fields is always 0.

Title Fields

Device Type	Displays the description of the adapter type.
Hardware Address	Displays the Ethernet network address currently used by the device.
Elapsed Time	Displays the real time period which has elapsed since last time the statistics were reset. Part of the statistics may be reset by the device driver during error recovery when a hardware error is detected. There will be another Elapsed Time displayed in the middle of the output when this situation has occurred in order to reflect the time differences between the statistics.

Transmit Statistics Fields

Packets	The number of packets transmitted successfully by the device.
Bytes	The number of bytes transmitted successfully by the device.
Interrupts	The number of transmit interrupts received by the driver from the adapter.
Transmit Errors	The number of output errors encountered on this device. This is a counter for unsuccessful transmissions due to hardware or network errors.
Packets Dropped	The number of packets accepted by the device driver for transmission which were not (for any reason) given to the device.
Max Packets on S/W Transmit Queue	The maximum number of outgoing packets ever queued to the software transmit queue.
S/W Transmit Queue Overflow	The number of outgoing packets which have overflowed the software transmit queue.
Current S/W+H/W Transmit Queue Length	The number of pending outgoing packets on either the software transmit queue or the hardware transmit queue.

Broadcast Packets	The number of broadcast packets transmitted without any error.
Multicast Packets	The number of multicast packets transmitted without any error.
No Carrier Sense	The number of unsuccessful transmissions due to the no carrier sense error.
DMA Underrun	The number of unsuccessful transmissions due to the DMA underrun error.
Lost CTS Errors	The number of unsuccessful transmissions due to the loss of the Clear-to-Send signal error.
Max Collision Errors	The number of unsuccessful transmissions due to too many collisions. The number of collisions encountered exceeded the number of retries on the adapter.
Late Collision Errors	The number of unsuccessful transmissions due to the late collision error.
Deferred	The number of outgoing packets deferred during transmission. Deferred means that the adapter had to defer while trying to transmit a frame. This condition occurs if the network is busy when the adapter is ready to transmit. The adapter will only defer the first attempt to send a packet. After that the adapter will transmit the packet without checking. If the network is still busy then a collision will be recorded.
SQE Test	Contains the number of "Signal Quality Error" Tests (i.e. Heartbeat) performed successfully during transmission.
Timeout Errors	The number of unsuccessful transmissions due to adapter reported timeout errors.
Single Collision Count	The number of outgoing packets with single (only one) collision encountered during transmission.
Multiple Collision Count	The number of outgoing packets with multiple (2 - 15) collisions encountered during transmission
Current HW Transmit Queue Length	The number of outgoing packets which currently exist on the hardware transmit queue.
CRC Errors	The number of incoming packets with the Checksum (FCS) error.
DMA Overrun	The number of incoming packets with the DMA overrun error.
Alignment Errors	The number of incoming packets with the alignment error.
No Resource Errors	The number of incoming packets dropped by the hardware due to the no resource error. This error usually occurs because the receive buffers on the adapter were exhausted. Some adapters may have the size of the receive buffers as a configurable parameter. Check the device configuration attributes for possible tuning information.
Receive Collision Errors	The number of incoming packets with the collision errors during the reception.
Packet Too Short Errors	The number of incoming packets with the length error indicating that the packet size is less than the Ethernet minimum packet size.
Packet Too Long Errors	The number of incoming packets with the length error indicating that the packet size is bigger than the Ethernet maximum packet size.
Packets Discarded by Adapter	The number of incoming packets dropped by the hardware for any other reasons.
Receiver Start Count	The number of times that the receiver (receive unit) on the adapter has been started.

Examples

1. To display the device generic statistics for **ent0**, type:

```
entstat ent0
```

This produces output similar to the following:

```
ETHERNET STATISTICS (ent0) :
Device Type: Ethernet High Performance LAN Adapter
Hardware Address: 02:60:8c:2e:d0:1d
Elapsed Time: 0 days 0 hours 8 minutes 41 seconds

Transmit Statistics:     Receive Statistics:
--------------------     -------------------
Packets: 3               Packets: 2
Bytes: 272               Bytes: 146
Interrupts: 3            Interrupts: 2
Transmit Errors: 0       Receive Errors: 0
Packets Dropped: 0       Packets Dropped: 0
Max Packets on S/W       Bad Packets: 0
Transmit Queue:0
S/W Transmit Queue
Overflow: 0
```

```
Current S/W+H/W Transmit
Queue Length: 0

Broadcast Packets: 2     CRC Errors: 0
Multicast Packets: 0     Broadcast Packets: 1
No Carrier Sense: 0      Multicast Packets: 0
DMA Underrun: 0          DMA Overrun: 0
Lost CTS Errors: 0       Alignment Errors: 0
Max Collision Errors: 0  No Resource Errors: 0
Late Collision Errors: 0 Receive Collision Errors: 0
Deferred: 0              Packet Too Short Errors: 0
SQE Test: 0              Packet Too Long Errors: 0
Timeout Errors: 0        Packets Discarded by Adapter: 0
Single Collision         Receiver Start Count: 1
Count: 0
Multiple Collision Count: 0
Current HW Transmit Queue
Length: 0

General Statistics:
-------------------
No mbuf Errors: 0
Adapter Reset Count: 0
Driver Flags: Up Broadcast Running Simplex
```

2. To display the Ethernet device generic statistics and the Ethernet device-specific statistics for **ent0**, type:

   ```
   entstat -all
   ```

   Results similar to the following will be displayed:

   ```
   ETHERNET STATISTICS (ent0) :
   Device Type: Ethernet High Performance LAN Adapter
   Hardware Address: 02:60:8c:2e:d0:1d
   Elapsed Time: 0 days 2 hours 6 minutes 30 seconds

   Transmit Statistics:      Receive Statistics:
   --------------------      -------------------
   Packets: 3                Packets: 2
   Bytes: 272                Bytes: 146
   Interrupts: 3             Interrupts: 2
   Transmit Errors: 0        Receive Errors: 0
   Packets Dropped: 0        Packets Dropped: 0
   Max Packets on S/W        Receiver Start Count: 1
   Transmit Queue:0
   Bad Packets: 0
   S/W Transmit Queue Overflow: 0
   Current S/W+H/W Transmit Queue Length: 0

   Broadcast Packets: 0      Broadcast Packets: 0
   Multicast Packets: 0      Multicast Packets: 0
   No Carrier Sense: 0       CRC Errors: 0
   DMA Underrun: 0           DMA Overrun: 0
   Lost CTS Errors: 0        Alignment Errors: 0
   Max Collision Errors: 0   No Resource Errors: 0
   Late Collision Errors: 0  Receive Collision Errors: 0
   Deferred: 0               Packet Too Short Errors: 0
   SQE Test: 0               Packet Too Long Errors: 0
   Timeout Errors: 0         Packets Discarded by Adapter: 0
   Single Collision Count: 0 Receiver Start Count: 1
   Multiple Collision Count: 0
   Current HW Transmit Queue Length: 0

   General Statistics:
   -------------------
   No mbuf Errors: 0
   Adapter Reset Count: 0
   Driver Flags: Up Broadcast Running Simplex

   Ethernet High Performance LAN Adapter Specific Statistics:
   ----------------------------------------------------------
   Receive Buffer Pool Size: 37
   Transmit Buffer Pool Size: 39
   In Promiscuous Mode for IP Multicast: No
   Packets Uploaded from Adapter: 0
   Host End-of-List Encountered: 0
   82586 End-of-List Encountered: 0
   Receive DMA Timeouts: 0
   Adapter Internal Data: 0x0 0x0 0x0 0x0 0x0
   ```

Related Information

The **optimizenet** command.

errlog Command

Purpose

Displays or clears the error log.

Syntax

errlog **-ls** **-seq** *SequenceNumber* | **-rm** *Days*

Description

The **errlog** command generates an error report from entries in the Virtual I/O Server error log or deletes all entries from the error log older the number of days specified by the *Days* parameter. To delete all error-log entries, specify a value of `0` for the *Days* parameter. If no flags are specified a summary report is displayed.

Flags

-ls	Displays detailed information about errors in the error log file.
-rm *Days*	Deletes error-log entries older than the number of days specified by the Days parameter.
-seq *SequenceNumber*	Displays information about a specific error in the error log file by the sequence number.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To display a complete detailed report, enter:

   ```
   errlog -ls
   ```
2. To delete error-log entries older than 5 days, enter:

   ```
   errlog -rm 5
   ```
3. To delete all error-log entries, enter:

   ```
   errlog -rm 0
   ```

Related Information

The **diagmenu** command.

exportvg Command

Purpose

Exports the definition of a volume group from a set of physical volumes.

Syntax

exportvg *VolumeGroup*

Description

The **exportvg** command removes the definition of the volume group specified by the *VolumeGroup* parameter from the system. The primary use of the **exportvg** command, coupled with the **importvg** command, is to allow portable volumes to be exchanged between systems. Only a complete volume group can be exported, not individual physical volumes.

Using the **exportvg** command and the **importvg** command, you can also switch ownership of data on physical volumes shared between two processors.

Mount points longer than 128 characters will not automatically be re-mounted when the volume group is imported using the **importvg** command and should not be used.

Note: A volume group that has a paging space volume on it cannot be exported.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To remove volume group **vg02** from the system, type:

   ```
   exportvg vg02
   ```

   Note: The volume group must be deactivated before exporting. The definition of **vg02** is removed from the system and the volume group cannot be accessed.

Restrictions

Mount points longer than 128 characters should not be used.

A volume group that has a paging space volume on it cannot be exported.

Related Information

The **mkvg** command, the **chvg** command, the **lsvg** command, the **reducevg** command, the **mirrorios** command, the **unmirrorios** command, the **activatevg** command, the **deactivatevg** command, the **importvg** command, the **extendvg** command, and the **syncvg** command.

extendlv Command

Purpose

Increases the size of a logical volume.

Syntax

extendlv *LogicalVolume Size PhysicalVolume* ...

Description

The **extendlv** command increases the size of the logical volume. The *LogicalVolume* parameter can be a logical volume name or a logical volume ID. To limit the allocation to specific physical volumes, use the names of one or more physical volumes in the *PhysicalVolume* parameter; otherwise, all the physical volumes in a volume group are available for allocating new physical partitions.

The *Size* parameter specifies the minimum size the logical volume should be increased by. When specifying *Size* the following conventions must be used:

Size	Logical volume size
###M/m	### MB
###G/g	### GB

Exit Status

See Virtual I/O Server command exit status

Examples

1. To increase the size of the logical volume **lv05** by three megabytes, type:

   ```
   extendlv lv05 3M
   ```
2. To request a logical volume named **lv05** with a minimum size of 10MB, type:

   ```
   extendlv lv05 10M
   ```

   The **extendlv** command will determine the number of partitions needed to create a logical volume of at least that size.

Related Information

The **mklv** command, the **lslv** command, and the **rmlv** command.

extendvg Command

Purpose

Adds physical volumes to a volume group.

Syntax

extendvg -f *VolumeGroup PhysicalVolume* ...

Description

The **extendvg** command increases the size of *VolumeGroup* by adding one or more *PhysicalVolumes*.

The physical volume is checked to verify that it is not already in another volume group. If the system believes the physical volume belongs to a volume group that is activated, it exits. But if the system detects a description area from a volume group that is not activated, it prompts the user for confirmation in continuing with the command. The previous contents of the physical volume are lost, so the user must be cautious when using the override function.

Flags

-f	Forces the physical volume to be added to the specified volume group unless it is a member of another volume group that has been activated by the **activatevg** command.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To add physical volumes **hdisk3** and **hdisk8** to volume group **vg3**, type:

   ```
   extendvg vg3 hdisk3 hdisk8
   ```

   Note: The volume group must be activated before extending.

Related Information

The **mkvg** command, the **chvg** command, the **lsvg** command, the **reducevg** command, the **mirrorios** command, the **unmirrorios** command, the **activatevg** command, the **deactivatevg** command, the **importvg** command, the **exportvg** command, and the **syncvg** command.

fsck Command

Purpose

Checks file system consistency and interactively repairs the file system.

Syntax

fsck *FileSystem* ...

Description

The **fsck** command checks and interactively repairs inconsistent file systems. Normally, the file system is consistent, and the **fsck** command merely reports on the number of files, used blocks, and free blocks in the file system. If the file system is inconsistent, the **fsck** command displays information about the inconsistencies found and prompts you for permission to repair them. If no *FileSystem* is specified, all file systems are checked.

The **fsck** command checks for the following inconsistencies:

- Blocks or fragments allocated to multiple files.
- i-nodes containing block or fragment numbers that overlap.
- i-nodes containing block or fragment numbers out of range.
- Discrepancies between the number of directory references to a file and the link count of the file.
- Illegally allocated blocks or fragments.
- i-nodes containing block or fragment numbers that are marked free in the disk map.
- i-nodes containing corrupt block or fragment numbers.
- A fragment that is not the last disk address in an i-node. This check does not apply to compressed file systems.
- Files larger than 32KB containing a fragment. This check does not apply to compressed file systems.
- Size checks:
    - ♦ Incorrect number of blocks.
    - ♦ Directory size not a multiple of 512 bytes.

  Note: These checks do not apply to compressed file systems.
- Directory checks:
    - ♦ Directory entry containing an i-node number marked free in the i-node map.
    - ♦ i-node number out of range.
    - ♦ Dot (.) link missing or not pointing to itself.
    - ♦ Dot dot (..) link missing or not pointing to the parent directory.
    - ♦ Files that are not referenced or directories that are not reachable.
- Inconsistent disk map.
- Inconsistent i-node map.

In addition to its messages, the **fsck** command records the outcome of its checks and repairs through its exit value. This exit value can be any sum of the following conditions:

0	All checked file systems are now okay.
2	The **fsck** command was interrupted before it could complete checks or repairs.
4	The **fsck** command changed the file system; the user must restart the system immediately.
8	The file system contains unrepaired damage.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To check a file system, enter:

```
fsck /dev/hd1
```

This command checks the unmounted file system located on the **/dev/hd1** device.

ftp Command

Purpose

Transfers files between a local and a remote host.

Syntax

ftp -d -g -i -n -v -f -k *realm* **-q-C** *HostName Port*

Description

The **ftp** command uses the File Transfer Protocol (FTP) to transfer files between the local host and a remote host or between two remote hosts.

The FTP protocol allows data transfer between hosts that use dissimilar file systems. Although the protocol provides a high degree of flexibility in transferring data, it does not attempt to preserve file attributes (such as the protection mode or modification times of a file) that are specific to a particular file system. Moreover, the FTP protocol makes few assumptions about the overall structure of a file system and does not provide or allow such functions as recursively copying subdirectories.

Issuing Subcommands

At the `ftp>` prompt, you can enter subcommands to perform tasks such as listing remote directories, changing the current local and remote directory, transferring multiple files in a single request, creating and removing directories, and escaping to the local shell to perform shell commands. See the Subcommands section for a description of each subcommand.

If you execute the **ftp** command and do not specify the *HostName* parameter for a remote host, the **ftp** command immediately displays the `ftp>` prompt and waits for an **ftp** subcommand. To connect to a remote host, execute the **open** subcommand**.** When the **ftp** command connects to the remote host, the **ftp** command then prompts for the login name and password before displaying the `ftp>` prompt again. The **ftp** command is unsuccessful if no password is defined at the remote host for the login name.

The **ftp** command interpreter, which handles all subcommands entered at the `ftp>` prompt, provides facilities that are not available with most file-transfer programs, such as:

- Handling file-name parameters to **ftp** subcommands
- Collecting a group of subcommands into a single subcommand macro
- Loading macros from a **$HOME/.netrc** file

These facilities help simplify repetitive tasks and allow you to use the **ftp** command in unattended mode.

The command interpreter handles file-name parameters according to the following rules:

- If a - (hyphen) is specified for the parameter, standard input (stdin) is used for read operations and standard output (stdout) is used for write operations.
- If the preceding check does not apply and file-name expansion is enabled (see the **-g** flag or the **glob** subcommand), the interpreter expands the file name according to the rules of the C shell. When globbing is enabled and a pattern-matching character is used in a subcommand that expects a single file name, results may be different than expected.

For example, the **append** and **put** subcommands perform file-name expansion and then use only the first file name generated. Other **ftp** subcommands, such as **cd**, **delete**, **get**, **mkdir**, **rename**, and **rmdir**, do not perform file-name expansion and take the pattern-matching characters literally.
- For the **get**, **put**, **mget**, and **mput** subcommands, the interpreter has the ability to translate and map between different local and remote file-name syntax styles (see the **case**, **ntrans**, and **nmap** subcommands) and the ability to modify a local file name if it is not unique (see the **runique** subcommand). Additionally, the **ftp** command can send instructions to a remote **ftpd** server to modify a remote file name if it is not unique (see the **sunique** subcommand).
- Use double quotes (" ") to specify parameters that include blank characters.

> **Note:** The **ftp** command interpreter does not support pipes. It also does not necessarily support all multibyte-character file names.

To end an **ftp** session when you are running interactively, use the **quit** or **bye** subcommand or the End of File (Ctrl-D) key sequence at the `ftp>` prompt. To end a file transfer before it has completed, press the Interrupt key sequence. The default Interrupt key sequence is Ctrl-C. The **stty** command can be used to redefine this key sequence.

The **ftp** command normally halts transfers being sent (from the local host to the remote host) immediately. The **ftp** command halts transfers being received (from the remote host to the local host) by sending an FTP ABOR instruction to the remote FTP server and discarding all incoming file transfer packets until the remote server stops sending them. If the remote server does not support the ABOR instruction, the **ftp** command does not display the `ftp>` prompt until the remote server has sent all of the requested file. Additionally, if the remote server does something unexpected, you may need to end the local **ftp** process.

Security and Automatic Login

If Standard is the current authentication method:

The **ftp** command also handles security by sending passwords to the remote host and permits automatic login, file transfers, and logoff.

If you execute the **ftp** command and specify the host name (*HostName*) of a remote host, the **ftp** command tries to establish a connection to the specified host. If the **ftp** command connects successfully, the **ftp** command searches for a local **$HOME/.netrc** file in your current directory or home directory. If the file exists, the **ftp** command searches the file for an entry initiating the login process and command macro definitions for the remote host. If the **$HOME/.netrc** file or automatic login entry does not exist the **ftp** command prompts the user for a user name and password. The command displays the prompt whether or not the *HostName* parameter is specified on the command line.

Note: The queuing system does not support multibyte host names.

If the **ftp** command finds a **$HOME/.netrc** automatic login entry for the specified host, the **ftp** command attempts to use the information in that entry to log in to the remote host. The **ftp** command also loads any command macros defined in the entry. In some cases (for example, when the required password is not listed in an automatic login entry), the **ftp** command prompts for the password before displaying the `ftp>` prompt.

Once the **ftp** command completes the automatic login, the **ftp** command executes the **init** macro if the macro is defined in the automatic login entry. If the **init** macro does not exist or does not contain a **quit** or **bye** subcommand, the **ftp** command then displays the `ftp>` prompt and waits for a subcommand.

Note: The remote user name specified either at the prompt or in a **$HOME/.netrc** file must exist and have a password defined at the remote host. Otherwise, the **ftp** command fails.

If Kerberos 5 is the current authentication method:

The **ftp** command will use the extensions to ftp specifications as defined in IETF draft document "draft-ietf-cat-ftpsec-09.txt". The FTP security extensions will be implemented using the Generic Security Service API (GSSAPI) security mechanism. The GSSAPI provides services independent to the underlying security and communication mechanism. The GSSAPI is defined in rfc 1508 and 1509.

The **ftp** command will use the AUTH and ADAT commands to authenticate with the **ftpd** daemon. If both support Kerberos authentication, then they will use the local users DCE credentials to authenticate the user on the remote system. If this fails and Standard authentication is configured on both systems, the process described above will be used.

The *HostName* parameter is the name of the host machine to which files are transferred. The optional *Port* parameter specifies the ID of the port through which to transmit. (The **/etc/services** file specifies the default port.)

Flags

-C Allows the user to specify that the outgoing file sent using the **send_file** command must be cached in the Network Buffer Cache (NBC). This flag cannot be used unless the **-q** flag is specified. This flag is only applicable when a file is being sent out in the binary mode with no protection.

-d Sends debugging information about **ftp** command operations to the **syslogd** daemon. If you specify the **-d** flag, you must edit the **/etc/syslog.conf** file and add one of the following entries:

```
user.info FileName
```

OR

```
user.debug FileName
```

> **Note:** The **syslogd** daemon debug level includes info level messages.

If you do not edit the **/etc/syslog.conf** file, no messages are produced. After changing the **/etc/syslog.conf** file, run the **refresh -s syslogd** or **kill -1** *SyslogdPID* command to inform the **syslogd** daemon of the changes to its configuration file. For more information about debug levels, refer to the **/etc/syslog.conf** file. Also, refer to the **debug** subcommand**.**

-g Disables the expansion of metacharacters in file names. Interpreting metacharacters can be referred to as expanding (sometimes called globbing) a file name. See the **glob** subcommand.

-i Turns off interactive prompting during multiple file transfers. See the **prompt** , **mget** , **mput**, and **mdelete** subcommands for descriptions of prompting during multiple file transfers.

-n Prevents an automatic login on the initial connection. Otherwise, the **ftp** command searches for a **$HOME/.netrc** entry that describes the login and initialization process for the remote host. See the **user** subcommand**.**

-q Allows the user to specify that the **send_file** subroutine must be used for sending the file on the network. This flag is only applicable when a file is being sent out in the binary mode with no protection.

-v Displays all the responses from the remote server and provides data transfer statistics. This display mode is the default when the output of the **ftp** command is to a terminal, such as the console or a display.

If stdin is not a terminal, the **ftp** command disables verbose mode unless the user invoked the **ftp** command with the **-v** flag or issued the **verbose** subcommand.

-f Causes the credentials to be forwarded. This flag will be ignored if Kerberos 5 is not the current authentication method.

-k *realm* Allows the user to specify the realm of the remote station if it is different from the local systems realm. For these purposes, a realm is synonymous with a DCE cell. This flag will be ignored if Kerberos 5 is not the current authentication method.

Subcommands

The following **ftp** subcommands can be entered at the `ftp>` prompt. Use double quotes (" ") to specify parameters that include blank characters.

!*Command Parameters* Invokes an interactive shell on the local host. An optional command, with one or more optional parameters, can be given with the shell command.

$*Macro Parameters*
Executes the specified macro, previously defined with the **macdef** subcommand. Parameters are not expanded.

?*Subcommand*
Displays a help message describing the subcommand. If you do not specify a *Subcommand* parameter, the **ftp** command displays a list of known subcommands.

account *Password*
Sends a supplemental password that a remote host may require before granting access to its resources. If the password is not supplied with the command, the user is prompted for the password. The password is not displayed on the screen.

append *LocalFile RemoteFile*
Appends a local file to a file on the remote host. If the remote file name is not specified, the local file name is used, altered by any setting made with the **ntrans** subcommand or the **nmap** subcommand. The **append** subcommand uses the current values for **form** , **mode**, **struct**, and **type** subcommands while appending the file.

ascii
Synonym for the **type ascii** subcommand.

bell
Sounds a bell after the completion of each file transfer.

binary
Synonym for the **type binary** subcommand.

block
Synonym for the **mode block** subcommand.

bye
Ends the file-transfer session and exits the **ftp** command. Same as the **quit** subcommand.

carriage-control
Synonym for the **form carriage-control** subcommand.

case
Sets a toggle for the case of file names. When the **case** subcommand is On, the **ftp** command changes remote file names displayed in all capital letters from uppercase to lowercase when writing them in the local directory. The default is Off (so the **ftp** command writes uppercase remote file names in uppercase in the local directory).

cd *RemoteDirectory*
Changes the working directory on the remote host to the specified directory.

cdup
Changes the working directory on the remote host to the parent of the current directory.

close
Ends the file-transfer session, but does not exit the **ftp** command. Defined macros are erased. Same as the **disconnect** subcommand.

copylocal
Toggles local copy. **copylocal** defaults to off. An effort is made by ftp to make sure you do not zero out a file by ftp'ing it to itself (eg. same hostname, same pathname). Turning **copylocal** ON bypasses this check.

cr
Strips the carriage return character from a carriage return and line-feed sequence when receiving records during ASCII-type file transfers. (The **ftp** command terminates each ASCII-type record with a carriage return and line feed during file transfers.)

Records on remote hosts with operating systems other than the one you are running can have single line feeds embedded in records. To distinguish these embedded line feeds from record delimiters, set the **cr** subcommand to Off. The **cr** subcommand toggles between On and Off.

debug 0 | 1
Toggles debug record keeping On and Off. Specify **debug** or **debug 1** to print each command sent to the remote host and save the restart control file. Specify **debug** again, or **debug 0**, to stop the debug record keeping. The Ctrl-C key sequence also saves the restart control file.

Specifying the **debug** subcommand sends debugging information about **ftp** command operations to the **syslogd** daemon. If you specify the **debug** subcommand, you must edit the **/etc/syslog.conf** file and add one of the following entries:

```
user.info FileName
```

OR

```
user.debug FileName
```

Note: The **syslogd** daemon debug level includes info level messages.

If you do not edit the /**etc**/**syslog.conf** file, no messages are produced. After changing the /**etc**/**syslog.conf** file, run the **refresh -s syslogd** or **kill -1** *SyslogdPID* command to inform the **syslogd** daemon of the changes to its configuration file. For more information about debug levels, refer to the /**etc**/**syslog.conf** file. Also, refer to the **ftp -d** flag.

delete *RemoteFile*
Deletes the specified remote file.

dir *RemoteDirectoryLocalFile*
Writes a listing of the contents of the specified remote directory (*RemoteDirectory*) to the specified local file (*LocalFile*). If the *RemoteDirectory* parameter is not specified, the **dir** subcommand lists the contents of the current remote directory. If the *LocalFile* parameter is not specified or is a - (hyphen), the **dir** subcommand displays the listing on the local terminal.

disconnect
Ends the file-transfer session but does not exit the **ftp** command. Defined macros are erased. Same as the **close** subcommand.

ebcdic
Synonym for the **type ebcdic** subcommand.

exp_cmd
Toggles between conventional and experimental protocol commands. The default is off.

file
Synonym for the **struct file** subcommand.

form carriage-control | non-print | telnet
Specifies the form of the file transfer. The **form** subcommand modifies the **type** subcommand to send the file transfer in the indicated form. Valid arguments are **carriage-control**, **non-print**, and **telnet**.

carriage-control
Sets the form of the file transfer to carriage-control.

non-print
Sets the form of the file transfer to non-print.

telnet
Sets the form of the file transfer to Telnet. Telnet is a Transmission Control Protocol/Internet Protocol (TCP/IP) protocol that opens connections to a system.

get *RemoteFile LocalFile*
Copies the remote file to the local host. If the *LocalFile* parameter is not specified, the remote file name is used locally and is altered by any settings made by the **case**, **ntrans**, and **nmap** subcommands**.** The **ftp** command uses the current settings for the **type**, **form**, **mode**, and **struct** subcommands while transferring the file.

glob
Toggles file-name expansion (globbing) for the **mdelete**, **mget**, and **mput** subcommands. If globbing is disabled, file-name parameters for these subcommands are not expanded. When globbing is enabled and a pattern-matching character is used in a subcommand that expects a single file name, results may be different than expected.

For example, the **append** and **put** subcommands perform file-name expansion and then use only the first file name generated. Other **ftp** subcommands, such as **cd**, **delete**, **get**, **mkdir**, **rename**, and **rmdir**, do not perform file-name expansion and take the pattern-matching characters literally.

Globbing for the **mput** subcommand is done locally in the same way as for the **csh** command. For the **mdelete** and **mget** subcommands, each file name is expanded separately at the remote machine and the lists are not merged. The expansion of a directory name can be different from the expansion of a file name, depending on the remote host and the **ftp** server.

To preview the expansion of a directory name, use the **mls** subcommand:

```
mls RemoteFile
```

To transfer an entire directory subtree of files, transfer a **tar** archive of the subtree in binary form, rather than using the **mget** or **mput** subcommand.

hash
Toggles hash sign (#) printing. When the **hash** subcommand is on, the **ftp** command displays one hash sign for each data block (1024 bytes) transferred.

help *Subcommand*
Displays help information. See the **?** subcommand.

image
Synonym for the **type image** subcommand.

lcd *Directory*
Changes the working directory on the local host. If you do not specify a directory, the **ftp** command uses your home directory.

local *M*
Synonym for the **type local** *M* subcommand.

ls *RemoteDirectory LocalFile*
Writes an abbreviated file listing of a remote directory to a local file. If the *RemoteDirectory* parameter is not specified, the **ftp** command lists the current remote directory. If the *LocalFile* parameter is not specified or is a - (hyphen), the **ftp** command displays the listing on the local terminal.

macdef *Macro*
Defines a subcommand macro. Subsequent lines up to a null line (two consecutive line feeds) are saved as the text of the macro. Up to 16 macros, containing at most 4096 characters for all macros, can be defined. Macros remain defined until either redefined or a **close** subcommand is executed.

The $ (dollar sign) and \ (backslash) are special characters in **ftp** macros. A $ symbol followed by one or more numbers is replaced by the corresponding macro parameter on the invocation line (see the **$** subcommand). A $ symbol followed by the letter i indicates that the macro is to loop, with the $i character combination being replaced by consecutive parameters on each pass.

The first macro parameter is used on the first pass, the second parameter is used on the second pass, and so on. A \ symbol prevents special treatment of the next character. Use the \ symbol to turn off the special meanings of the $ and \. (backslash period) symbols.

mdelete *RemoteFiles*
Expands the files specified by the *RemoteFiles* parameter at the remote host and deletes the remote files.

mdir *RemoteDirectories LocalFile*
Expands the directories specified by the *RemoteDirectories* parameter at the remote host and writes a listing of the contents of those directories to the file specified in the *LocalFile* parameter. If the *RemoteDirectories* parameter contains a pattern-matching character, the **mdir** subcommand prompts for a local file if none is specified. If the *RemoteDirectories* parameter is a list of remote directories separated by blanks, the last argument in the list must be either a local file name or a - (hyphen).

If the *LocalFile* parameter is - (hyphen), the **mdir** subcommand displays the listing on the local terminal. If interactive prompting is on (see the **prompt** subcommand), the **ftp** command prompts the user to verify that the last parameter is a local file and not a remote directory.

mget *RemoteFiles*
Expands the *RemoteFiles* parameter at the remote host and copies the indicated remote files to the current directory on the local host. See the **glob** subcommand for more information on file-name expansion. The remote file names are used locally and are altered by any settings made by the **case**, **ntrans**, and **nmap** subcommands. The **ftp** command uses the current settings for the **form**, **mode**, **struct**, and **type** subcommands while transferring the files.

mkdir *RemoteDirectory*
Creates the directory specified in the *RemoteDirectory* parameter on the remote host.

mls *RemoteDirectories LocalFile*
Expands the directories specified in the *RemoteDirectories* parameter at the remote host and writes an abbreviated file listing of the indicated remote directories to a local file. If the *RemoteDirectories* parameter contains a pattern-matching character, the **mls** subcommand prompts for a local file if none is specified. If the *RemoteDirectories* parameter is a list of remote directories separated by blanks, the last argument in the list must be either a local file name or a - (hyphen).

If the *LocalFile* parameter is - (hyphen), the **mls** subcommand displays the listing on the local terminal. If interactive prompting is on (see the **prompt** subcommand), the **ftp** command prompts the user to verify that the last parameter is a local file and not a remote directory.

mode stream | block

Sets file-transfer mode. If an argument is not supplied, the default is **stream**.

block
: Sets the file-transfer mode to block.

stream
: Sets the file-transfer mode to stream.

modtime

Shows the last modification time of the specified file on the remote machine. If the **ftp** command is not connected to a host prior to execution, the **modtime** subcommand terminates with an error message. The **ftp** command ignores parameter beyond the first parameter. If the *FileName* parameter is not specified, the **ftp** command prompts for a file name. If no file name is given, the **ftp** command sends a usage message to standard output and terminates the subcommand.

If the name specified by the *FileName* parameter exists on the remote host, and the name specifies a file, then the **ftp** command sends a message containing the last modification time of the file to standard output and terminates the subcommand. If *FileName* specifies a directory, the **ftp** command sends an error message to standard output and terminates the subcommand.

Note: The **modtime** subcommand interprets metacharacters when allowed.

mput *LocalFiles*

Expands the files specified in the *LocalFiles* parameter at the local host and copies the indicated local files to the remote host. See the **glob** subcommand for more information on file-name expansion. The local file names are used at the remote host and are altered by any settings made by the **ntrans** and **nmap** subcommands. The **ftp** command uses the current settings for the **type**, **form**, **mode**, and **struct** subcommands while transferring the files.

nlist *RemoteDirectoryLocalFile*

Writes a listing of the contents of the specified remote directory (*RemoteDirectory*) to the specified local file (*LocalFile*). If the *RemoteDirectory* parameter is not specified, the **nlist** subcommand lists the contents of the current remote directory. If the *LocalFile* parameter is not specified or is a - (hyphen), the **nlist** subcommand displays the listing on the local terminal.

nmap *InPattern OutPattern*

Turns the file-name mapping mechanism On or Off. If no parameters are specified, file-name mapping is turned off. If parameters are specified, source file names are mapped for the **mget** and **mput** subcommands and for the **get** and **put** subcommands when the destination file name is not specified. This subcommand is useful when the local and remote hosts use different file-naming conventions or practices. Mapping follows the pattern set by the *InPattern* and *OutPattern* parameters.

The *InPattern* parameter specifies the template for incoming file names, which may have already been processed according to the **case** and **ntrans** settings. The template variables `$1` through `$9` can be included in the *InPattern* parameter. All characters in the *InPattern* parameter, other than the $ (dollar sign) and the \$ (backslash, dollar sign), are treated literally and are used as delimiters between *InPattern* variables. For example, if the *InPattern* parameter is `$1.$2` and the remote file name is `mydata.dat`, the value of `$1` is `mydata` and the value of `$2` is `dat`.

The *OutPattern* parameter determines the resulting file name. The variables `$1` through `$9` are replaced by their values as derived from the *InPattern* parameter, and the variable `$0` is replaced by the original file name. Additionally, the sequence *Sequence1*,*Sequence2* is replaced by the value of *Sequence1*, if *Sequence1* is not null; otherwise, it is replaced by the value of *Sequence2*. For example, the subcommand:

```
nmap $1.$2.$3 $1,$2.$2,file
```

would yield `myfile.data` from `myfile.data` or `myfile.data.old`, `myfile.file` from `myfile`, and `myfile.myfile` from `.myfile`. Use the \ (backslash) symbol to prevent the special meanings of the $ (dollar sign), (left bracket), (right bracket), and , (comma) in the *OutPattern* parameter.

non-print

Synonym for the **form non-print** subcommand.

ntrans *InCharacters OutCharacters*

Turns the file-name character translation mechanism On and Off. If no parameters are specified, character translation is turned off. If parameters are specified, characters in source file names are translated for **mget** and **mput** subcommands and for **get** and **put** subcommands when the destination file name is not specified.

This subcommand is useful when the local and remote hosts use different file-naming conventions or practices. Character translation follows the pattern set by the *InCharacters* and *OutCharacters* parameter. Characters in a source file name matching characters in the *InCharacters* parameter are replaced by the corresponding characters in the *OutCharacters* parameter.

If the string specified by the *InCharacters* parameter is longer than the string specified by the *OutCharacters* parameter, the characters in the *InCharacters* parameter are deleted if they have no corresponding character in the *OutCharacters* parameter.

open *HostName Port*

Establishes a connection to the FTP server at the host specified by the *HostName* parameter. If the optional port number is specified, the **ftp** command attempts to connect to a server at that port. If the automatic login feature is set (that is, the **-n** flag was not specified on the command line), the **ftp** command attempts to log in the user to the FTP server.

You must also have a **$HOME/.netrc** file with the correct information in it and the correct permissions set. The **.netrc** file must be in your home directory.

passive

Toggles passive mode for file transfers. When a file transfer command (such as **get**, **mget**, **put**, or **mput**) is invoked with passive mode off, the **ftp** server opens a data connection back to the client. In passive mode, the client opens data connections to the server when sending or receiving data.

private

Sets the protection level to "private." At this level, data is integrity and confidentially protected.

prompt

Toggles interactive prompting. If interactive prompting is on (the default), the **ftp** command prompts for verification before retrieving, sending, or deleting multiple files during the **mget**, **mput**, and **mdelete** subcommands. Otherwise, the **ftp** command acts accordingly on all files specified.

protect

This command returns the current level of protection.

proxy *Subcommand*

Executes an **ftp** command on a secondary control connection. This subcommand allows the **ftp** command to connect simultaneously to two remote FTP servers for transferring files between the two servers. The first **proxy** subcommand should be an **open** subcommand to establish the secondary control connection. Enter the **proxy ?** subcommand to see the other **ftp** subcommands that are executable on the secondary connection.

The following subcommands behave differently when prefaced by the **proxy** subcommand:

- The **open** subcommand does not define new macros during the automatic login process.
- The **close** subcommand does not erase existing macro definitions.
- The **get** and **mget** subcommands transfer files from the host on the primary connection to the host on the secondary connection.
- The **put**, **mput**, and **append** subcommands transfer files from the host on the secondary connection to the host on the primary connection.
- The **restart** subcommand can be handled by the **proxy** command.
- The **status** subcommand displays accurate information.

File transfers require that the FTP server on the secondary connection must support the PASV (passive) instruction.

put *LocalFile RemoteFile*

Stores a local file on the remote host. If you do not specify the *RemoteFile* parameter, the **ftp** command uses the local file name to name the remote file, and the remote file name is altered by any settings made by the **ntrans** and **nmap** subcommands. The **ftp** command uses the current settings for the **type**, **form**, **mode**, and **struct** subcommands while transferring the files.

pwd

Displays the name of the current directory on the remote host.

quit

Closes the connection and exits the **ftp** command. Same as the **bye** subcommand.

quote *String*

Sends the string specified by the *String* parameter verbatim to the remote host. Execute the **remotehelp** or **quote help** subcommand to display a list of valid values for the *String* parameter.

> **Note:** "Quoting" commands that involve data transfers can produce unpredictable results.

record
Synonym for the **struct record** subcommand.

recv *RemoteFile LocalFile*
Copies the remote file to the local host. Same as the **get** subcommand.

reinitialize
Reinitializes an FTP session by flushing all I/O and allowing transfers to complete. Resets all defaults as if a user had just started an FTP session without logging in to a remote host.

remotehelp *Subcommand*
Requests help from the remote FTP server.

rename *FromName ToName*
Renames a file on the remote host.

reset
Clears the reply queue. This subcommand resynchronizes the command parsing.

restart get | put | append
Restarts a file transfer at the point where the last checkpoint was made. To run successfully, the subcommand must be the same as the aborted subcommand, including structure, type, and form. Valid arguments are **get**, **put**, and **append**.

rmdir *RemoteDirectory*
Removes the remote directory specified by the *RemoteDirectory* parameter at the remote host.

runique
(ReceiveUnique) Toggles the facility for creating unique file names for local destination files during **get** and **mget** subcommands. If this facility is Off (the default), the **ftp** command overwrites local files. Otherwise, if a local file has the same name as that specified for a local destination file, the **ftp** command modifies the specified name of the local destination file with `.1`. If a local file is already using the new name, the **ftp** command appends the postfix .2 to the specified name. If a local file is already using this second name, the **ftp** command continues incrementing the postfix until it either finds a unique file name or reaches .99 without finding a unique file name. If the **ftp** command cannot find a unique file name, the **ftp** command reports an error and the transfer does not take place. Note that the **runique** subcommand does not affect local file names generated from a shell command.

safe
Sets the protection level to "safe." At this level, data is integrity protected.

send *LocalFile RemoteFile*
Stores a local file on the remote host. Same as the **put** subcommand.

sendport
Toggles the use of FTP PORT instructions. By default, the **ftp** command uses a PORT instruction when establishing a connection for each data transfer. When the use of PORT instructions is disabled, the **ftp** command does not use PORT instructions for data transfers. The PORT instruction is useful when dealing with FTP servers that ignore PORT instructions while incorrectly indicating the instructions have been accepted.

site *Args*
Displays or sets the idle time-out period, displays or sets the file-creation umask, or changes the permissions of a file, using the **chmod** command. Possible values for the *Args* parameter are **umask** and **chmod**.

size *RemoteFile*
Displays the size in bytes of the remote file specified by the *RemoteFile* parameter.

status
Displays the current status of the **ftp** command as well as the status of the subcommands.

stream
Synonym for the **mode stream** subcommand.

struct file | record
Sets the data transfer structure type. Valid arguments are **file** and **record**.

file
Sets the data-transfer structure type to file.

record
Sets the data-transfer structure type to record.

sunique
(Send/Store Unique) Toggles the facility for creating unique file names for remote destination files during **put** and **mput** subcommands. If this facility is off (the default), the **ftp** command overwrites remote files. Otherwise, if a remote file has the same name as that specified for a remote destination file, the remote FTP server modifies the name of the remote destination file. Note that the remote server must support the STOU instruction.

system
Shows the type of operating system running on the remote machine.

telnet
Synonym for the **form telnet** subcommand.

tenex
Synonym for the **type tenex** subcommand.

trace
Toggles packet tracing.

type ascii | binary | ebcdic | image | local *M* **| tenex**
Sets the file-transfer type. Valid arguments are **ascii**, **binary**, **ebcdic**, **image**, **local** *M*, and **tenex**. If an argument is not specified, the current type is printed. The default type is **ascii**; the **binary** type can be more efficient than **ascii**.

ascii
Sets the file-transfer type to network ASCII. This type is the default. File transfer may be more efficient with binary-image transfer. See the **binary** argument for further information.

binary
Sets the file-transfer type to binary image. This type can be more efficient than an ASCII transfer.

ebcdic
Sets the file-transfer type to EBCDIC.

image
Sets the file-transfer type to binary image. This type can be more efficient than an ASCII transfer.

local *M*
Sets the file-transfer type to local. The *M* parameter defines the decimal number of bits per machine word. This parameter does not have a default.

tenex
Sets the file-transfer type to that needed for TENEX machines.

user *User Password Account*
Identifies the local user (*User*) to the remote FTP server. If the *Password* or *Account* parameter is not specified and the remote server requires it, the **ftp** command prompts for the password or account locally. If the *Account* parameter is required, the **ftp** command sends it to the remote server after the remote login process completes.

> **Note:** Unless automatic login is disabled by specifying the **-n** flag on the command line, the **ftp** command sends the *User*, *Password*, and *Account* parameters automatically for the initial connection to the remote server. You also need a **.netrc** file in your home directory in order to issue an automatic login.

verbose
Toggles verbose mode. When the verbose mode is on (the default), the **ftp** command displays all responses from the remote FTP server. Additionally, the **ftp** command displays statistics on all file transfers when the transfers complete.

Examples

1. To invoke the **ftp** command, log in to the system `canopus`, display local help information, display remote help information, display status, toggle the **bell**, **prompt**, **runique**, **trace**, and **verbose** subcommands, and then quit, enter:

```
$ ftp canopus
Connected to canopus.austin.century.com.
220 canopus.austin.century.com FTP server (Version 4.1 Sat Nov 23 12:52:09 CST 1991) ready.
Name (canopus:eric): dee
331 Password required for dee.
Password:
230 User dee logged in.
ftp> help
Commands may be abbreviated. Commands are:
!          delete          mdelete          proxy          runique
$          debug           mdir             sendport       send
account    dir             mget             put            size
append     disconnect      mkdir            pwd            status
ascii      form            mls              quit           struct
bell       get             mode             quote          sunique
binary     glob            modtime          recv           system
bye        hash            mput             remotehelp     tenex
case       help            nmap             rstatus        trace
cd         image           nlist            rhelp          type
cdup       lcd             ntrans           rename         user
close      ls              open             reset          verbose
cr         macdef          prompt           rmdir          ?
clear      private         protect          safe
ftp> remotehelp
214-The following commands are recognized(* =>'s unimplemented).
 USER   PORT   RETR   MSND*  ALLO   DELE   SITE*  XMKD   CDUP
 PASS   PASV   STOR   MSOM*  REST*  CWD    STAT*  RMD    XCUP
 ACCT*  TYPE   APPE   MSAM*  RNFR   XCWD   HELP   XRMD   STOU
 REIN*  STRU   MLFL*  MRSQ*  RNTO   LIST   NOOP   PWD
 QUIT   MODE   MAIL*  MRCP*  ABOR   NLST   MKD    XPWD
 AUTH   ADAT   PROT   PBSZ   MIC    ENC    CCC
214 Direct comments to ftp-bugs@canopus.austin.century.com.
ftp> status
Connected to canopus.austin.century.com.
No proxy connection.
Mode: stream; Type: ascii; Form: non-print; Structure: file
Verbose: on; Bell: off; Prompting: on; Globbing: on
Store unique: off; Receive unique: off
Case: off; CR stripping: on
Ntrans: off
Nmap: off
Hash mark printing: off; Use of PORT cmds: on
ftp> bell
Bell mode on.
ftp> prompt
Interactive mode off.
ftp> runique
Receive unique on.
ftp> trace
Packet tracing on.
ftp> verbose
Verbose mode off.
ftp> quit
$
```

2. To invoke the **ftp** command, log in to the system `canopus`, print the working directory, change the working directory, set the file transfer type to ASCII, send a local file to the remote host, change the working directory to the parent directory, and then quit, enter:

```
$ ftp canopus
Connected to canopus.austin.century.com.
220 canopus.austin.century.com FTP server (Version 4.1 Sat Nov 23 12:52:09 CST 1991) ready.
Name (canopus:eric): dee
331 Password required for dee.
Password:
230 User dee logged in.
ftp> pwd
257 "/home/dee" is current directory.
ftp> cd desktop
250 CWD command successful.
ftp> type ascii
200 Type set to A.
ftp> send typescript
200 PORT command successful.
150 Opening data connection for typescript (128.114.4.99,1412).
226 Transfer complete.
ftp> cdup
250 CWD command successful.
ftp> bye
221 Goodbye.
$
```

3. To invoke the **ftp** command with automatic logon (using the **.netrc** file), open a session with the system `canopus`, log in, change the working directory to the parent directory, print the working directory, list the contents of the current directory, delete a file, write a listing of the contents of the current directory to a local file, close the session, and then quit, enter:

```
$ ftp canopus
Connected to canopus.austin.century.com.
220 canopus.austin.century.com FTP server (Version 4.1 Sat Nov 23 12:52:09 CST 1991) ready.
331 Password required for dee.
230 User dee logged in.
ftp> cdup
250 CWD command successful.
ftp> pwd
257 "/home" is current directory.
ftp> dir
200 PORT command successful.
150 Opening data connection for /usr/bin/ls (128.114.4.99,1407)
(0 bytes).
total 104
drwxr-xr-x   2 system        32 Feb 23 17:55 bin
Drwxr-xr-x  26 rios        4000 May 30 17:18 bin1
drwxr-xr-x   2 system        32 Feb 23 17:55 books
drwxrwxrwx  18 rios        1152 Jun  5 13:41 dee
-r--r--r--   1 system      9452 May 17 12:21 filesystems
drwxr-xr-x   2 system        32 Feb 23 17:55 jim
drwxr-xr-x   5 system        80 Feb 23 17:55 krs
drwxrwxrwx   2 rios       16432 Feb 23 17:36 lost+found
```

```
-rwxr-xr-x   1 rios        3651 May 24 16:45 oldmail
drwxr-xr-x   2 system       256 Feb 23 17:55 pubserv
drwxrwxrwx   2 system       144 Feb 23 17:55 rein989
drwxr-xr-x   2 system       112 Feb 23 17:55 reinstall
226 Transfer complete.
ftp> delete oldmail
250 DELE command successful.
ftp> mdir /home/dee/bin binlist
output to local-file: binlist? y
200 PORT command successful.
150 Opening data connection for /usr/bin/ls (128.114.4.99,1408) (0 bytes).
226 Transfer complete.
ftp> close
221 Goodbye.
ftp> quit
$
```

Files

/usr/samples/tcpip/netrc	Contains the sample **.netrc** file.
/etc/syslog.conf	Contains configuration information for the **syslogd** daemon.

Related Information

The **stty** command.

grep Command

Purpose

Searches for a pattern in a file.

Syntax

grep -E | **-F -i -h -H -L -r** | **-R -s -v -w -x -y -b -n** | **-c** | **-l** | **-q -p** *Separator* { **-e** *PatternList* ... **-f** *PatternFile* ... | *PatternList* ... } *File* ...

Description

The **grep** command searches for the pattern specified by the *Pattern* parameter and writes each matching line to standard output. The patterns are limited regular expressions in the style of the **ed** or **egrep** command. The **grep** command uses a compact non-deterministic algorithm.

The **grep** command displays the name of the file containing the matched line if you specify more than one name in the *File* parameter. Characters with special meaning to the shell ($, *, , |, ^, (,), \ ) must be in quotation marks when they appear in the *Pattern* parameter. When the *Pattern* parameter is not a simple string, you usually must enclose the entire pattern in single quotation marks. In an expression such as `a-z`, the - (minus sign) cml specifies a range, according to the current collating sequence. A collating sequence may define equivalence classes for use in character ranges. If no files are specified, **grep** assumes standard input.

Notes:

1. Do not run the **grep** command on a special file because it produces unpredictable results.
2. Input lines should not contain the NULL character.
3. Input files should end with the newline character.
4. The newline character will not be matched by the regular expressions.
5. Although some flags can be specified simultaneously, some flags override others. For example, the **-l** option takes precedence over all other flags. And if you specify both the **-E** and **-F** flags, the last one specified takes priority.

Flags

-b Precedes each line by the block number on which it was found. Use this flag to help find disk block numbers by context. The **-b** flag cannot be used with input from stdin or pipes.

-c Displays only a count of matching lines.

-E Treats each pattern specified as an extended regular expression (ERE). A NULL value for the ERE matches every line.

-e *PatternList* Specifies one or more search patterns. This works like a simple pattern but is useful when the pattern begins with a - (minus). Patterns should be separated by a new-line character. A NULL pattern can be specified by two adjacent new-line characters or a quotation mark followed by a new-line character ("\n). Each pattern is treated like a basic regular expression (BRE) unless the **-E** or **-F** flag is also specified. Multiple **-e** and **-f** flags are accepted by **grep**. All of the specified patterns are used when matching lines, but the order of evaluation is unspecified.

-F Treats each specified pattern as a string instead of a regular expression. A NULL string matches every line.

-f *PatternFile* Specifies a file containing search patterns. Each pattern should be separated by a new-line character, and an empty line is considered a NULL pattern. Each pattern is treated like a basic regular expression (BRE), unless the **-E** or **-F** flag is also specified.

-h Prevents the name of the file containing the matching line from being appended to that line. Suppresses file names when multiple files are specified.

-H If the **-r** or **-R** option is specified and a symbolic link referencing a file of type directory is specified on the command line, **grep** will search the files of the directory referenced by the symbolic link and all the files in the file hierarchy below it.

-i Ignores the case (uppercase or lowercase) of letters when making comparisons.

-l Lists just the names of files (once) which contain matching lines. Each file name is separated by a new-line character. If standard input is searched, a path name of `(StandardInput)` is returned. The **-l** flag with any combination of the **-c** and **-n** flags behaves like the **-l** flag only.

-L If the **-r** or **-R** option is specified and a symbolic link referencing a file of type directory is specified on the command line or encountered during the traversal of a file hierarchy, **grep** shall search the files of the directory referenced by the symbolic link and all the files in the file hierarchy below it. If both **-H** and **-L** are specified, the last option specified on the command line takes effect.

-n Precedes each line with the relative line number in the file. Each file starts at line 1, and the line counter is reset for each file processed.

-p*Separator* Displays the entire paragraph containing matched lines. Paragraphs are delimited by paragraph separators, as specified by the *Separator* parameter, which are patterns in the same form as the search pattern. Lines containing the paragraph separators are used only as separators; they are never included in the output. The default paragraph separator is a blank line.

-q Suppresses all writing to standard output, regardless of matching lines. Exits with a zero status if an input line is selected. The **-q** flag with any combination of the **-c**, **-l** and **-n** flags behaves like the **-q** flag only.

-r Searches directories recursively. By default, links to directories are followed.

-R Searches directories recursively. By default, links to directories are not followed.

-s Suppresses error messages ordinarily written for nonexistent or unreadable files. Other error messages are not suppressed.

-v Displays all lines not matching the specified pattern.

-w Does a word search.

-x Displays lines that match the specified pattern exactly with no additional characters.

-y Ignores the case of letters when making comparisons.

PatternList Specifies one or more patterns to be used during the search. The patterns are treated as if they were specified using the **-e** flag.

File Specifies a name of a file to be searched for patterns. If no *File* variable is given, the standard input is used.

Exit Status

This command returns the following exit values:

0 A match was found.

1 No match was found.

>1 A syntax error was found or a file was inaccessible (even if matches were found).

Examples

1. To use a pattern that contains some of the pattern-matching characters *, ^, ?, , , \(, \), \{, and \}, enter:

   ```
   grep  "^a-zA-Z"  pgm.s
   ```

   This displays every line in `pgm.s` whose first character is a letter.
2. To display all lines that do not match a pattern, enter: `grep   -v  "^#" pgm.s`
   This displays every line in `pgm.s` whose first character is not a `#` (pound sign).
3. To display all lines in the `file1` file that match either the `abc` or `xyz` string, enter:
   `grep   -E   "abc|xyz"  file1`
4. To search for a `$` (dollar sign) in the file named `test2`, enter:

   ```
   grep \\$ test2
   ```

The `\\` (double backslash) characters are necessary in order to force the shell to pass a `\$` (single backslash, dollar sign) to the **grep** command. The `\` (single backslash) character tells the **grep** command to treat the following character (in this example the `$`) as a literal character rather than an expression character. Use the **fgrep** command to avoid the necessity of using escape characters such as the backslash.

5. To search recursively through /**tmp** to find files which have the word `IBM` without recursing through links pointing to directories, type:

   ```
   grep -R IBM /tmp
   ```

   OR

   ```
   grep -r -H IBM /tmp
   ```
6. To search recursively through /**tmp** to find files which have the word `IBM` and recurse through links as well, type:

   ```
   grep -r IBM /tmp
   ```

   OR

   ```
   grep -R -L IBM /tmp
   ```

Files

/usr/bin/grep	Contains the **grep** command.

head Command

Purpose

Displays the first few lines of a file.

Syntax

head **-***Count* | **-c** *Number* | **-n** *Number* *File* ...

Description

The **head** command writes to standard output a specified number of lines or bytes of each of the specified files, or of the standard input. If no flag is specified with the **head** command, the first 10 lines are displayed by default. The *File* parameter specifies the names of the input files. An input file must be a text file. When more than one file is specified, the start of each file will look like the following:

```
==> filename <==
```

To display a set of short files, identifying each one, enter:

```
example% head -9999 filename1 filename2...
```

Flags

-*Count*	Specifies the number of lines from the beginning of each specified file to be displayed. The *Count* variable must be a positive decimal integer. This flag is equivalent to the **-n** *Number* flag, but should not be used if portability is a consideration.
-c *Number*	Specifies the number of bytes to display. The *Number* variable must be a positive decimal integer.
-n *Number*	Specifies the number of lines from the beginning of each specified file to be displayed. The *number* variable must be a positive decimal integer. This flag is equivalent to the **-***Count* flag.

Exit Status

This command returns the following exit values:

0	Successful completion.
>0	An error occurred.

Examples

To display the first five lines of the `Test` file, enter:

```
head -5 Test
```

OR

```
head -n 5 Test
```

Related Information

The **tail** command.

hostmap Command

Purpose

Directly manipulates address-mapping entries in the system configuration database.

Syntax

To Add an Address-to-Host Name Mapping

hostmap -addr *IPAddress* **-host** *HostName*...

To Delete an Address-to-Host Name Mapping

hostmap -rm *IPAddress*

To Show all Address-to-Host Name Mappings

hostmap -ls

Description

The **hostmap** low-level command adds, deletes, or lists address-mapping entries in the system configuration database. Entries in the database are used to map an Internet Protocol (IP) address (local or remote) to its equivalent host names.

An Internet Protocol (IP) address of a given local or remote host may be associated with one or more host names. Represent an IP address in dotted decimal format. Represent a host name as a string with a maximum length of 255 characters, and do not use any blank characters.

Notes:

1. Valid host names or alias host names must contain at least one alphabetic character. If you choose to specify a host name or alias that begins with an x followed by any hexadecimal digit (0-f), the host name or alias must also contain at least one additional letter that cannot be expressed as a hexadecimal digit. The system interprets a leading x followed by a hexadecimal digit as the base 16 representation of an address unless there is at least one character in the host name or alias that is not a hexadecimal digit. Thus, xdeer would be a valid host name, whereas xdee would not.
2. The **hostmap** command does not recognize the following addresses: .08, .008, .09, and .009. Addresses with leading zeros are interpreted as octal, and numerals in octal cannot contain 8s or 9s.

Flags

-addr *IPAddress*	Adds an IP address-to-host name mapping entry for the given Internet Protocol address in the database. Specify the host names with the **-host** flag.
-host *HostName*...	Specifies a list of host names. Entries in the list should be separated by blanks.
-ls	Shows all entries in the database.
-rm *IPAddress*	Deletes the IP address-to-host name mapping entry in the database that corresponds to the given address specified by the IPAddress variable.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To add an entry in the database associating an address with a series of host names, enter the command in the following format:

```
hostmap  -addr 192.100.201.7  -host alpha bravo charlie
```

The IP address 192.100.201.7 is specified as the address of the host that has a primary host name of alpha with synonyms of **bravo** and **charlie**. **Note:** If you attempt to use .08, .008, .09, or .009 in an address to add, you will get an error message that states "IP Address already exists," although the address is not in the database.

2. To list all entries in the database, enter the command in the following format:

```
hostmap -ls
```

Related Information

The **hostname** command, and the **mktcpip** command.

hostname Command

Purpose

Sets or displays the name of the current host system.

Syntax

hostname *HostName*

Description

The **hostname** command sets or displays the name of the current host system.

Parameters

Hostname	Sets the primary name of the host.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To set the hostname to **rotterdam**, type:

```
hostname rotterdam
```

Related Information

The **mktcpip** command, the **startnetsvc** command, the **stopnetsvc** command, the **cfglnagg** command, the **netstat** command, the **entstat** command, the **cfgnamesrv** command, the **hostmap** command, the **traceroute** command, the **ping** command, the **optimizenet** command.

importvg Command

Purpose

Imports a new volume group definition from a set of physical volumes.

Syntax

importvg **-vg** *VolumeGroup PhysicalVolume*

Description

The **importvg** command makes the previously exported volume group known to the system. The *PhysicalVolume* parameter specifies only one physical volume to identify the volume group; any remaining physical volumes (those belonging to the same volume group) are found by the **importvg** command and included in the import. An imported volume group is automatically activated. When a volume group with file systems is imported, the /**etc**/**filesystems** file is updated with values for the new logical volumes and mount points.

After importing the volume group, you must run the **fsck** command before the file systems can be mounted. Care should be taken to avoid using mount point longer than 128 characters as the mount point information would be missing from the LVCB (logical volume control block) if it is longer than 128 characters. In this case, the **importvg** command will not be able to update the /**etc**/**filesystems** file with the stanza for the newly imported logical volume.

The **importvg** command changes the name of a logical volume if the name already exists in the system. It prints a message and the new name to standard error, and updates the /**etc**/**filesystems** file to include the new logical volume name.

Flags

-vg *VolumeGroup*	Specifies the name to use for the new volume group. If this flag is not used, the system automatically generates a new name. The volume group name can only contain the following characters: "A" through "Z," "a" through "z," "0" through "9," or "_" (the underscore), "-" (the minus sign), or "." (the period). All other characters are considered invalid.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To import the volume group **bkvg** from physical volume **hdisk07**, type:

   ```
   importvg -vg bkvg hdisk07
   ```

   The volume group **bkvg** is made known to the system.

Restrictions

Mount points cannot be longer than 128 characters.

Related Information

The **activatevg** command, the **chvg** commands, the **deactivatevg** command, the **exportvg** command, the **extendvg** command, the **lsvg** command, the **mirrorios** command, the **mkvg** command, the **syncvg** command, and the **unmirrorios** command.

installios Command

Purpose

Installs the Virtual I/O Server. This command is run from the HMC.

Syntax

installios **-p** *partition_name* **-i** *ipaddr* or *hostname* **-S** *subnet_mask* **-g** *gateway* **-d** *path* **-s** *system_name* **-m** *mac_address* **-r** *profile* **-n** **-P** *speed* **-D** *duplex* **-l** *language* | **-u**

Description

The **installios** command installs the Virtual I/O Server. It must be run from the HMC. All of the flags are optional. If no flags are provided, the **installios** wizard will be invoked and the user will be prompted to interactively enter the information contained in the flags.

Flags

-s Specifies the managed system. The name of the managed system maintained by the HMC. This name must match the name shown on the HMC, not a host name.

-p Specifies the partition name. The name of the LPAR that will be installed with Virtual I/O Server operating system. This partition must be of type `Virtual I/O Server` and the name given for it must match the name shown on the HMC, not a host name.

-r Specifies the profile name. The name of the profile that contains the hardware resources being installed to.

-d Specifies the path to installation images. Either **/dev/cdrom** or the path to a system backup of the Virtual I/O Server created by the **backupios** command. The path may also specify a remote location mountable by NFS such as **hostname:/path_to_backup**

-i Specifies the client IP address. The IP address with which the client's network interface will be configured for network installation of the Virtual I/O Server operating system.

-S Specifies the client subnet mask. The subnet mask with which the client's network interface will be configured for network installation of the Virtual I/O Server operating system.

-g Specifies the client gateway. The default gateway that the client will use during network installation of the Virtual I/O Server operating system.

-m Specifies the client MAC address. The MAC address of the client network interface through which the network installation of the Virtual I/O Server will take place.

-P Specifies speed (optional) The communication speed with which to configure the client's network interface. This value can be 10, 100, or 1000, and is 100 by default if this flag is not specified.

-D Specifies duplex (optional). The duplex setting with which to configure the client's network interface. This value can be `full` or `half` and is set to `full` by default if this flag is not specified.

-n Specifies not to configure the client's network interface (optional): If this flag is specified, then the client's network interface will not be configured with the IP settings specified in the flags given to this command after the installation has completed.

-l Specifies language (optional): The language in which the license agreement will be displayed before the installation. Upon viewing the license, a prompt will be shown asking if the license is to be accepted. If the prompt is answered with `y`, then the installation will proceed and the Virtual I/O Server license will be automatically accepted after the installation. If the prompt is answered with `n`, then the **installios** command will exit and the installation will not proceed. If this flag is not specified, then the installation will proceed, but the Virtual I/O Server will not be usable until the license is manually accepted after the installation.

-u Unconfigure **installios** (optional). Will manually unconfigure the **installios** installation resources. This flag is only needed if a problem occurs during the installation and **installios** does not automatically unconfigure itself.

invscout Command

Purpose

Surveys the host system for currently installed microcode or Vital Product Data (VPD).

Syntax

invscout **-vpd** | **-report** **-model** *Type-Model* **-serial** *SerialNumber* **-version**

invscout **-rpm** *rpmPackage rpmOption* ...

invscout **-install** *Device* **-file** *FileName*

Description

The **invscout** command executes one instance of the stand alone version of the Inventory Scout process. The **invscout** command starts the server daemon side of a client-server version. The Inventory scout process supports two survey types:

- Microcode Survey
- Vital Product Data (VPD) Survey

Flags

-file *FileName*	Specifies the location of the upgrade.
-install *Device*	Installs microcode for a given device.
-model *Type-Model*	Machine type and model. For a VPD survey, allows input of the host platform machine type and model for hosts that use this information.
-report	For a Microcode Survey, sends a copy of the formatted text report file to the screen from which the command was invoked. This flag is ignored if the **-vpd** flag is used.
-rpm *rpmPackage*	The Red Hat Package Manager is used to install, upgrade, and remove rpm packages. RPM options: -i Installs a new package. -e Uninstalls a package. -qp Query for an uninstalled package. --force Used to install a package even if it is already installed. --ignoreos Allows installation of a package even if the operating system of the host and binary RPM are different.
-serial *SerialNumber*	Serial number. For a VPD survey, allows input of the host serial number for hosts that use this information.
-version	Displays the versions of this command and of the logic database currently in use.
-vpd	Sets the survey or concatenation type to VPD (the default is Microcode).

Exit Status

The following exit values are returned:

0	Successful completion.
>0	An error occurred.

Examples

1. To generate report on microcode levels of all the devices, type:

   ```
   invscout -report
   ```
2. To get the vpd survey of the partition. type:

   ```
   invscout -vpd
   ```

ioslevel Command

Purpose

Reports the latest installed maintenance level of the system.

Syntax

ioslevel

Description

The **ioslevel** command displays the Virtual I/O Server level.

Exit Status

See Virtual I/O Server command exit status

Related Information

The **lssw** command, the **updateios** command, the **remote_management** command, the **oem_setup_env** command, and the **oem_platform_level** command.

ldfware Command

Purpose

Loads the system's flash EPROM with the specified file, which must contain a valid binary flash EPROM image, and then reboots the system.

Syntax

ldfware **-device** *Device* **-file** *filename*

ldfware **-commit**

Description

Periodically, you need to install fixes for your server firmware. If you do not use an HMC to manage your server, you must get your fixes through your operating system. If your server is not connected to the Internet, you must obtain optical media that contains the server firmware fix and install the server firmware fix from the optical media. The flash update image file is copied to the file system from diskette. The user needs to provide the image on a diskette if the user does not have access to remote file systems or any other files that are on the system. If enough space is not available, an error is reported stating additional system memory is needed. After the file is copied, a warning screen asks for confirmation to continue the update flash. Continuing the update flash reboots the system. The current flash image is not saved.

After you download and install a firmware fix, the fix is temporarily installed until you install it permanently. You might want to use the new level of firmware for a period of time to verify that it works correctly. When you are sure that the new level of firmware works correctly, you can permanently install the firmware fix. Be aware that if you install the firmware fix permanently (copy the temporary firmware level from the temporary side to the permanent side, so that the temporary and permanent sides contain the same level of firmware), you cannot return to the level that was previously on the permanent side. To install it permanently, use the **-commit** flag.

Attention: The **ldfware** command reboots the entire system. Do not use this command if more than one user is signed onto the system

Flags

-commit	Commits the temporary image when booted from the temporary image. This effectively causes the permanent image to be copied over by the temporary image.
-device *Device*	Specifies that the flash update image file is on diskette.
-file *FileName*	Flash update image file source follows this flag.

Exit Status

The following exit codes are returned:

0	Completed successfully.
1	Command terminated due to an error.
2	Command was aborted by the user.
3	Command was aborted by the user using the F10 key.

license Command

Purpose

View and accept the license agreement.

Syntax

license { **-view** **-accept** } **-lang** *Name*

license **-ls**

Description

The **license** command is used to view and accept the Virtual I/O Server license agreement. If no flags are specified, the current status of the license agreement is displayed. If the license has been accepted, the date and time of acceptance is displayed.

Flags

-accept	Accepts the license agreement
-lang *Name*	Specifies the language-territory (locale name) that the license will be displayed in. The default is en_US.
-ls	Lists available languages.
-view	Displays the Virtual I/O Server license agreement.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To view the license in the en_US locale, type:

   ```
   license -view
   ```
2. To accept the license in the fr_FR locale, type:

   ```
   license -accept -lang fr_FR
   ```
3. To view if the license has been accepted, type:

   ```
   license
   ```

Related Information

The **chlang** command.

loginmsg Command

Purpose

Modifies the Virtual I/O Server partition's login herald.

Syntax

loginmsg { **-reset** | "*Herald string*" }

Description

The **loginmsg** command sets the Virtual I/O Server partition's login herald.

Flags

-reset	Reset the login message back to the system default.

Examples

1. To set the login herald to `Welcome` followed by `login:` on a separate line, type:

   ```
   loginmsg "Welcome\nlogin:"
   ```
2. To reset the login herald back to the system default, type:

   ```
   loginmsg -reset
   ```

IVM lpcfgop Command

Purpose

Perform a partition configuration image operation. This command is valid only in an Integrated Virtualization Manager environment.

Syntax

To clear logical partition configuration data at next system restart:

lpcfgop -o clear **--force -m** *ManagedSystem*

To disable logical partition configuration data at next system restart:

lpcfgop -o disable **--force -m** *ManagedSystem*

To dump logical partition configuration data to a file:

lpcfgop -o dump **-f** *DumpFile* **-m** *ManagedSystem*

To enable logical partition configuration data at next system restart:

lpcfgop -o enable **-m** *ManagedSystem*

Description

The **lpcfgop** command performs a partition configuration image operation such as clearing, disabling, enabling, or dumping the logical partition configuration data.

Flags

-o *Operation*	The type of operations: • clear: Marks the logical partition configuration data to be cleared when the managed system is restarted. • disable: Marks the logical partition configuration data to be disabled when the managed system is restarted. This will temporarily place the system back into the factory configuration partition mode. • dump: Dumps all logical partition configuration data from managed system firmware to a file. This data may be used by service. Use the **bkprofdata** command to create backups that can be restored. • enable: Enables partition configuration data so that it will not be disabled or cleared when the managed system is restarted.
-f *DumpFile*	The name of the file to write the dump to in the current working directory. If not specified, the default file will be **/var/adm/lpm/lparConfig.dump**
--force	Skip confirmation prompts for clear and disable operations.
-m *ManagedSystem*	The name of the managed system. This attribute is optional because there is only one system to manage. The name may either be the user-defined name for the managed system, or be in the form tttt-mmm*sssssss, where tttt is the machine type, mmm is the model, and sssssss is the serial number of the managed system.

Exit Status

This command has a return code of zero on success.

Security

This command is not accessible by padmin.

Examples

1. Clear the partition configuration data (confirmation will be required) during the next reboot:

   ```
   lpcfgop -o clear
   ```
2. Dump the logical partition configuration data to lparData.dump:

   ```
   lpcfgop -o dump -f lparData.dump
   ```

Related Information

The **bkprofdata** command and the **rstprofdata** command.

ls Command

Purpose

Displays the contents of a directory.

Syntax

To Display Contents of Directory or Name of File

ls -1 -A -C -F -H | **-L -N -R -X -a -b -c -d -e -f -g -i -l -m -n -o -p -q -r -s -t -u -U -x** *File* ...

To Display Contents of Directory

ls -f -C -d -i -m -s -X -x -1 -U *Directory* ...

Description

The **ls** command writes to standard output the contents of each specified *Directory* parameter or the name of each specified *File* parameter, along with any other information you ask for with the flags. If you do not specify a *File* or *Directory* parameter, the **ls** command displays the contents of the current directory.

Specifying more than one of the options in the mutually exclusive pairs is not considered an error. The last option specified in each pair determines the output format.

By default, the **ls** command displays all information in alphabetic order by file name. The collating sequence is determined by the **LANG** or **LC_COLLATE** environment variable.

When the **ls** command displays the contents of a directory, it does not show entries for files whose names begin with a **.** (dot) unless you use the **-a** or **-A** flag. If the command is executed by root, it uses the **-A** flag by default.

There are three main ways to format the output:

- List one entry per line.
- List entries in multiple columns by specifying either the **-C** or **-x** flag. The **-C** flag is the default format when output is to a tty. The **ls** command displays single column output if file or directory names are too long.
- List entries in a comma-separated series by specifying the **-m** flag.

To determine the number of character positions in the output line, the **ls** command uses the **COLUMNS** environment variable. If this variable is not set, the command gets the current column value of the display. If the **ls** command cannot determine the number of character positions by either of these methods, it uses a default value of 80.

The mode displayed with the **-U** flag is the same as with the **-l** flag, except for the addition of an 11th character interpreted as follows:

E Indicates a file has extended attributes (EA) information. The EA of a file is displayed by using the **getea** command.

- Indicates a file does not have extended attributes information.

The mode displayed with the **-e** and **-l** flags is interpreted as follows:

If the first character is:

d The entry is a directory.

b The entry is a block special file.

c The entry is a character special file.

l The entry is a symbolic link, and either the **-N** flag was specified or the symbolic link did not point to an existing file.

p The entry is a first-in,first-out (FIFO) special file.

s The entry is a local socket.

- The entry is an ordinary file.

The next nine characters are divided into three sets of three characters each. The first set of three characters show the owner's permission. The next set of three characters show the permission of the other users in the group. The last set of three characters shows the permission of anyone else with access to the file. The three characters in each set indicate, respectively, read, write, and execute permission of the file. Execute permission of a directory lets you search a directory for a specified file.

Permissions are indicated as follows:

r	Read
w	Write (edit)
x	Execute (search)
-	Corresponding permission not granted

The group-execute permission character is **s** if the file has set-group-ID mode. The user-execute permission character is **s** if the file has set-user-ID mode. The last character of the mode (usually **x** or **-**) is **T** if the 01000 (octal) bit of the mode is set (see the **chmod** command for the meaning of this mode). The indications of set-ID and 01000 bit of the mode are capitalized (**S** and **T**, respectively) if the corresponding execute permission is not set. The mode **t** indicates that the sticky bit is on for the file or the directory.

The mode displayed with the **-e** flag is the same as with the **-l** flag, except for the addition of an 11th character interpreted as follows:

+ Indicates a file has extended security information. For example, the file may have extended **ACL**, **TCB**, or **TP** attributes in the mode.

- Indicates a file does not have extended security information.

When the size of the files in a directory are listed, the **ls** command displays a total count of blocks, including indirect blocks.

Flags

-A Lists all entries except **.** (dot) and **..** (dot-dot).

-a Lists all entries in the directory, including the entries that begin with a **.** (dot).

-b Displays nonprintable characters in an octal (\n*nn*) notation.

-c Uses the time of last modification of the i-node for either sorting (when used with the **-t** flag) or for displaying (when used with the **-l** flag). This flag must be used with either the **-t** or **-l** flag, or both.

-C Sorts output vertically in a multicolumn format. This is the default method when output is to a terminal.

-d Displays only the information for the directory named. Directories are treated like files, which is helpful when using the **-l** flag to get the status of a directory.

-e Displays the mode (including security information), number of links, owner, group, size (in bytes), time of last modification, and name of each file. If the file is a special file, the size field contains the major and minor device numbers. If the file is a symbolic link, the path name of the linked-to file is printed preceded by a -> (minus, greater than) sign. The attributes of the symbolic link are displayed.

-E Lists space reservation, fixed extent size, and extent allocation flag information for a file. **-l** must be specified with this flag.

-f Lists the name in each slot for each directory specified in the *Directory* parameter. This flag turns off the **-l**, **-t**, **-s**, and **-r** flags, and turns on the **-a** flag. The order of the listing is the order in which entries appear in the directory.

-F Puts a / (slash) after each file name if the file is a directory, an * (asterisk) if the file can be executed, an = (equal sign) if the file is a socket, a | (pipe) sign if the file is a FIFO, and an @ for a symbolic link.

Note: Symbolic links are displayed with the trailing -> only if the **-N** flag is used or if the link points to a nonexistent file. Otherwise, information about the target file is displayed. You can also invoke this option by entering the **ls -f** command.

-g Displays the same information as the **-l** flag, except the **-g** flag suppresses display of the owner and symbolic link information.

-H If a symbolic link referencing a file of type directory is specified on the command line, the **ls** command shall evaluate the file information and file type to be those of the file referenced by the link, and not the link itself; however, the **ls** command shall write the name of the link itself and not the file referenced by the link.

-i Displays the i-node number in the first column of the report for each file.

-L Lists the file or directory contents that the link references. This is the default action. Symbolic links are followed. If the **-l** option is used, the **-N** option becomes the default, and no symbolic links are followed. When the **-l** option is used, only the **-L** option can override the **-N** default.

-l (Lower case L) Displays the mode, number of links, owner, group, size (in bytes), and time of last modification for each file. If the file is a special file, the size field contains the major and minor device numbers. If the time of last modification is greater than six months ago, the time field is shown in the format **month date year** where as files modified within six months the time field is shown as **month date time** format.

If the file is a symbolic link, the path name of the linked-to file is printed preceded by a ->. The attributes of the symbolic link are displayed. The **-n**, **-g**, and **-o** flag overrides the **-l** flag.

Notes:

1. A symbolically linked file is followed by an arrow and the contents of the symbolic link.
2. The performance of the **ls** command when used with the **-l** option can be improved by executing the **mkpasswd** command. This is helpful when a directory contains files owned by different users, such as the /**tmp** directory.

-m Uses stream output format (a comma-separated series).

-n Displays the same information as the **-l** flag, except that the **-n** flag displays the user and the group IDs instead of the user and group names.

-N Does not follow symbolic links when determining the status of a file.

Note: If both the **-L** and **-N** options are used, the last one will dominate. Also, any time a symbolic link is given that includes a / (slash) as the final character, the link will automatically be followed regardless of any options used.

-o Displays the same information as the **-l** flag, except the **-o** flag suppresses display of the group and symbolic link information.

-p Puts a slash after each file name if that file is a directory. This is useful when you pipe the output of the **ls** command to the **pr** command, as follows:

```
ls  -p |  pr  -5  -t  -w80
```

-q Displays nonprintable characters in file names as a `?` (question mark).

-r Reverses the order of the sort, giving reverse alphabetic or the oldest first, as appropriate.

-R Lists all subdirectories recursively.

-s Gives size in kilobytes (including indirect blocks) for each entry.

-t Sorts by time of last modification (latest first) instead of by name.

-U Displays similar information as the **-l** flag. Displays the mode (including security information and named extended attribute information), number of links, owner, group, size (in bytes), time of last modification, and name of each file. If the file is a special file, the size field contains the major and minor device numbers. If the file is a symbolic link, the path name of the linked-to file is printed preceded by a **->** (minus, greater than) sign. The attributes of the symbolic link are displayed.

-u Uses the time of the last access, instead of the time of the last modification, for either sorting (when used with the **-t** flag) or for displaying (when used with the **-l** flag). This flag has no effect if it is not used with either the **-t** or **-l** flag, or both.

-x Sorts output horizontally in a multi-column format.

-X Prints long user names when used with other flags that display user names. The upper limit is determined by the **max_logname** ODM attribute in the PdAt and CuAt object classes. If a user name is greater than the **max_logname** attribute, it will be truncated to the number of characters as specified by the **max_logname** attribute, less one character.

-1 Forces output into one-entry-per-line format. This is the default when the output is not directed to a terminal.

Exit Status

This command returns the following exit values:

0	All files were written successfully.
>0	An error occurred.

Examples

1. To list all files in the current directory, type:

   ```
   ls -a
   ```

   This lists all files, including **.** (dot), **..** (dot-dot), and other files with names beginning with a dot.
2. To display detailed information, type:

   ```
   ls -l chap1 .profile
   ```

   This displays a long listing with detailed information about `chap1` and `.profile`.
3. To display detailed information about a directory, type:

   ```
   ls -d -l .manual  manual/chap1
   ```

   This displays a long listing for the directories `.` and `manual`, and for the file `manual/chap1`. Without the **-d** flag, this would list the files in the `.` and `manual` directories instead of the detailed information about the directories themselves.
4. To list the files in order of modification time, type:

   ```
   ls -l -t
   ```

   This displays a long listing of the files that were modified most recently, followed by the older files.
5. To display detailed information with expanded user and group name, type:

   ```
   ls  -lX .profile
   ```

   This displays a long listing with detailed information about **.profile**.
6. To display about whether extended attributes are set on the files in current directory, type:

   ```
   ls -U
   ```

   Example output:

   ```
     -rwSr-x---+   1 root     system           28 Apr 29 03:23 only_aixc
     -rwSr-x---E   1 root     system            4 Apr 29 03:23 only_aixc_ea
     -rw-r--r--E   1 root     system            4 Apr 29 03:23 only_ea
     ----------+   1 root     system          265 Apr 29 03:23 only_nfs4
     ----------E   1 root     system           64 Apr 29 03:23 only_nfs4_ea
     -rw-r--r---   1 root     system            4 Apr 29 03:23 only_regular
   ```

Files

/usr/bin/ls	Contains the **ls** command.
/etc/passwd	Contains user IDs.
/etc/group	Contains group IDs.
/usr/share/lib/terminfo/*	Contains terminal information.

Related Information

The **chmod** command.

lsdev Command

Purpose

Displays Virtual I/O Server devices and their characteristics.

Syntax

To list devices

lsdev **-type** *DeviceType*... **-virtual** **-field** *FieldName*... **-fmt** *Delimiter* **-state** *State*

To display information about a specific device:

lsdev { **-dev** *DeviceName* | **-plc** *PhysicalLocationCode* } **-child** **-field** *FieldName*... **-fmt** *Delimiter*

lsdev { **-dev** *DeviceName* | **-plc** *PhysicalLocationCode* } **-attr** *Attribute* | **-range** *Attribute* | **-slot** | **-vpd** | **-parent**

lsdev **-vpd**

lsdev **-slots**

Description

The **lsdev** command displays information about devices in the Virtual I/O Server. If no flags are specified, a list of all devices, both physical and virtual, in the Virtual I/O Server is displayed. To list devices, both physical and virtual, of a specific type use the **-type** *DeviceType* flag. Use the **-virtual** flag to list only virtual devices. Combining both the **-type** and **-virtual** flags will list the virtual devices of the specified type.

To display information about a specific device, use the **-dev** *DeviceName* or **-plc** *PhysicalLocationCode*. Use either the **-child**, **-parent**, **-attr**, **-range**, **-slot**, or **-vpd** flag to specify what type of information is displayed. If none of these flags are used, the name, status, and description of the device will be displayed.

Using the **-vpd** flag, without specifying a device, displays platform-specific information for all devices.

If the **-fmt** *Delimiter* flag is specified the **lsdev** command returns all output in a delimiter separated format. The **-state** *State* flag limits the output to devices in the indicated state. The **-slots** flag produces a list of I/O slot information for built-in adapters that are not hot-pluggable but can have Dynamic LPAR operations performed on them.

The **lsdev** output will be truncated to 80 characters per line, unless the **-fmt** flag is used. If the **-fmt** flag is used, the output lines will not be truncated and line wrap may occur, based on what the terminal window line length is set to.

Flags

-attr *Attribute*

Displays information about attributes of a given device. If an attribute is specified, its current value is displayed. If no attribute is specified, the following information is displayed about all attributes for the give device:

attribute
: Attribute name

value
: Current value of the attribute

description
: Description of the attribute

user
: Whether attribute can be set by the user (TRUE/FALSE)

-child

Displays the name, status, physical location code, and description for each child of the specified device (**-dev** *DeviceName* or **-plc** *PhysicalLocationCode*).

-dev *DeviceName*
Specifies the device logical name for the device for which information is listed. This flag cannot be used with the **-plc** flag.

-field *FieldName*
Specifies the list of fields to be displayed. The following fields are supported:

name
    Device name
status
    Device status
physloc
    Physical location code
description
    Description of the device
parent

Note: The **-field** flag cannot be combined with the **-parent**, **-attr**, **-range**, **-slot**, or **-vpd** flags.

-fmt *Delimiter*
Specifies a delimiter character to separate output fields.

-parent
Displays the name, status, physical location code, and description of the parent device of the given device (**-dev** *DeviceName* or **-plc** *PhysicalLocationCode*).

-plc *PhysicalLocationCode*
Specifies the device physical location code for the device for which information is listed. This flag cannot be used with the **-dev** flag.

-range *Attribute*
Displays the allowed values for the specified attribute.

-slot
Displays the slot, description, and device name of the specified device (**-dev** *DeviceName* or **-plc** *PhysicalLocationCode*). The device must be in a PCI hot swappable slot.

-slots
Displays a list of I/O slot information for built-in adapters that are not hot-pluggable but can have Dynamic LPAR operations performed on them.

-State *State*
Limits the output to devices in the specified state. The following states are supported:

0, defined
    Server Virtual Adapter.
1, available
    Server Virtual Adapter Physical Location Code
2, stopped
    Client Partition ID

-type *DeviceType*
Specifies the device type. This flag can be used to restrict output to devices in the specified types. Both physical and virtual devices are listed.

Supported types are as follows:

adapter
    Lists adapters
disk
    Lists disks
lv
    Lists logical volumes and volume groups
optical
    Lists optical devices (cdrom/dvdrom)
tape
    Lists tape devices
tty
    Lists tty devices
ent4sea
    Lists all physical Ethernet adapters and Etherchannel available for creating a shared Ethernet adapter
ven4sea
    Lists all virtual Ethernet adapters available for creating shared Ethernet adapter
ent4ip
    Lists all adapters over which interface can be configured.

-virtual
Restricts output to virtual devices only.

-vpd

Displays platform-specific information for all devices or for a single device when **-dev** *DeviceName* or **-plc** *PhysicalLocationCode* are specified.

Exit Status

Table 1. Command specific return codes

Return code	Description
12	Specified logical volume belongs to the operating system.
13	Specified physical or logical volume is not valid physical or logical volume

See Virtual I/O Server command exit status.

Examples

1. To list all virtual adapters and display the **name** and **status** fields, type:

   ```
   lsdev -type adapter -virtual -field name status
   ```

   The system displays a message similar to the following:

   ```
   name   status

   vhost0  Available
   vhost1  Available
   vhost2  Available
   ent6    Available
   ent7    Available
   ent8    Available
   ent9    Available
   ```
2. To list all devices of type **disk** and display the name and physical location fields, type:

   ```
   lsdev -type disk -field name physloc
   ```

   The system displays a message similar to the following:

   ```
   name    physloc

   hdisk0 U9111.520.10004BA-T15-L5-L0
   hdisk1 U9111.520.10004BA-T15-L8-L0
   hdisk2 U9111.520.10004BA-T16-L5-L0
   hdisk3 U9111.520.10004BA-T16-L8-L0
   hdisk4 UTMP0.02E.00004BA-P1-C4-T1-L8-L0
   hdisk5 UTMP0.02E.00004BA-P1-C4-T2-L8-L0
   hdisk6 UTMP0.02F.00004BA-P1-C8-T2-L8-L0
   hdisk7 UTMP0.02F.00004BA-P1-C4-T2-L8-L0
   hdisk8 UTMP0.02F.00004BA-P1-C4-T2-L11-L0
   vtscsi0 U9111.520.10004BA-V1-C2-L1
   vtscsi1 U9111.520.10004BA-V1-C3-L1
   vtscsi2 U9111.520.10004BA-V1-C3-L2
   vtscsi3 U9111.520.10004BA-V1-C4-L1
   vtscsi4 U9111.520.10004BA-V1-C4-L2
   vtscsi5 U9111.520.10004BA-V1-C5-L1
   ```
3. To display the parent of a devices, type:

   ```
   lsdev -dev hdisk0 -parent
   ```

   The system displays a message similar to the following:

   ```
   parent

   scsi0
   ```
4. To display all I/O slots that are not hot-pluggable but can have DLPAR operations performed on them, type:

   ```
   lsdev -slots
   ```

   The system displays a message similar to the following:

   ```
   U787A.001.DNZ00Y1-P1-C1  Logical I/O Slot  pci4 sisscsia0
   U787A.001.DNZ00Y1-P1-T5  Logical I/O Slot  pci3 ent0 ent1
   U787A.001.DNZ00Y1-P1-T7  Logical I/O Slot  pci2 usbhc0 usbhc1
   ```

```
U9111.520.10DFD8C-V2-C0  Virtual I/O Slot  vsa0
U9111.520.10DFD8C-V2-C2  Virtual I/O Slot  vhost0
U9111.520.10DFD8C-V2-C4  Virtual I/O Slot  Unknown
```

Related Information

The **cfgdev** command, the **chdev** command, the **chpath** command, the **lsmap** command, the **lspath** command, the **mkpath** command, the **mkvdev** command, the **rmdev** command, and the **rmpath** command.

lsfailedlogin Command

Purpose

Lists the contents of the failed login log to the screen.

Syntax

lsfailedlogin

Description

The **lsfailedlogin** command dumps the contents of the failed login log. The failed login file records unsuccessful login attempts by any user on the Virtual I/O Server.

Exit Status

See Virtual I/O Server command exit status.

Security

This command can only be executed by the prime administrator (padmin) user.

Examples

1. To list all failed logins, type:

```
lsfailedlogin
```

Related Information

The **lsgcl** command.

lsfware Command

Purpose

Displays microcode and firmware levels of the system and adapters and devices.

Syntax

lsfware **-all** | **-dev** *Name*

Description

The **lsfware** command displays the platform system firmware microcode level and the service processor microcode levels, if supported. Not all systems contain a service processor, nor do all systems support displaying the system processor level. Information on a specific device is displayed with the **-dev** flag.

The **-all** flag displays system firmware/microcode for all devices. The output from the **-all** flag is always delimiter separated.

Flags

-all Displays microcode level information for all supported devices.

-dev *Name* Displays microcode level information for the named device.

Examples

1. To display the system firmware level and service processor (if present), type:

   ```
   lsfware
   ```

   The system displays a message similar to the following:

   ```
   System Firmware level is TCP99256
   ```
2. To display the microcode level for all supported devices, type:

   ```
   lsfware -all
   ```

   The system displays a message similar to the following:

   ```
   sys0|system:TCP99256
   rmt0|C009
   scraid0|adapter:4.20.18|adapter-boot:4.00.26
   raid-dasd|22:FFC #:DDYS-T0.524D3031.53393446
   raid-dasd|26:FFC #:DDYS-T0.524D3031.53393446
   raid-dasd|2e:FFC #:DDYS-T0.525A3034.53393243
   ....
   ```

lsgcl Command

Purpose

Lists the contents of the global command log to the screen.

Syntax

lsgcl

Description

The **lsgcl** command lists the contents of the global command log (gcl). This log contains a listing of all commands that have been executed by all Virtual I/O Server users. Each listing contains the date and time of execution as well as the userid the command was executed from.

Global command log file format is as follows:

Date	Time	userid	Command	Command options
mmm dd yyyy	hh:mm:ss	cccccccc	Command	Command options span to 80 characters then wraps to the next row

Exit Status

See Virtual I/O Server command exit status.

Security

This command can only be executed by the prime administrator (padmin) user.

Examples

1. To list the contents of the global command log, type:

   ```
   lsgcl
   ```

Related Information

The **lsfailedlogin** command.

IVM lshwres Command

Purpose

Lists the hardware resources of a managed system. This command is valid only in an Integrated Virtualization Manager environment.

Syntax

To list attributes for physical I/O buses

lshwres -r io **--rsubtype** bus **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list attributes for I/O pools

lshwres -r io **--rsubtype** iopool **--level** pool **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list system attributes for I/O pools

lshwres -r io **--rsubtype** iopool **--level** sys **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list attributes for physical I/O slots

lshwres -r io **--rsubtype** slot **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list attributes for physical I/O units

lshwres -r io **--rsubtype** unit **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list partition attributes for memory

lshwres -r mem **--level** lpar **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list system attributes for memory

lshwres -r mem **--level** sys **--maxmem --filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list partition attributes for processors

lshwres -r proc **--level** lpar **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list shared processor pool attributes

lshwres -r proc **--level** pool **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list system attributes for processors

lshwres -r proc **--level** sys **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list virtual ethernet adapter attributes

lshwres -r virtualio **--rsubtype** eth **--level** lpar **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list system attributes for virtual ethernet adapters

lshwres -r virtualio **--rsubtype** eth **--level** sys **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list virtual SCSI adapter attributes

lshwres -r virtualio **--rsubtype** scsi **--level** lpar **--filter** "*FilterData*" **-F** "*AttributeNames*" --**header** -**m** *ManagedSystem*

To list virtual serial adapter attributes

lshwres -r virtualio **--rsubtype** serial **--level** lpar --**filter** "*FilterData*" -**F** "*AttributeNames*" --**header** -**m** *ManagedSystem*

To list partition attributes for virtual I/O slots

lshwres -r virtualio **--rsubtype** slot **--level** lpar --**filter** "*FilterData*" -**F** "*AttributeNames*" --**header** -**m** *ManagedSystem*

To list virtual I/O slot attributes

lshwres -r virtualio **--rsubtype** slot **--level** slot --**filter** "*FilterData*" -**F** "*AttributeNames*" --**header** -**m** *ManagedSystem*

Description

The **lshwres** command lists the hardware resources of the managed-system, including physical I/O, virtual I/O, memory, and processing.

Flags

-r *ResourceType*
--rsubtype *ResourceSubtype*
--level *ResourceLevel*

The combination of -r, --rsubtype, and --level indicates which type of attributes and objects to list. The valid combinations are enumerated below.

- **-r io --rsubtype bus**: List attributes for physical I/O buses
  - ♦ Attributes: unit_phys_loc,bus_id,backplane_phys_loc,bus_drc_index,bus_drc_name
  - ♦ Filters: units, buses
- **-r io --rsubtype slot**: List attributes for physical I/O slots
  - ♦ Attributes: unit_phys_loc, bus_id, phys_loc, drc_index, lpar_name, lpar_id, slot_io_pool_id, description, feature_codes, adapter_feature_codes, pci_vendor_id, pci_device_id, pci_subs_vendor_id, pci_subs_device_id, pci_class, pci_revision_id, bus_grouping, iop, iop_info_stale, vpd_stale, vpd_type, vpd_model, vpd_serial_num, parent_slot_drc_index, drc_name
  - ♦ Filters: {lpar_ids | lpar_names}, units, buses, slots, pools
- **-r io --rsubtype unit**: List attributes for physical I/O units
  - ♦ Attributes: unit_phys_loc
  - ♦ Filters: units
- **-r mem --level lpar**: List partition attributes for memory
  - ♦ Attributes: lpar_name, lpar_id, curr_min_mem, curr_mem, curr_max_mem, pend_min_mem, pend_mem, pend_max_mem, run_min_mem, run_mem
  - ♦ Filters: {lpar_ids | lpar_names}
- **-r mem --level sys --maxmem**: List system attributes for memory
  - ♦ Attributes: configurable_sys_mem, curr_avail_sys_mem, pend_avail_sys_mem, installed_sys_mem, deconfig_sys_mem, sys_firmware_mem, mem_region_size
  - ♦ Additional attributes with --maxmem: required_min_mem_aix_linux
  - ♦ Filters: None
- **-r proc --level lpar**: List partition attributes for processors
  - ♦ Attributes: lpar_name, lpar_id, curr_shared_proc_pool_id, curr_proc_mode, curr_min_proc_units, curr_proc_units, curr_max_proc_units, curr_min_procs, curr_procs, curr_max_procs, curr_sharing_mode, curr_uncap_weight, pend_shared_proc_pool_id, pend_proc_mode, pend_min_proc_units, pend_proc_units, pend_max_proc_units, pend_min_procs, pend_procs, pend_max_procs, pend_sharing_mode, pend_uncap_weight, run_proc_units, run_procs, run_uncap_weight
  - ♦ Filters: {lpar_ids | lpar_names}

- **-r proc --level pool**: List shared processor pool attributes
    - Attributes: shared_proc_pool_id, configurable_pool_proc_units, curr_avail_pool_proc_units, pend_avail_pool_proc_units
    - Filters: None
- **-r proc --level sys**: List system attributes for processors
    - Attributes: configurable_sys_proc_units, curr_avail_sys_proc_units, pend_avail_sys_proc_units, installed_sys_proc_units, deconfig_sys_proc_units, min_proc_units_per_virtual_proc, max_shared_proc_pools
    - Filters: None
- **-r virtualio --rsubtype eth --level lpar**: List virtual Ethernet adapter attributes
    - Attributes: lpar_name, lpar_id, slot_num, state, ieee_virtual_eth, port_vlan_id, addl_vlan_ids, is_trunk, is_required, mac_addr
    - Filters: {lpar_ids | lpar_names}, vlans, slots

- **-r virtualio --rsubtype eth --level sys**: List system attributes for virtual ethernet adapters
    - Attributes: max_vlans_per_port, mac_prefix
    - Filters: None
- **-r virtualio --rsubtype scsi --level lpar**: List virtual SCSI adapter attributes
    - Attributes: lpar_name, lpar_id, slot_num, state, adapter_type, remote_lpar_id, remote_lpar_name, remote_slot_num, is_required
    - Filters: {lpar_ids | lpar_names}, slots
- **-r virtualio --rsubtype serial --level lpar**: List virtual serial adapter attributes
    - Attributes: lpar_name, lpar_id, slot_num, state, adapter_type, supports_hmc, remote_lpar_id, remote_lpar_name, remote_slot_num, is_required
    - Filters: {lpar_ids | lpar_names}, slots
- **-r virtualio --rsubtype slot --level lpar**: List partition attributes for virtual I/O slots
    - Attributes: lpar_name, lpar_id, curr_max_virtual_slots, pend_max_virtual_slots
    - Filters: {lpar_ids | lpar_names}
- **-r virtualio --rsubtype slot --level slot**: List virtual I/O slot attributes
    - Attributes: slot_num, lpar_name, lpar_id, config, state, drc_name
    - Filters: {lpar_ids | lpar_names}, slots

--maxmem *MaximumMemory*

When this option is specified, the required minimum memory amount needed for partitions to support the maximum memory quantity specified is listed. All memory quantities are in megabytes, and are a multiple of the memory region size for the managed- system.

This information is useful for specifying memory amounts in partition profiles.

This option is only valid when listing system level memory resources.

-m *ManagedSystem*

The name of the managed system. This attribute is optional because there is only one system to manage. The name may either be the user-defined name for the managed system, or be in the form tttt-mmm*ssssssss, where tttt is the machine type, mmm is the model, and ssssssss is the serial number of the managed system.

--filter *FilterData*

The filters to apply to the resources to be listed. Filters are used to select which resources of the specified resource type are to be listed. If no filters are used, then all of the resources of the specified resource type will be listed. For example, specific partitions can be listed by using a filter to specify the names or IDs of the partitions to list. Otherwise, if no filter is used, then all the partitions in the managed system will be listed.

The filter data consists of filter name/value pairs, which are in comma separated value (CSV) format. The filter data must be enclosed in double quotes.

The format of the filter data is as follows:

```
"filter-name=value,filter-name=value,..."
```

Note that certain filters accept a comma separated list of values, as follows:

```
""filter-name=value,value,...",..."
```

When a list of values is specified, the filter name/value pair must be enclosed in double quotes. Depending on the shell being used, nested double quote characters may need to be preceded by an escape character, which is usually a '\' character.

Unless otherwise indicated, multiple values can be specified for each filter.

Valid filter names:

buses
    The bus ID of the I/O bus to view
lpar_ids | lpar_names
    Name or ID of the partition to view
pools
    The pool ID of the I/O pool to view
slots
    For physical I/O slots, the DRC index of the slot to view. For virtual I/O slots, the virtual slot number of the slot to view
units
    The the physical location code of the unit to view
vlans
    The virtual LAN of the virtual Ethernet adapters to view

-F *AttributeNames* A delimiter separated list of attribute names for the desired attribute values to be displayed for each resource. If no attribute names are specified, then values for all of the attributes for the resource will be displayed.

When this option is specified, only attribute values will be displayed. No attribute names will be displayed. The attribute values displayed will be separated by the delimiter which was specified with this option.

This option is useful when only attribute values are desired to be displayed, or when the values of only selected attributes are desired to be displayed.

Attribute names:

adapter_type
    Indicates whether the virtual SCSI or serial adapter is a client or server. Valid values are client and server.
backplane_phys_loc
    Physical location code of the backplane on which the bus resides.
bus_drc_index
    DRC index, in hex, of the bus.
bus_drc_name
    DRC name of the bus.
bus_grouping
    Indicates whether bus grouping is required. Possible values:
    - 0 - not required
    - 1 - required

bus_id
    I/O bus unique ID
config
    Virtual slot configuration state. Possible values:
    - empty - no adapter
    - ethernet - virtual ethernet adapter
    - scsi - virtual scsi adapter
    - serial - virtual serial adapter
    - vmc - virtual management channel adapter

configurable_pool_proc_units
    Total number of configurable processing units in the shared processing pool
configurable_sys_mem
    Total amount, in megabytes, of configurable memory on the managed system
configurable_sys_proc_units
    Total number of configurable processing units on the managed system
curr_avail_pool_proc_units
    Current number of configurable processing units in the shared processing pool that are not assigned to partitions
curr_avail_sys_mem
    Current amount, in megabytes, of configurable memory on the managed system that is not assigned to partitions

curr_avail_sys_proc_units
Current number of configurable processing units on the managed system that are not assigned to partitions

curr_max_mem
Maximum amount of memory, in megabytes, that can be dynamically assigned to the partition

curr_max_proc_units
Maximum number of processing units that can be dynamically assigned to the partition. This attribute is only valid for partitions using shared processors.

curr_max_procs
Maximum number of processors or virtual processors that can be dynamically assigned to the partition

curr_max_virtual_slots
Maximum number of virtual slots that can be dynamically configured for the partition

curr_mem
Current amount of memory, in megabytes, assigned to the partition

curr_min_mem
Minimum amount of memory, in megabytes, that can be dynamically assigned to the partition

curr_min_proc_units
Minimum number of processing units that can be dynamically assigned to the partition. This attribute is only valid for partitions using shared processors.

curr_min_procs
Minimum number of processors or virtual processors that can be dynamically assigned to the partition

curr_proc_mode
Indicates whether the partition is using dedicated or shared processors. The mode cannot change dynamically. Valid values:

◊ ded - dedicated
◊ shared - shared

curr_proc_units
Current number of processing units assigned to the partition. This attribute is only valid for partitions using shared processors.

curr_procs
Current number of processors or virtual processors assigned to the partition

curr_shared_proc_pool_id
The shared processor pool that this partition is currently participating in. This attribute is only valid for partitions using shared processors.

curr_sharing_mode
The current sharing mode for a partition. Valid values:

◊ keep_idle_procs - valid with dedicated processor mode
◊ share_idle_procs - valid with dedicated processor mode
◊ cap - capped mode. valid with shared processor mode
◊ uncap - uncapped mode. valid with shared processor mode

curr_uncap_weight
The current weighted average of processing priority when in uncapped sharing mode Valid values are 0 - 255

deconfig_sys_mem
Amount of memory, in megabytes, on the managed system that has been unconfigured. This includes memory that has been unconfigured by the system due to hardware failure, and memory that has been manually unconfigured

deconfig_sys_proc_units
Number of processing units on the managed system that have been unconfigured. This includes processing units that have been unconfigured by the system due to hardware failure, and processing units that have been manually unconfigured

description
A description of the I/O adapter which is in the slot.

drc_index
The DRC index, in hexadecimal, of the I/O slot.

drc_name
The DRC name of the I/O slot.

feature_codes

The most likely feature code for the I/O adapter. The feature code may be different for each operating system. The **adapter_feature_codes** attribute should be used to distinguish the feature code specific to each operating system.

adapter_feature_codes

List of possible feature codes for the I/O adapter sorted by likely match. Each element in the list consists of two feature codes - one for each operating system. This list follows the following format:

```
aix_feature_code1/linux_feature_code1,aix_feature_code2/linux_feature_code2,...
```

installed_sys_mem

Total amount, in megabytes, of memory installed on the managed system

installed_sys_proc_units

Total number of processing units installed on the managed system

io_pool_id

A unique identifier for an I/O pool. I/O pools are used in some I/O clustering environments.

iop

Indicates whether the I/O adapter is an I/O processor. Valid values:

◊ 0 - no
◊ 1 - yes

iop_info_stale

Indicates whether the information about the I/O processor is stale. Valid values:

◊ 0 - no
◊ 1 - yes

is_required

Indicates whether the I/O slot or virtual I/O adapter is required for the partition. Valid values:

◊ 0 - no
◊ 1 - yes

is_trunk

Indicates whether the virtual ethernet adapter is the trunk or uplink adapter for the virtual LAN. Valid values:

◊ 0 - no
◊ 1 - yes

lpar_ids

Partition ID. Valid values are 1 through the maximum number of partitions supported on the managed system (max_lpars).

lpar_name

User defined name for the partition.

mac_addr

MAC address for the virtual Ethernet adapter

mac_prefix

The first 3 bytes of the MAC address to be assigned to all virtual Ethernet adapters for this managed system. This must be specified as a 3 byte hexadecimal value (e.g. 32ab10) and can only be configured in the **mkgencfg** command.

max_io_pools

Maximum number of I/O pools supported on the managed system

max_shared_proc_pools

Maximum number of shared processing pools which are supported on the managed system

mem_region_size

The memory region size, in megabytes, for the managed system. The memory regions size represents the granularity of memory allocation to partitions.

min_proc_units_per_virtual_proc

Minimum number of processing units that are required for each virtual processor assigned to a partition

parent_slot_drc_index

DRC index, in hex, of the parent slot. A value of none indicates there is no parent slot

pci_class

PCI class code for the I/O adapter. This value is displayed in hex.

pci_revision_id

Vendor-assigned code indicating the revision number of the I/O adapter. This value is displayed in hexadecimal.

pci_subs_device_id
: Vendor-assigned code identifying the type of driver that is installed for the I/O adapter. This value is displayed in hexadecimal.

pci_subs_vendor_id
: ID of the vendor that made the driver for the I/O adapter. This value is displayed in hexadecimal.

pci_vendor_id
: ID of the vendor that made this I/O adapter. This value is displayed in hexadecimal.

pend_avail_pool_proc_units
: After system restart, the number of configurable processing units in the shared processing pool that are not assigned to partitions

pend_avail_sys_mem
: After system restart, the amount, in megabytes, of configurable memory on the managed system that is not assigned to partitions

pend_avail_sys_proc_units
: After system restart, the number of configurable processing units on the managed system that are not assigned to partitions

pend_max_mem
: After partition restart, the maximum amount of memory, in megabytes, that can be dynamically assigned to the partition

pend_max_proc_units
: After partition restart, the maximum number of processing units that can be dynamically assigned to the partition. This attribute is only valid for partitions using shared processors.

pend_max_procs
: After partition restart, the maximum number of processors or virtual processors that can be dynamically assigned to the partition.

pend_max_virtual_slots
: After partition restart, the maximum number of virtual slots that can be dynamically created for the partiton.

pend_mem
: The target amount of memory, in megabytes, assigned to the partition.

pend_min_mem
: After partition restart, the minimum amount of memory, in megabytes, that can be dynamically assigned to the partition.

pend_min_proc_units
: After partition restart, the minimum number of processing units that can be dynamically assigned to the partition. This attribute is only valid for partitions using shared processors.

pend_min_procs
: After partition restart, the minimum number of processors or virtual processors that can be dynamically assigned to the partition.

pend_proc_mode
: Indicates whether the partition will be using dedicated or shared processors after restart. Valid values:
  - ◊ ded - dedicated
  - ◊ shared - shared

pend_proc_units
: The target number of processing units assigned to the partition. This attribute is only valid for partitions using shared processors.

pend_procs
: The target number of processors or virtual processors assigned to the partition

pend_shared_proc_pool_id
: The shared processor pool that a partition is will participate in after restart. This attribute is only valid for partitions using shared processors.

pend_sharing_mode
: The target sharing mode for a partition. Valid values:
  - ◊ keep_idle_procs - valid with pending dedicated processor mode
  - ◊ share_idle_procs - valid with pending dedicated processor mode
  - ◊ cap - capped mode. valid with pending shared processor mode
  - ◊ uncap - uncapped mode. valid with pending shared processor mode

pend_uncap_weight
: The target weighted average of processing priority when in uncapped sharing mode Valid values are: 0 - 255

phys_loc
Physical location code of the slot.

port_vlan_id
Port virtual LAN ID for the virtual ethernet adapter.

remote_lpar_id
For client adapters, this specifies the ID of the partition which has the virtual serial/SCSI server adapter for this adapter. For server adapters, this specifies the ID of the partition which has the only client virtual serial/SCSI adapter allowed to connect to this adapter. A value of any indicates that any client virtual serial/SCSI adapter should be allowed to connect to this adapter

remote_lpar_name
The partition name which corresponds to the remote_lpar_id.

remote_slot_num
For client adapters, this specifies the virtual slot number of the virtual serial/SCSI server adapter for this adapter. For server adapters, this specifies the virtual slot number of the only client virtual serial/SCSI adapter allowed to connect to this adapter. A value of any indicates that any client virtual serial/SCSI adapter should be allowed to connect to this adapter.

required_min_mem_aix_linux
For an AIX or Linux partition, this is the required minimum memory amount, in megabytes, for the maximum memory amount specified with the --**maxmem** parameter.

run_mem
Current amount of memory, in megabytes, that the partition has varied on.

run_min_mem
The amount of memory, in megabytes, returned by a running partition's operating system to represent the smallest memory assignment which is currently supported by the operating system. If the operating system does not support this feature, then the value will be 0.

run_proc_units
Number of processing units that are varied on for the partition.

run_procs
Number of processors or virtual processors that are varied on for the partition.

run_uncap_weight
The operating system's current setting for the weighted average of processing priority when in uncapped sharing mode. Valid values are: 0 - 255

shared_proc_pool_id
A unique decimal identifier for a shared processing pool.

slot_io_pool_id
A unique decimal identifier of the I/O pool to which the slot is assigned. A value of `none` indicates that the slot is not assigned to any I/O pools.

slot_num
Virtual slot number.

slots
A list of slot DRC indices.

state
The state of a virtual slot. A virtual slot must be able to transition into the off state before a dynamic reconfiguration of the slot can succeed. Valid states:

◊ off - the virtual slot is ready for dynamic reconfiguration
◊ on - the virtual slot is not yet ready for dynamic reconfiguration

supports_hmc
Indicates whether or not the virtual serial adapter is enabled for HMC client support. Valid values:

◊ 0 - no
◊ 1 - yes

sys_firmware_mem
Amount of memory, in megabytes, on the managed system that is being used by system firmware.

unit_phys_loc
Physical location code of the I/O unit.

vpd_model
Model of the I/O adapter.

vpd_serial_num
Serial number of the I/O adapter.

vpd_stale
: Indicates whether or not type, model, and serial number of the adapter is not current. Valid values:
  - ◊ 0 - not current
  - ◊ 1 - current

vpd_type
: Type of the I/O adapter.

--header Display a header record, which is a delimiter separated list of attribute names for the attribute values that will be displayed. This header record will be the first record displayed. This option is only valid when used with the -F option.

Exit Status

This command has a return code of zero on success.

Security

This command is accessible by all users.

Examples

1. List system level memory information and include the minimum memory required to support a maximum of 1024 MB:

   ```
   lshwres -r mem --level sys --maxmem 1024
   ```
2. List all memory information for partitions lpar1 and lpar2, and only display attribute values, following a header of attribute names:

   ```
   lshwres -r mem --level lpar --filter "\"lpar_names=lpar1,lpar2\"" -F --header
   ```
3. List all I/O units on the system:

   ```
   lshwres -r io --rsubtype unit
   ```
4. List all virtual Ethernet adapters on the managed system:

   ```
   lshwres -r virtualio --rsubtype eth --level lpar
   ```
5. List all virtual slots for partition lpar1:

   ```
   lshwres -r virtualio --rsubtype slot --level slot --filter "lpar_names=lpar1"
   ```
6. List only the installed and configurable processors on the system:

   ```
   lshwres -r proc --level sys -F installed_sys_proc_units,configurable_sys_proc_units
   ```

Related Information

The **lssyscfg** command.

IVM lsled Command

Purpose

List states for physical and virtual LEDs. This command is valid only in an Integrated Virtualization Manager environment.

Syntax

To list physical platform LEDs:

lsled -r sa **-t** phys **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list virtual partition LEDs:

lsled -r sa **-t** virtuallpar **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list virtual platform LEDs:

lsled -r sa **-t** virtualsys **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

Description

The **lsled** command lists the states of physical and virtual LEDs.

Flags

-r *ResourceType*
The type of LED resources to list. The only valid value is sa for System Attention (SA) LEDs.

-t *SystemAttentionType*
The type of System Attention (SA) LEDs to list.

- **-r sa -t virtuallpar**: List attributes for virtual partition system attention LEDs
  - ♦ Attributes: lpar_id, lpar_name, state
  - ♦ Filters: { lpar_ids | lpar_names}
- **-r sa -t virtualsys**: List attributes for virtual system attention LEDs
  - ♦ Attributes: state
  - ♦ Filters: None
- **-r sa -t phys**: List attributes for physical system attention LEDs
  - ♦ Attributes: state
  - ♦ Filters: None

-m *ManagedSystem*
The name of the managed system. This attribute is optional because there is only one system to manage. The name may either be the user-defined name for the managed system, or be in the form tttt-mmm*ssssssss, where tttt is the machine type, mmm is the model, and ssssssss is the serial number of the managed system.

--filter *FilterData*
The filters to apply to the resources to be listed. Filters are used to select which resources of the specified resource type are to be listed. If no filters are used, then all of the resources of the specified resource type will be listed. For example, specific partitions can be listed by using a filter to specify the names or IDs of the partitions to list. Otherwise, if no filter is used, then all the partitions in the managed system will be listed.

The filter data consists of filter name/value pairs, which are in comma separated value (CSV) format. The filter data must be enclosed in double quotes.

The format of the filter data is as follows:

```
"filter-name=value,filter-name=value,..."
```

Note that certain filters accept a comma separated list of values, as follows:

```
""filter-name=value,value,...",..."
```

When a list of values is specified, the filter name/value pair must be enclosed in double quotes. Depending on the shell being used, nested double quote characters may need to be preceded by an escape character, which is usually a '\' character.

Unless otherwise indicated, multiple values can be specified for each filter.

Valid filter names for -r lpar

lpar_ids
    ID of the partitions to view
lpar_names
    Name of the partitions to view

-F *AttributeNames*

A delimiter separated list of attribute names for the desired attribute values to be displayed for each resource. If no attribute names are specified, then values for all of the attributes for the resource will be displayed.

When this option is specified, only attribute values will be displayed. No attribute names will be displayed. The attribute values displayed will be separated by the delimiter which was specified with this option.

This option is useful when only attribute values are desired to be displayed, or when the values of only selected attributes are desired to be displayed.

Attribute Names

- **lpar_id**: Unique integer identifier for the partition
- **lpar_name**: name of the partition
- **state**: The current state of the LED. Valid values:
  - ♦ off - the LED is off
  - ♦ on - the LED is on

--header

Display a header record, which is a delimiter separated list of attribute names for the attribute values that will be displayed. This header record will be the first record displayed. This option is only valid when used with the -F option.

Exit Status

This command has a return code of zero on success.

Secuirty

This command is accessible by all users.

Examples

1. Display the physical system attention LED for the system:

   ```
   lsled -r sa -t phys
   ```
2. Display all of the virtual partition system attention LEDs:

   ```
   lsled -r sa -t virtuallpar
   ```
3. Display the virtual partition system attention LEDs for partitions lpar1 and lpar2:

   ```
   lsled -r sa -t virtuallpar --filter \"lpar_names=lpar1,lpar2\"
   ```

Related Information

The **chled** command.

lslparinfo Command

Purpose

Displays the logical partition number and name.

Syntax

lslparinfo

Description

The **lslparinfo** command displays LPAR number and LPAR name. If LPAR does not exist, -1 is displayed for LPAR number and NULL for LPAR name.

Examples

1. To display the logical partition number and name, type:

```
lslparinfo
```

IVM lslparutil Command

Purpose

List utilization metrics for the managed system and partitions. This command is available only in an Integrated Virtualization Manager environment.

Syntax

To list utilization metrics:

lslparutil {**-r** sys | pool | lpar } **--startyear** *Year* **--startmonth** *Month* **--startday** *Day* **--starthour** *Hour* **--startminute** *Minute* **--endyear** *Year* **--endmonth** *Month* **--endday** *Day* **--endhour** *Hour* **--endminute** *Minute* **-n** *NumberOfEvents* **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header** **-m** *ManagedSystem*

To list utilization configuration attributes:

lslparutil **-r** config **-F** "*AttributeNames*" **--header** **-m** *ManagedSystem*

Description

The **lslparutil** command lists utilization metrics for the managed system and partitions. The sample rate at which these metrics are gathered can be changed, or disabled. The information gathered from this command allows for the determination of processor utilization on a system and partition level over a given period of time.

Flags

-r *ResourceType*

The type of resources to list:

- **-r config**: List configuration attributes for utilization monitoring
  - ♦ **Attributes**: sample_rate
  - ♦ **Filters**: None
- **-r lpar**: List utilization data for partitions
  - ♦ **Attributes**: time, sys_time, event_type, resource_type, time_cycles, lpar_id, uptime, curr_proc_mode, curr_proc_units, curr_procs, curr_sharing_mode, curr_uncap_weight, curr_5250_cpw_percent, curr_mem,entitled_cycles, capped_cycles, uncapped_cycles
  - ♦ **Filters**: { lpar_ids | lpar_names}
- **-r pool**: List utilization data for shared processor pools
  - ♦ **Attributes**: time, sys_time, event_type, resource_type, time_cycles, shared_proc_pool_id, curr_avail_pool_proc_units, configurable_pool_proc_units, borrowed_pool_proc_units, total_pool_cycles, utilized_pool_cycles
  - ♦ **Filters**: None
- **-r sys**: List utilization data for the managed system
  - ♦ **Attributes**: time, sys_time, event_type, resource_type, state, configurable_sys_proc_units, configurable_sys_mem, curr_avail_sys_proc_units, curr_avail_5250_cpw_percent, curr_avail_sys_mem, sys_firmware_mem, proc_cycles_per_second
  - ♦ **Filters**: None

--startyear *Year*
Starting year filter (default 1970)

--startmonth *Month*
Starting month filter (default 1) January is 1. December is 12

--startday *Day*
Starting day filter (default 1)

--starthour *Hour*
Starting hour filter (default 0)

--startminute *Minute*
Starting minute filter (default 0)

--endyear *Year*
Ending year filter (default now)

--endmonth *Month*
Ending month filter (default now)

--endday *Day*
Ending day filter (default now)

--endhour *Hour*
Ending hour filter (default now)

--endminute *Minute*
Ending minute filter (default now)

--n *NumberOfEvents*
Starting from the most recent event, the maximum number of samples to return.

-m *ManagedSystem*
The name of the managed system. This attribute is optional because there is only one system to manage. The name may either be the user-defined name for the managed system, or be in the form tttt-mmm*ssssssss, where tttt is the machine type, mmm is the model, and ssssssss is the serial number of the managed system.

--filter *FilterData*
The filters to apply to the resources to be listed. Filters are used to select which resources of the specified resource type are to be listed. If no filters are used, then all of the resources of the specified resource type will be listed. For example, specific partitions can be listed by using a filter to specify the names or IDs of the partitions to list. Otherwise, if no filter is used, then all the partitions in the managed system will be listed.

The filter data consists of filter name/value pairs, which are in comma separated value (CSV) format. The filter data must be enclosed in double quotes.

The format of the filter data is as follows:

```
"filter-name=value,filter-name=value,..."
```

Note that certain filters accept a comma separated list of values, as follows:

```
""filter-name=value,value,...",..."
```

When a list of values is specified, the filter name/value pair must be enclosed in double quotes. Depending on the shell being used, nested double quote characters may need to be preceded by an escape character, which is usually a '\' character.

Unless otherwise indicated, multiple values can be specified for each filter.

Valid filter names for -r lpar

lpar_ids
    ID of the partitions to view

lpar_names
    Name of the partitions to view

-F *AttributeNames*
A delimiter separated list of attribute names for the desired attribute values to be displayed for each resource. If no attribute names are specified, then values for all of the attributes for the resource will be displayed.

When this option is specified, only attribute values will be displayed. No attribute names will be displayed. The attribute values displayed will be separated by the delimiter which was specified with this option.

This option is useful when only attribute values are desired to be displayed, or when the values of only selected attributes are desired to be displayed.

Attribute Names:

borrowed_pool_proc_units
: Processing units that are being borrowed from powered off partitions with dedicated processors.

capped_cycles
: The number of capped processing cycles utilized by this partition since the system started.

configurable_pool_proc_units
: The number of configurable processing units in the shared pool at the time of the sample

configurable_sys_mem
: The amount of configurable system memory (in megabytes) at the time of the sample.

configurable_sys_proc_units
: The number of configurable system processing units at the time of the sample.

curr_5250_cpw_percent
: The 5250 CPW percent assigned to the partition at the time of the sample.

curr_avail_5250_cpw_percent
: The 5250 CPW percent available to be assigned to partitions at the time of the sample.

curr_avail_pool_proc_units
: The number of processing units available to be assigned to partitions at the time of the sample.

curr_avail_sys_mem
: The amount of memory (in megabytes) available to be assigned to partitions at the time of the sample.

curr_avail_sys_proc_units
: The number of processing units available to be assigned to partitions at the time of the sample.

curr_mem
: The amount of memory (in megabytes) assigned to the partition at the time of the sample.

entitled_cycles
: The number of processing cycles to which the partition has been entitled since the system started. This value is based on the number of processing units assigned to the partition, and may be greater than, or smaller than the number of cycles actually used.

curr_proc_mode
: The processing mode for the partition at the time of the sample. Valid values

- ◊ ded: dedicated processor mode
- ◊ shared: shared processor mode curr_proc_units

curr_procs
: The number of processors or virtual processors assigned to the partition at the time of the sample.

curr_sharing_mode
: The sharing mode of the partition at the time of the sample. Valid values:

- ◊ keep_idle_procs: valid with dedicated processor mode
- ◊ share_idle_procs: valid with dedicated processor mode
- ◊ cap: capped mode. valid with shared processor mode
- ◊ uncap: uncapped mode. valid with shared processor mode

curr_uncap_weight
: The current weighted average of processing priority when in uncapped sharing mode at the time of the sample. The smaller the value, the lower the weight. Valid values are 0-255

event_type

The type of event. This will be displayed with a constant value of sample for all samples except the config resource type.

lpar_id

The unique integer identifier for the partition

proc_cycles_per_second

Processing cycles per second on one physical processor. This value is static for a particular managed system.

resource_type

The resource type queried. Valid values are sys, pool, and lpar, depending on which value is supplied for the **-r** flag.

sample_rate

The rate at which samples are obtained. This can be changed with the **chlparutil** command. Valid values:

- 0: Samples will not be retrieved.
- 30: Samples will be retrieved every 30 seconds. This is the default value.

shared_proc_pool_id

The unique decimal identifier for a shared processing pool.

state

The state of the managed system at the time of the sample.

sys_firmware_mem

Amount of memory, in megabytes, on the managed system that is being used by system firmware at the time of the sample.

sys_time

The time on the managed system that the sample was taken. The time and sys_time attributes will have the same value.

time

The time on the management partition that this sample was taken. The time and sys_time attributes will have the same value.

time_cycles

The number of time cycles since the system was started.

total_pool_cycles

The total number of processing cycles available in the shared pool since the system was started.

uncapped_cycles

The number of uncapped processing cycles utilized by this partition since the system was started.

uptime

The amount of time (in seconds) that the partition has been running at the time of the sample.

utilized_pool_cycles

The number of processing cycles in the shared pool that have been utilized since the system was started.

--header

Display a header record, which is a delimiter separated list of attribute names for the attribute values that will be displayed. This header record will be the first record displayed. This option is only valid when used with the -F option.

Exit Status

This command has a return code of zero on success.

Security

This command is accessible by all users.

Examples

1. To list the last five shared processing pool utilization metrics, type:

```
lslparutil -r pool -n 5
```

2. To calculate the shared processing pool utilization in percent over a five-minute time period, type:

```
lslparutil -r pool --startyear 2005 --startmonth 6 --startday 26 --starthour 12 --startminute 8 \
 --endyear 2005 --endmonth 6 --endday 26 --endhour 12 --endminute 13 -F time,total_pool_cycles, \
utilized_pool_cycles

06/26/2005 12:08:01,134841811733640,467081011935
06/26/2005 12:08:31,134854347365860,467116506907
06/26/2005 12:09:02,134866883128692,467152556956
06/26/2005 12:09:32,134879415157938,467188374373
06/26/2005 12:10:02,134891946956456,467223704573
06/26/2005 12:10:32,134904482088726,467258616569
06/26/2005 12:11:03,134917026289150,467295577359
06/26/2005 12:11:33,134929553859752,467333227651
06/26/2005 12:12:03,134942086330068,467368397739
06/26/2005 12:12:33,134954622214624,467403199531
06/26/2005 12:13:04,134967149091025,467439053292

Pool utilization = (utilized_pool_cycle / total_pool_cycles) * 100
Pool utilization = ((467439053292 - 467081011935) / (134967149091025 - 134841811733640)) * 100
Pool utilization = 0.29%
```

3. To calculate the processing utilization in percent for partition 1 over the last 11 samples, type:

```
lslparutil -r lpar -F time,lpar_id,entitled_cycles,capped_cycles,uncapped_cycles --filter lpar_ids

06/26/2005 12:08:01,1,13475439617080,353179654833,93964052971
06/26/2005 12:08:31,1,13476693184663,353213970760,93964052971
06/26/2005 12:09:02,1,13477946765207,353248812551,93964052971
06/26/2005 12:09:32,1,13479199972343,353283141535,93964052971
06/26/2005 12:10:02,1,13480453156357,353317211748,93964052971
06/26/2005 12:10:32,1,13481706673802,353350985013,93964052971
06/26/2005 12:11:03,1,13482961098044,353386674795,93964052971
06/26/2005 12:11:33,1,13484213859686,353423048854,93964052971
06/26/2005 12:12:03,1,13485467110700,353456792591,93964052971
06/26/2005 12:12:33,1,13486720703117,353490258336,93964052971
06/26/2005 12:13:04,1,13487973395246,353524992184,93964052971

Processor utilization = ((capped_cycles + uncapped_cycles) / entitled_cycles) * 100
Processor utilization = (((353524992184 - 353179654833) + (93964052971 - 93964052971)) / (13487973
Processor utilization = 2.76%
```

Related Information

The **chlparutil** and **lssyscfg** commands.

lslv Command

Purpose

Displays information about a logical volume.

Syntax

lslv **-map** | **-pv** *LogicalVolume* **-field** *FieldName* **-fmt** *Delimiter*

lslv **-free** **-field** *Fieldname* **-fmt** *Delimiter*

Description

The **lslv** command displays the characteristics and status of the *LogicalVolume* or lists the logical volume allocation map for the physical partitions on the *PhysicalVolume* in which the logical volume is located. The logical volume can be a name or identifier.

If no flags are specified, the following status is displayed:

Logical volume	Name of the logical volume. Logical volume names must be unique systemwide and can range from 1 to 15 characters.
Volume group	Name of the volume group. Volume group names must be unique systemwide and can range from 1 to 15 characters.
Logical volume identifier	Identifier of the logical volume.
Permission	Access permission; `read-only` or `read-write`.
Volume group state	State of the volume group. If the volume group is activated with the **activatevg** command, the state is either `active/complete` (indicating all physical volumes are active) or `active/partial` (indicating all physical volumes are not active). If the volume group is not activated with the **activatevg** command, the state is `inactive`.
Logical volume state	State of the logical volume. The `Opened/stale` status indicates the logical volume is open but contains physical partitions that are not current. `Opened/syncd` indicates the logical volume is open and synchronized. `Closed` indicates the logical volume has not been opened.
Type	Logical volume type.
Write verify	Write verify state of On or Off.
Mirror write consistency	Mirror write consistency state of Yes or No.
Max LPs	Maximum number of logical partitions the logical volume can hold.
PP size	Size of each physical partition.
Copies	Number of physical partitions created for each logical partition when allocating.
Schedule policy	Sequential or parallel scheduling policy.
LPs	Number of logical partitions currently in the logical volume.
PPs	Number of physical partitions currently in the logical volume.
Stale partitions	Number of physical partitions in the logical volume that are not current.
Bad blocks	Bad block relocation policy.
Inter-policy	Inter-physical allocation policy.
Strictness	Current state of allocation. Possible values are `strict`, `nonstrict`, or `superstrict`. A strict allocation states that no copies for a logical partition are allocated on the same physical volume. If the allocation does not follow the strict criteria, is called nonstrict. A nonstrict allocation states that at least one occurrence of two physical partitions belong to the same logical partition. A superstrict allocation states that no partition from one mirror copy may reside the same disk as another mirror copy.

Intra-policy Intra-physical allocation policy.

Upper bound If the logical volume is super strict, upper bound is the maximum number of disks in a mirror copy.

Relocatable Indicates whether the partitions can be relocated if a reorganization of partition allocation takes place.

Mount point File system mount point for the logical volume, if applicable.

Label Specifies the label field for the logical volume.

PV distribution The distribution of the logical volume within the volume group. The physical volumes used, the number of logical partitions on each physical volume, and the number of physical partitions on each physical volume are shown.

striping width The number of physical volumes being striped across.

strip size The number of bytes per stripe.

The **-free** flag displays logical volumes that are available to be used as backing devices for virtual storage.

Full scripting support is available by using the **-field** *FieldNames* and **-fmt** *Delimiter* flags. The **-field** flag will allow the user to select which output fields to display and in what order, while the **-fmt** flag provides scriptable output. The output fields will be displayed in the order they appear on the command line.

Flags

-free Lists only logical volumes that are available for use as a backing device for virtual SCSI.

-field Specifies the list of fields to display. The following fields are supported if no flags are specified:

lvname
: Name of the logical volume. Logical volume names must be unique systemwide and can range from 1 to 15 characters.

vgname
: Name of the volume group. Volume group names must be unique systemwide and can range from 1 to 15 characters.

lvid
: Identifier of the logical volume.

access
: Access permission: read-only or read-write.

vgstate
: State of the volume group. If the volume group is activated with the **activatevg** command, the state is either `active/complete` (indicating all physical volumes are active) or `active/partial` (indicating all physical volumes are not active). If the volume group is not activated with the **deactivatevg** command, the state is `inactive`.

lvstate
: State of the logical volume. The `opened/stale` status indicates the logical volume is open but contains physical partitions that are not current. `Opened/syncd` indicates the logical volume is open and synchronized. `Closed` indicates the logical volume has not been opened.

type
: Logical volume type.

wverify
: Write verify state of on or off.

mwc
: Mirror write consistency state of on or off.

maxlps
: Maximum number of logical partitions the logical volume can hold.

ppsize
: Size of each physical partition.

copies
: Number of physical partitions created for each logical partition when allocating.

spolicy
: Sequential or parallel scheduling policy.

lps
: Number of logical partitions currently in the logical volume.

pps
: Number of physical partitions currently in the logical volume.

stale

Number of physical partitions in the logical volume that are not current.

bbpolicy
: Bad block relocation policy.

inter
: Inter-physical allocation policy.

intra
: Intra-physical allocation policy.

ubound
: If the logical volume is super strict, upper bound is the maximum number of disks in a mirror copy.

relocatable
: Indicates whether the partitions can be relocated if a reorganization of partition allocation takes place.

mount
: File system mount point for the logical volume, if applicable.

label
: Specifies the label field for the logical volume.

separatepv
: The strictness value. Current state of allocation, strict, nonstrict, or superstrict. A strict allocation states that no copies for a logical partition are allocated on the same physical volume. If the allocation does not follow the strict criteria, it is called nonstrict. A nonstrict allocation states that at least one occurrence of two physical partitions belong to the same logical partition. A superstrict allocation states that no partition from one mirror copy may reside the same disk as another mirror copy.

serialio
: Serialization of overlapping IOs state of yes or no. If serialization is turned on (yes), then overlapping IOs are not allowed on a block range, and only a single IO in a block range is processed at any one time. Most applications, such as file systems and databases, perform serialization; therefore, serialization should be turned off (no). The default setting for new logical volumes is no.

The following fields are supported if the **-pv** flag is specified:

pvname
: Physical volume disk name

copies
: The following three fields:
  - ◊ The number of logical partitions containing at least one physical partition (no copies) on the physical volume
  - ◊ The number of logical partitions containing at least two physical partitions (one copy) on the physical volume
  - ◊ The number of logical partitions containing three physical partitions (two copies) on the physical volume

inband
: The percentage of physical partitions on the physical volume that belong to the logical volume and were allocated within the physical volume region specified by Intra-physical allocation policy.

dist
: The number of physical partitions allocated within each section of the physical volume: outer edge, outer middle, center, inner middle, and inner edge of the physical volume.

The following fields are supported if the **-map** flag is specified:

lpnum
: Logical partition number.

pvname1
: Physical volume name where the logical partition's first physical partition is located.

ppnum1
: First physical partition number allocated to the logical partition.

pvname2
: Physical volume name where the logical partition's second physical partition (first copy) is located.

ppnum2
: Second physical partition number allocated to the logical partition.

The following fields are supported if the **-free** flag is specified:

lvname
: Logical partition number.

size
: Physical volume name where the logical partition's first physical partition is located.

vgname
: Name of the volume group. Volume group names must be unique systemwide and can range from 1 to 15 characters.

-fmt
Specifies a delimiter character to separate output fields.

-map
Lists the following fields for each logical partition:

LPs
: Logical partition number.

PV1
: Physical volume name where the logical partition's first physical partition is located.

PP1
: First physical partition number allocated to the logical partition.

PV2
: Physical volume name where the logical partition's second physical partition (first copy) is located.

PP2
: Second physical partition number allocated to the logical partition.

-pv
Lists the following fields for each physical volume in the logical volume:

PV
: Physical volume name.

Copies
: The following three fields:
  - ◊ The number of logical partitions containing at least one physical partition (no copies) on the physical volume
  - ◊ The number of logical partitions containing at least two physical partitions (one copy) on the physical volume
  - ◊ The number of logical partitions containing three physical partitions (two copies) on the physical volume

In band
: The percentage of physical partitions on the physical volume that belong to the logical volume and were allocated within the physical volume region specified by Intra-physical allocation policy.

Distribution
: The number of physical partitions allocated within each section of the physical volume: outer edge, outer middle, center, inner middle, and inner edge of the physical volume.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To display information about logical volume **lv03**, type:

   ```
   lslv lv03
   ```

   Information about logical volume **lv03**, its logical and physical partitions, and the volume group to which it belongs is displayed.
2. To display information about logical volume **lv03** by physical volume, type:

   ```
   lslv -pv lv03
   ```

   The characteristics and status of **lv03** are displayed, with the output arranged by physical volume.
3. To display a list of logical volumes that can be used as backing devices, type:

   ```
   lslv -free
   ```

   The system displays a message similar to the following:

```
LV NAME          SIZE(megabytes) VOLUME GROUP
lv00             64              rootvg
lv01             64              rootvg
```

4. To display only the type and volume group of logical volume hd6 and separate the data by a : (colon) , type:

   ```
   lslv hd6 -field type vgname -fmt :
   ```

   The system displays a message similar to the following:

   ```
   paging:rootvg
   ```

Related Information

The **mklv** command, the **extendlv** command, and the **rmlv** command.

lsmap Command

Purpose

Displays the mapping between physical, logical, and virtual devices.

Syntax

lsmap { **-vadapter** *ServerVirtualAdapter* | **-plc** *PhysicalLocationCode* | **-all** }

lsmap **-type** *BackingDeviceType* | **-net**

lsmap **-fmt** *Delimiter* **-field** *FieldNames*

Description

The **lsmap** command displays the mapping between virtual host adapters and the physical devices they are backed to. Given a device name (*ServerVirtualAdapter*) or physical location code (*PhysicalLocationCode*) of a server virtual adapter, the device name of each connected virtual target device (child devices), its logical unit number, backing device(s) and the backing devices physical location code is displayed. If the **-net** flag is specified the supplied device must be a virtual server Ethernet adapter.

The **-fmt** flag divides the output by a user-specified delimiter/character (delimiter). The delimiter can be any non-white space character. This format is provided to facilitate scripting.

The **-type** flag limits the types of backing devices displayed to the user-specified list, *BackingDeviceType*. With the exception of the net type, any combination of device types may be specified. The net type can not be combined with any other type.

Flags

-all
Specifies **lsmap** output should be displayed for all virtual SCSI devices. If used with the **-net** flag, virtual Ethernet adapters will be displayed.

-field *FieldName*
Specifies the list of fields to display. The following fields are supported:

svsa
: Server virtual adapter.

physloc
: Server virtual adapter Physical location code

clientid
: Client partition ID

vtd
: Virtual target device. Not valid if the **-net** flag is specified.

lun
: Logical unit number. Not valid if the **-net** flag is specified.

backing
: Backing device

bdphysloc
: Backing device physical location code

svea
: Shared Ethernet adapter. Only valid if the **-net** flag is specified.

-fmt *delimiter*
Specifies a delimiter character to separate output fields.

-net
Specifies the supplied device is a server virtual Ethernet adapter or if used with the **-all** flag all virtual Ethernet adapters and backing devices are displayed.

-plc *PhysicalLocatoinCode*
Specifies the device physical location code of a server virtual adapter. This flag cannot be used with the **-vadapter** flag.

-type
Specifies the type of devices to display. The following fields are supported:

disk

List physical backing devices.

lv
: List logical volume backing devices.

optical
: List optical backing devices.

net
: List network devices. (This option can not be used in combination with disk, lv, or optical)

-vadapter *VirtualServerAdapter*
: Specifies the device name of a server virtual adapter. This flag cannot be used with the **-plc** flag.

Output Field Definitions

Field	Description
SVSA	Server Virtual SCSI Adaper
Physloc	Physical Location Code
VTD	Virtual Target Device
LUN	Logical Unit Number
SVEA	Server Virtual Ethernet Adapter
SEA	Shared Ethernet Adapter

Exit Status

Return code	Description
11	No VTDs associated with device
12	No SEAs associated with device
15	Specified device is not a server virtual SCSI adapter
16	Specified device is not a server virtual Ethernet adapter
17	Specified device in not in the AVAILABLE state

Examples

1. To list all virtual target devices and backing devices mapped to the server virtual SCSI adapter **vnode2**, type:

   ```
   lsmap -vadapter vhost2
   ```

   The system displays a message similar to the following:

   ```
    SVSA         Physloc                                         Client Partition ID
   ------------ -------------------------------------------- ------------------
   vhost0       U9111.520.10004BA-V1-C2                          0x00000004

   VTD                   vtscsi0
   LUN                   0x8100000000000000
   Backing device        vtd0-1
   Physloc

   VTD                   vtscsi1
   LUN                   0x8200000000000000
   Backing device        vtd0-2
   Physloc

   VTD                   vtscsi2
   LUN                   0x8300000000000000
   Backing device        hdisk2
   Physloc               U787A.001.0397658-P1-T16-L5-L0
   ```
2. To list the shared Ethernet adapter and backing device mapped to the virtual server Ethernet adapter **ent4**, type:

   ```
   lsmap -vadapter ent4 -net
   ```

   The system displays a message similar to the following:

   ```
   SVEA   Physloc
   ------ --------------------------------------------
   ent4   P2-I1/E1

   SEA                   ent5
   ```

```
Backing device        ent1
Physloc               P2-I4/E1
```

3. To list the shared Ethernet adapter and backing device mapped to the virtual server Ethernet adapter **ent5** in script format separated by a : (colon), type:

   ```
   lsmap -vadapter ent5 -fmt ":"
   ```

   The system displays a message similar to the following:

   ```
   ent5:ent8:ent2
   ```
4. To list all virtual target devices and backing devices, where the backing devices are of type disk or lv, type:

   ```
   lsmap -all -type disk lv
   ```

   The system displays a message similar to the following:

   ```
   SVSA            Physloc                                      Client Partition ID
   --------------- -------------------------------------------- ------------------
   vhost0          U9117.570.10D1B0E-V4-C3                      0x00000000

   VTD                   vtscsi0
   LUN                   0x8100000000000000
   Backing device        hdisk0
   Physloc               U7879.001.DQD0KN7-P1-T12-L3-L0

   VTD                   vtscsi2
   LUN                   0x8200000000000000
   Backing device        lv04
   Physloc

   SVSA            Physloc                                      Client Partition ID
   --------------- -------------------------------------------- ------------------
   vhost1          U9117.570.10D1B0E-V4-C4                      0x00000000

   VTD                   vtscsi1
   LUN                   0x8100000000000000
   Backing device        lv03
   Physloc
   ```

Related Information

The **cfgdev** command, the **chdev** command, the **chpath** command, the **lsdev** command, the **lspath** command, the **mkpath** command, the **mkvdev** command, the **rmdev** command, the **rmpath** command.

lsnetsvc Command

Purpose

Displays the status of a network service.

Syntax

lsnetsvc *NetworkService*

Description

The **lsnetsvc** command displays the status of a network service. Use the *NetworkService* parameter to specify which service should have its status displayed.

Parameters

NetworkService	The following values may be used: inetd Returns the status of the **inetd** subsystem. The **inetd** subsystem must be in the active state for the **telnet** and **ftp** daemons to be active. If the **inetd** subsystem is in the inoperative state, executing the **startnetsvc** command with any of the supported network services will reactivate the **inetd** subsystem. telnet Returns the status of the **telnet** daemon ftp Returns the status of the **ftp** daemon

Exit Status

9	Invalid network service

Examples

1. To list the status of the **inetd** subsystem, type:

   ```
   lsnetsvc inetd
   ```

   This command will return either `active` or `not active`.
2. To list the status of the **telnet** daemon, type:

   ```
   lsnetsvc telnet
   ```

   This command will return either `active` or `not active`.
3. To list the status of the **ftp** daemon, enter:

   ```
    lsnetsvc ftp
   ```

   This command will return either `active` or `not active`.

Related Information

The **entstat** command, the **hostmap** command, the **hostname** command, the **mktcpip** command, the **netstat** command, the **optimizenet** command, the **startnetsvc** command, and the **stopnetsvc** command.

lspath Command

Purpose

Displays information about paths to a MultiPath I/O (MPIO) capable device.

Syntax

lspath **-dev** *DeviceName* **-pdev** *Parent* **-status** *Status* **-conn** *Connection* **-field** *FieldName* **-fmt** *Delimiter*

lspath **-dev** *DeviceName* **-pdev** *Parent* **-conn** *Connection* **-lsattr** **-attr** *Attribute*...

lspath **-dev** *DeviceName* **-pdev** *Parent* **-conn** *Connection* **-range** **-attr** *Attribute*

Description

The **lspath** command displays one of three types of information about paths to an MPIO capable device. It either displays the operational status for one or more paths to a single device, or it displays one or more attributes for a single path to a single MPIO capable device. The first syntax shown above displays the operational status for one or more paths to a particular MPIO capable device. The second syntax displays one or more attributes for a single path to a particular MPIO capable device. Finally, the third syntax displays the possible range of values for an attribute for a single path to a particular MPIO capable device.

Displaying Path Status with the lspath Command

When displaying path status, the set of paths to display is obtained by searching the device configuration database for paths that match the following criteria:

- The target device name matches the device specified with the **-dev** flag. If the **-dev** flag is not present, then the target device is not used in the criteria.
- The parent device name matches the device specified with the **-pdev** flag. If the **-pdev** flag is not present, then parent is not used in the criteria.
- The connection matches the connection specified with the **-conn** flag. If the **-conn** flag is not present, then connection is not used in the criteria.
- The path status matches status specified with the **-status** flag. If the **-status** flag is not present, the path status is not used in the criteria.

If none of the **-dev**, **-pdev**, **-conn**, or **-status** flags are specified, then all paths known to the system are displayed.

By default, this command will display the information in columnar form. When no flags are specified that qualify the paths to display, the format of the output is:

```
status device  parent
```

Possible values that can appear for the status column are:

enabled
    Indicates that the path is configured and operational. It will be considered when paths are selected for IO.

disabled
    Indicates that the path is configured, but not currently operational. It has been manually disabled and will not be considered when paths are selected for IO.

failed
    Indicates that the path is configured, but it has had IO failures that have rendered it unusable. It will not be considered when paths are selected for IO.

defined
    Indicates that the path has not been configured into the device driver.

missing
    Indicates that the path was defined in a previous boot, but it was not detected in the most recent boot of the system.

detected

Indicates that the path was detected in the most recent boot of the system, but for some reason it was not configured. A path should only have this status during boot and so this status should never appear as a result of the **lspath** command.

Displaying Path Attributes with the lspath Command

When displaying attributes for a path, the path must be fully qualified. Multiple attributes for a path can be displayed, but attributes belonging to multiple paths cannot be displayed in a single invocation of the **lspath** command. Therefore, in addition to the **-lsattr**, **-dev**, and **-pdev** flags, the **-conn** flags are required to uniquely identify a single path. For example:

- if only one path between a device and a specific parent, the **-conn** flag is not required
- if there are multiple paths between a device and a specific parent, the **-conn** flag is required

Furthermore, the **-status** flag is not allowed.

By default, this command will display the information in columnar form.

```
attribute   value    description         user_settable
```

Flags

-attr *Attribute*
Identifies the specific attribute to list. The '*Attribute*' is the name of a path specific attribute. When this flag is provided, only the identified attribute is displayed. Multiple instances of this flag may be used to list multiple attributes. If this flag is not specified at all, all attributes associated with the identified path will be listed.

-lsattr
Displays the attribute names, current values, descriptions, and user-settable flag values for a specific path.

-dev *Name*
Specifies the logical device name of the target device whose path information is to be displayed.

-field *FieldNames*
Specifies the list of fields to display. The following fields are supported:

status
Status of the path

name
Name of the device

parent
Name of the parent device

conn
Path connection.

-fmt *Delimiter*
Specifies a delimiter character to separate output fields.

-pdev *Parent*
Indicates the logical device name of the parent device of the path(s) to be displayed.

-range
Displays the legal values for an attribute name. The **-range** flag displays the list attribute values in a vertical column as follows:

```
Value1
Value2
.
.
ValueN
```

The **-range** flag displays the range attribute values as x...n(+i) where x is the start of the range, n is the end of the range, and i is the increment.

-status *Status*
The **-status** *Status* flag indicates the status to use in qualifying the paths to be displayed. When displaying path information, the allowable values for this flag are:

enabled
Display paths that are **enabled** for MPIO path selection.

disabled
Display paths that are **disabled** from MPIO path selection.

failed
Display paths that are **failed** due to IO errors.

available

Display paths whose **path_status** is **PATH_AVAILABLE** (that is, paths that are configured in the system, includes **enabled**, **disabled**, and **failed** paths).

defined
: Display paths whose **path_status** is **PATH_DEFINED**.

missing
: Display paths whose **path_status** is **PATH_MISSING**.

-conn *Connection*
: Indicates the connection information to use in qualifying the paths to be displayed.

Exit Status

Return code	Description
1	Invalid status value.

Examples

1. To display, without column headers, the set of paths whose operational status is disabled, enter:

   ```
   lspath -status disabled
   ```

   The system will display a message similar to the following:

   ```
    disabled  hdisk1   scsi1
   disabled  hdisk2   scsi1
   disabled  hdisk23  scsi8
   disabled  hdisk25  scsi8
   ```
2. To display the set of paths whose operational status is failed, enter:

   ```
    lspath -status failed
   ```

   The system will display a message similar to the following:

   ```
     failed  hdisk1   scsi1
   failed  hdisk2   scsi1
   failed  hdisk23  scsi8
   failed  hdisk25  scsi8
   ```
3. If the target device is a SCSI disk, to display all attributes for the path to parent scsi0 at connection 5,0, use the command:

   ```
    lspath -dev hdisk10 -pdev scsi0 -conn "5,0" -lsattr
   ```

   The system will display a message similar to the following:

   ```
   weight     1      Order of path failover selection  true
   ```

Related Information

The **lsmap** command, the **mkpath** command, the **chpath** command, and the **rmpath** command.

lspv Command

Purpose

Displays information about a physical volume within a volume group.

Syntax

lspv **-avail** | **-free** | **-size** **-field** *Fieldname*... **-fmt** *Delimiter*

lspv **-map** | **-lv** | **-pv** | **-size** **-field** *Fieldname* **-fmt** *Delimiter*

Description

The **lspv** command displays information about the physical volume if the specific physical volume name is given. If the **lspv** command is run without any arguments, the default is to print every known physical volume in the system along with its physical disk name, physical volume identifiers (PVIDs), to which volume group, if any, the physical volume belongs, and the state if the volume group is active.

When the *PhysicalVolume* parameter is used, the following characteristics of the specified physical volume are displayed:

Physical volume	Name of the physical volume
Volume group	Name of volume group. Volume group names must be unique systemwide names and can be from 1 to 15 characters long.
PV Identifier	The physical volume identifier for this physical disk.
VG Identifier	The volume group identifier of which this physical disk is a member.
PVstate	State of the physical volume. If the volume group that contains the physical volume is activated with the **activatevg** command, the state is active, missing, or removed. If the physical volume is deactivated with the **deactivatevg** command, the state is varied off.
Allocatable	Allocation permission for this physical volume.
Logical volumes	Number of logical volumes using the physical volume.
Stale PPs	Number of physical partitions on the physical volume that are not current.
VG descriptors	Number of volume group descriptors on the physical volume.
PP size	Size of physical partitions on the volume.
Total PPs	Total number of physical partitions on the physical volume.
Free PPs	Number of free physical partitions on the physical volume.
Used PPs	Number of used physical partitions on the physical volume.
Free distribution	Number of free partitions available in each intra-physical volume section.
Used distribution	Number of used partitions in each intra-physical volume section.

Flags

-avail Lists only physical volumes that are available for use as a backing device for virtual SCSI.

-free Lists only physical volumes that are available for use as a backing device for virtual SCSI and are not already backing devices.

-field *FieldNames* Specifies the list of fields to display. The following fields are supported if no physical volume is specified:

pvname
: Physical volume disk name

pvid
: Physical volume identifier

vgname
: Volume group the physical volume is in

pvstate
: Physical volume state (active, missing, removed, varied off)

The following fields are supported if a physical volume is specified:

pvname
: Physical volume disk name

vgname
: Volume group the physical volume is in

pvid
: Physical volume identifier

vgid
: Volume group identifier

pvstate
: Physical volume state (active, missing, removed, varied off)

allocatable
: Allocation permission for this physical volume.

stale
: Number of stale partitions on the disk

ppsize
: Physical partition size

numlv
: Number of logical volumes

size
: Number of physical partitions and total disk size

vgds
: Number of volume group descriptor areas within the volume group.

free
: Number of free partitions and free space

pvused
: Number of used partitions and used space

maxreq
: Maximum transfer size of physical volume

freedist
: Number of free partitions available in each intra-physical volume section.

usedist
: Number of used partitions in each intra-physical volume section

hotspare

The following fields are supported if the **-lv** flag is specified:

lvname
: Name of the logical volume to which the physical partitions are allocated.

lps
: The number of logical partitions within the logical volume that are contained on this physical volume.

pps
: The number of physical partitions within the logical volume that are contained on this physical volume.

dist
: The number of physical partitions, belonging to the logical volume, that are allocated within each of the following sections of the physical volume: outer edge, outer middle, center, inner middle and inner edge of the physical volume.

mount
: File system mount point for the logical volume, if applicable.

-field *FieldNames* (continued)

The following fields are supported if the **-pv** flag is specified:

range
: A range of consecutive physical partitions contained on a single region of the physical volume.

ppstate
: The current state of the physical partitions: `free`, `used`, `stale`, or `vgda`

region
: The intra-physical volume region in which the partitions are located.

lvname
: The name of the logical volume to which the physical partitions are allocated.

type
: The type of the logical volume to which the partitions are allocated.

mount
: File system mount point for the logical volume, if applicable.

The following fields are supported if the **-map** flag is specified:

physical
: Physical volume name and physical partition number.

logical
: Logical volume name and logical partition number. If mirrored the mirror number is also shown. If the partition is stale this is also shown.

The following fields are supported if the **-free** or **-avail** flag is specified:

pvname
: Physical volume disk name.

pvid
: Physical volume identifier

size
: Size of the physical volume.

The following fields are supported if the **-size** flag and a physical volume are specified:

pvname
: Physical volume disk name.

pvid
: Physical volume identifier

size
: Size of the physical volume.

-fmt *Delimiter*
: Specifies a delimiter character to separate output fields.

-lv
: Lists the following fields for each logical volume on the physical volume:

LVname
: Name of the logical volume to which the physical partitions are allocated.

LPs
: The number of logical partitions within the logical volume that are contained on this physical volume.

PPs
: The number of physical partitions within the logical volume that are contained on this physical volume.

Distribution
: The number of physical partitions, belonging to the logical volume, that are allocated within each of the following sections of the physical volume: outer edge, outer middle, center, inner middle and inner edge of the physical volume.

Mount Point
: File system mount point for the logical volume, if applicable.

-map
: Lists the following fields for each logical volume on the physical volume: PVname:PPnum LVname: LPnum :Copynum PPstate Where:

PVname
: Name of the physical volume as specified by the system.

PPnum
: Physical partition number.

LVname
: Name of the logical volume to which the physical partitions are allocated. Logical volume names must be system-wide unique names, and can range from 1 to 64 characters.

LPnum
: Logical partition number. Logical partition numbers can range from 1 to 64,000.

Copynum
: Mirror number.

PPstate
: Only the physical partitions on the physical volume that are not current are shown as stale.

-pv
: Lists the following fields for each physical partition on the physical volume:

Range

A range of consecutive physical partitions contained on a single region of the physical volume.

State

The current state of the physical partitions: free, used, stale, or vgda. **Note:** If a volume group is converted to a big vg format, it may be necessary to use some data partitions for volume group descriptor area. These partitions will be marked vgda.

Region

The intra-physical volume region in which the partitions are located.

LVname

The name of the logical volume to which the physical partitions are allocated.

Type

The type of the logical volume to which the partitions are allocated.

Mount Point

File system mount point for the logical volume, if applicable.

-size Displays the size of one or all physical volumes in megabytes.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To display the status and characteristics of physical volume **hdisk3**, type:

   ```
   lspv hdisk3
   ```
2. To display all physical volumes in the system, type:

   ```
   lspv
   ```

   You should see output similar to the following:

   ```
   hdisk0   0000000012345678              rootvg   active
   hdisk1   10000BC876543258              vg00     active
   hdisk2   ABCD000054C23486              None
   ```

   The previous example shows that physical volume **hdisk0** contains the volume group rootvg, and it is activated. Physical volume **hdisk1** contains the volume group **vg00**, and it is activated. Physical volume **hdisk2** does not contain an active volume group.
3. To display all physical volumes that can be virtual SCSI backing devices, type:

   ```
   lspv -avail
   ```

   Output similar to the following is displayed:

   ```
   hdisk1   0000000012345678   None   vhost0
   hdisk2   0000000012345678   None   vhost3 vhost4
   hdisk3   10000BC876543258   None   None
   hdisk4   ABCD000054C23486   None   None
   ```

   This example shows that physical volume **hdisk1** is not in the rootvg volume group, and is a backing device for the virtual SCSI adapter vhost0. Physical volume hdisk2 is a backing device for virtual SCSI adapters vhost3 and vhost4. Physical volumes hdisk3 and hdisk4 are not associated with any virtual SCSI adapters.
4. To display all physical volumes that can be virtual SCSI backing devices and are not currently a backing device, type:

   ```
   lspv -free
   ```

   Output similar to the following is displayed:

   ```
   hdisk3   10000BC876543258   None   None
   hdisk4   ABCD000054C23486   None   None
   ```

Related Information

The **migratepv** command.

IVM lsrefcode Command

Purpose

Lists reference codes for partitions or the managed system. This command is valid only in an Integrated Virtualization Manager environment.

Syntax

To list reference codes for the managed system:

lsrefcode -r sys **-n** *Number* **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list reference codes for partitions:

lsrefcode -r lpar **-n** *Number* **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

Description

The **lsrefcode** command lists reference codes for partitions or the managed system.

Flags

-r *ResourceType*
The type of resources to list:

- **-r sys**: List reference codes for the managed system.
  - **Attributes**: refcode_num, time_stamp, refcode, word2, word3, word4, word5, word6, word7, word8, word9, fru_call_out_loc_codes
  - **Filters**: None
- **-r lpar**: List reference codes for partitions.
  - **Attributes**: lpar_name, lpar_id, time_stamp, refcode, word2, word3, word4, word5, word6, word7, word8, word9, fru_call_out_loc_codes
  - **Filters**: { lpar_ids | lpar_names}

-n *Number*
The number of reference codes to list. The default is to list one. Reference codes are listed in order, with the most recent reference code first.

-m *ManagedSystem*
The name of the managed system. This attribute is optional because there is only one system to manage. The name may either be the user-defined name for the managed system, or be in the form tttt-mmm*ssssssss, where tttt is the machine type, mmm is the model, and ssssssss is the serial number of the managed system.

--filter *FilterData*
The filters to apply to the resources to be listed. Filters are used to select which resources of the specified resource type are to be listed. If no filters are used, then all of the resources of the specified resource type will be listed. For example, specific partitions can be listed by using a filter to specify the names or IDs of the partitions to list. Otherwise, if no filter is used, then all the partitions in the managed system will be listed.

The filter data consists of filter name/value pairs, which are in comma separated value (CSV) format. The filter data must be enclosed in double quotes.

The format of the filter data is as follows:

```
"filter-name=value,filter-name=value,..."
```

Certain filters accept a comma separated list of values, as follows:

```
""filter-name=value,value,...",..."
```

When a list of values is specified, the filter name/value pair must be enclosed in double quotes. Depending on the shell being used, nested double quote characters may need to be preceded by an escape character, which is usually a '\' character.

Unless otherwise indicated, multiple values can be specified for each filter.

Valid filter names for -r lpar:

lpar_ids
: ID of the partition to view

lpar_names
: Name of the partitions to view

-F *AttributeNames*

A delimiter separated list of attribute names for the desired attribute values to be displayed for each resource. If no attribute names are specified, then values for all of the attributes for the resource will be displayed.

When this option is specified, only attribute values will be displayed. No attribute names will be displayed. The attribute values displayed will be separated by the delimiter which was specified with this option.

This option is useful when only attribute values are desired to be displayed, or when the values of only selected attributes are desired to be displayed.

Attribute Names:

fru_call_out_loc_codes
: Location codes of the field replaceable unit related to the reference code

lpar_id
: Unique integer identifier for the partition

lpar_name
: Name of the partition

refcode
: The ASCII reference code string

refcode_num
: The sequence number of the reference code.

time_stamp
: The time that the reference code was created in the format: MM/DD/YYYY HH:MM:SS where MM is the two digit month, DD is the two digit day, YYYY is the four digit year, HH is the two digit hour, MM is the two digit minute, and SS is the two digit second.

fru_call_out_loc_codes
: The Field Replaceable Unit (FRU) numbers for the FRUs whose absence or failure caused this reference code to be received. This field may be used for other values.

--header

Display a header record, which is a delimiter separated list of attribute names for the attribute values that will be displayed. This header record will be the first record displayed. This option is only valid when used with the -F option.

Exit Status

This command has a return code of zero on success.

Security

This command is accessible by all users.

Examples

1. To list the current reference code for the managed system, type:

   ```
   lsrefcode -r sys
   ```
2. To list the current reference code for all partitions, type:

   ```
   lsrefcode -r lpar
   ```
3. To list the last 25 reference codes for partitions p1 and p2, only viewing the lpar_id and refcode attributes, type:

   ```
   lsrefcode -r lpar -n 25 --filter \"lpar_names=p1,p2\" -F lpar_id,refcode
   ```

lssp Command

Purpose

Lists and displays information about storage pools.

Syntax

List all available storage pools

lssp **-field** *Fieldname* **-fmt** *Delimiter*

Display information about a specific storage pool

lssp **-detail** | **-bd** **-sp** *StoragePool* **-field** *Fieldname* **-fmt** *Delimiter*

Display the default storage pool

lssp **-default**

Description

The **lssp** command displays information about storage pools in the Virtual I/O Server. If no flags are specified, a list of all defined storage pools, their total size, free space, minimum allocation size, and number of backing devices contained in the pool is displayed. If the **-detail** flag is specified, detailed information about the storage pool is displayed. If the **-bd** flag is specified, a list of all the backing devices in the specified (or default) storage pool is displayed along with their size and associated virtual target device, or None, and virtual host adapter, or None. If the **-default** flag is specified, the default storage pool is displayed.

This command will provide full scripting support through the use of the **-field** and **-fmt** flags.

If the storage pool specified is not prefixed with `sp_`, the **lssp** command will automatically add it. If rootvg is specified, no prefix is added.

Flags

-default
Displays the default storage pool.

-field *FieldName*
The following fields are supported if no flags are specified:

pool
Storage pool name

size
Total size

free
Free space

alloc
Minimum allocation size

bds
Number of physical volumes

lvs
Number of backing devices

The following fields are supported if the **-detail** flag is specified:

pvname
Name of the physical volume.

pvid
Physical volume identifier

size
The size of the physical volume.

The following fields are supported if the **-bd** flag is specified:

bdname
: Logical partition number.

size
: The size of the logical volume.

vtd
: Virtual target device.

svsa
: Server virtual SCSI adapter.

-fmt *Delimiter* Specifies a delimiter character to separate output fields.

-bd Displays information about the backing devices in the storage pool.

-detail Displays information about the physical volumes in the storage pool.

-sp *StoragePool* Specifies which storage pool to display information about.

Examples

1. To list all storage pools, type:

   ```
   lssp
   ```

   The system displays output similar to the following:

   ```
   POOL            SIZE(mb)   FREE(mb)   ALLOC SIZE(mb)  BDs
   sp_test         34624      34624      32              1
   rootvg          34688      17024      64              1
   ```
2. To display the default storage pool, type:

   ```
   lssp -default
   ```
3. To display detailed information about the storage pool sp_sp00, type:

   ```
   lssp -detail -sp sp_sp00
   ```

   The system will display output similar to the following:

   ```
   NAME            PVID                                SIZE(megabytes)
   hdisk3          00cdfd8c85bd4b2e                    34624
   hdisk2          00cdfd8c525d94a2                    34624
   ```
4. To display information about the backing devices in the storage pool rootvg, type:

   ```
   lssp -bd -sp rootvg
   ```

   The system displays output similar to the following:

   ```
   NAME            SIZE(megabytes) VTD             SVSA
   lv01            96              vtscsi1         vhost0
   lv02            64              vtscsi2         vhost0
   ```

IVM lssvcevents Command

Purpose

List attributes of console or serviceable events. This command is valid only in an Integrated Virtualization Manager environment.

Syntax

To list console events:

lssvcevents -t console **-d** *NumberDays* | **-i** *NumberMinutes* **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list serviceable events:

lssvcevents -t hardware **-d** *NumberDays* | **-i** *NumberMinutes* **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list field replaceable units (frus) for a specific serviceable events

lssvcevents -t fru **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list comments for a specific serviceable event

lssvcevents -t comment **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list service objects associated with a specific serviceable event

lssvcevents -t service_object **-filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list the status of dlpar events:

lssvcevents **-t** dlpar **-d** *NumberDays* | **-i** *NumberMinutes* **--filter** "FilterData" **-F** "AttributeNames" **--header -m** *ManagedSystem*

Description

The **lssvcevents** command lists attributes of console or serviceable events.

Flags

-t *EventType*

The type of event to list:

console
Console events - events created by the applications
◊ **Attributes**: time, userid, pid, name, category, severity
◊ **Filters**: severities, categories, name

hardware
Serviceable events
◊ **Attributes**: problem_num, pmh_num, refcode, status, first_time, last_time, sys_name, reporting_name, sys_mtms, reporting_mtms, enclosure_mtms, failing_mtms, text, firmware_fix, created_time, analyzing_sfp, refcode_extension, firmware_pkg_name, firmware_pkg_status, reporting_sfp_name, reporting_sfp_mtms, failing_sfp_mtms, severity, lpar_id, lpar_name, lpar_hostname, lpar_os_type, notification_type, notification_status, duplicate_count, analyzing_sfp_mtms, analyzing_sfp_name, called_home_sys_mtms, sys_log_id, platform_log_id, subsystem_id, creator_id, ipl_state, symptom,

failing_lpar_id, failing_lpar_name, failing_lpar_os_type, fru_part_nums, fru_phys_locs

◊ **Filters**: status, problem_nums, refcodes, fru_part_nums, fru_phys_locs, reporting_mtms, failing_mtm

fru

Field replaceable units (frus) for a serviceable event.

◊ **Attributes**: part_num, class, description, phys_loc, prev_replaced, replaced_time, serial_num, replacement_grp, ccin, logic_ctl_mtms, power_ctl_mtms

◊ **Filters**: problem_nums (required)

comment

Comments for a serviceable event.

◊ **Attributes**: time,commenter,text

◊ **Filters**: problem_nums (required), status

service_object

Service objects for a serviceable event. These are not accessible to users without the DEUser or SRUser roles.

◊ **Attributes**: key, sys_log_id, notification_type, platform_log_id, severity, creator_id, refcode, subsystem_id, lpar_os_type

◊ **Filters**: problem_nums (required), status

dlpar

Status of DLPAR events - events used to synchronize resources that can be dynamically configured while the partition is active. These can be used to determine why the runtime and pending values of a resource are not synchronized. Typically, the last two status records are stored per partition per resource type.

◊ **Attributes**: lpar_id, resource_type, sequence_num, status_code, time, internal_rc, drmgr_cmd, drmgr_rc, drmgr_stdout, drmgr_stderr

◊ **Filters**: lpar_ids, resource_types

-d *NumberDays*
The number of previous days for which to view events. This may not be used with the **-i** flag, and is only applicable to **-t** *console*, and **-t** *hardware*. If this flag and the **-i** flag are omitted, the default is 7 days.

-i *NumberMinutes*
The number of minutes for which to view events. This may not be used with the **-d** flag, and is only applicable to **-t** *console*, and **-t** *hardware*.

-m *ManagedSystem*
The name of the managed system. This attribute is optional because there is only one system to manage. The name may either be the user-defined name for the managed system, or be in the form tttt-mmm*ssssssss, where tttt is the machine type, mmm is the model, and ssssssss is the serial number of the managed system.

--filter *FilterData*
The filters to apply to the resources to be listed. Filters are used to select which resources of the specified resource type are to be listed. If no filters are used, then all of the resources of the specified resource type will be listed. For example, specific partitions can be listed by using a filter to specify the names or IDs of the partitions to list. Otherwise, if no filter is used, then all the partitions in the managed system will be listed.

The filter data consists of filter name/value pairs, which are in comma separated value (CSV) format. The filter data must be enclosed in double quotes.

The format of the filter data is as follows:

```
"filter-name=value,filter-name=value,..."
```

Note that certain filters accept a comma separated list of values, as follows:

```
""filter-name=value,value,...",..."
```

When a list of values is specified, the filter name/value pair must be enclosed in double quotes. Depending on the shell being used, nested double quote characters may need to be preceded by an escape character, which is usually a '\' character.

Unless otherwise indicated, multiple values can be specified for each filter.

Valid filter names for -t console:

severities, categories, name

Valid filter names for -t hardware:

status, problem_nums, refcodes, fru_part_nums, fru_phys_locs, reporting_mtms, failing_mtms

Valid filter names for -t fru:

problem_nums (required)

Valid filter names for -t comments and -t service_objects:

problem_nums (required), status

Valid filter names for -t dlpar:

lpar_ids, resource_types

-F *AttributeNames*

A delimiter separated list of attribute names for the desired attribute values to be displayed for each resource. If no attribute names are specified, then values for all of the attributes for the resource will be displayed.

When this option is specified, only attribute values will be displayed. No attribute names will be displayed. The attribute values displayed will be separated by the delimiter which was specified with this option.

This option is useful when only attribute values are desired to be displayed, or when the values of only selected attributes are desired to be displayed.

Attribute names:

analyzing_sfp
Name or MTMS of the service focal point system analyzing the event.

analyzing_sfp_mtms
MTMS of the service focal point system analyzing the event.

analyzing_sfp_name
Name of the service focal point system analyzing the event.

called_home_sys_mtms
MTMS of the system to which the event was called home.

category
Category or client type of the console event. Valid values:

◊ GUI: Web interface
◊ CLI: Command line interface
◊ AP: Access process

ccin
CCIN of the FRU

class
Class or type of the FRU

commenter
Name of the person adding a comment to the event

created_time
Time the event was created.

creator_id
The ID of the entity that created the event. Valid values:

◊ C: Hardware Management Console
◊ E: Service processor

◊ H: POWER Hypervisor
◊ W: Power
◊ L: Partition firmware

description
: Description of the FRU

drmgr_cmd
: The command used for a dlpar event. The drmgr command is run on the client partition's operating system to synchronize a particular resource.

drmgr_rc
: The return code for the command used for a dlpar event. The drmgr command is run on the client partition's operating system to synchronize a particular resource.

drmgr_stdout
: The standard output for the command used for a dlpar event. The drmgr command is run on the client partition's operating system to synchronize a particular resource.

drmgr_stderr
: The standard error for the command used for a dlpar event. The drmgr command is run on the client partition's operating system to synchronize a particular resource.

duplicate_count
: Number of duplicates for this event.

enclosure_mtms
: Enclosure MTMS

failing_lpar_id
: Unique ID for the failing partition

failing_lpar_name
: Name of the failing partition

failing_lpar_os_type
: Operating system type of the failing partition

failing_mtms
: MTMS of the failing system

failing_sfp_mtms
: MTMS of the failing systems service focal point

firmware_fix
: Indicates whether a firmware fix is available for the event

firmware_pkg_name
: Package name of a possible firmware fix

firmware_pkg_status
: Package status of a possible firmware fix

first_time
: First time this event was reported

fru_part_nums
: Part numbers of the FRUs

fru_phys_locs
: Unique physical location codes of the FRUs

internal_rc
The return code for a dlpar event. This will be non zero only if the DLPAR command was unexpectedly not able to be sent to the client partition.

ipl_state
State of the system when this event occurred

key
Unique ID for a service object for a particular serviceable event.

last_time
Last time this event was reported

logic_ctl_mtms
MTMS of the unit that logically controls the unit that the FRU is located in

lpar_hostname
Hostname of the partition that created this event

lpar_id
ID of the partition that created this event

lpar_name
Name of the partition that created this event

lpar_os_type
Operating system type of the partition that created this event

name
Name of the application that created the console event

notification_status
The status of the notification type

notification_type
The notification type for the event. Valid values:
◊ Yes: Call home
◊ No: Customer notify

part_num
Part number of the FRU

phys_loc
Unique physical location code of the FRU

pid
Process ID of the process generating the console event

platform_log_id
Unique ID of the platform log for this event

pmh_num
PMH or tracking number

power_ctl_mtms
MTMS of the unit that power controls the unit that the FRU is located in

prev_replaced
Indicates if the FRU has been previously replaced

problem_num
A unique ID for the event

refcode

Reference code for the event

refcode_extension

Extended reference code for the event

replaced_time

Time the FRU was replaced

replacement_grp

Replacement priority and grouping for the FRU. Valid values:

- ◊ H: Multiple high priority FRUs should be acted on as a group
- ◊ M: Medium priority FRUs should be acted on, one at a time, in the order given.
- ◊ A: Medium priority group A FRUs should be acted on as a group.
- ◊ B: Medium priority group B FRUs should be acted on as a group.
- ◊ C: Medium priority group C FRUs should be acted on as a group.
- ◊ L: Low priority FRUs should be acted on only after all other priority call-outs failed to resolve the problem.

reporting_mtms

MTMS of the reporting unit

reporting_name

Name of the reporting unit

reporting_sfp_mtms

MTMS of the service focal point reporting the event

reporting_sfp_name

Name of the service focal point reporting the event

resource_type

The resource type for the DLPAR event. These are all the resources that can be dynamically reconfigured. Valid values:

- ◊ mem
- ◊ proc
- ◊ proc_units
- ◊ uncap_weight
- ◊ memory

sequence_num

The sequence number for this DLPAR event. Each resource type may have more than one entry. The sequence number distinguishes the entries, and increments over time.

serial_num

Serial number of the FRU

severity

Severity of the event. Valid values:

- ◊ 10: Recovered error, general (10)
- ◊ 20: Predictive error, general (20)
- ◊ 21: Predictive error, degraded performance (21)
- ◊ 22: Predictive fault may be corrected after platform re-IPL (22)
- ◊ 23: Predictive error, fault may be corrected after IPL, degraded performance (23)
- ◊ 24: Predictive error, loss of redundancy (24)
- ◊ 40: Unrecovered error, general (40)
- ◊ 41: Unrecovered error, bypassed with degraded performance (41).
- ◊ 44: Unrecovered error, bypassed with loss of redundancy (44)
- ◊ 45: Unrecovered error, bypassed with loss of redundancy and performance (45)
- ◊ 48: Unrecovered error, bypassed with loss of function (48)
- ◊ 60: Error on diagnostic test, general (60)

◊ 61: Error on diagnostic test, resource may produce incorrect result (61)

status
: Status of the event. Valid values:
◊ Open: Event is in the open state
◊ Closed: Event has been closed

status_code
: Status code for the event. Valid values for dlpar events:
◊ 0: Synchronization successful
◊ 1: Synchronization in progress
◊ 2: Resource will not synchronize because the partition is a workload group participant
◊ 3: Resource will not synchronize because the partition communication state is not active
◊ 4: Resource will not synchronize because the partition does not support DLPAR of this resource type
◊ 5: Resource will not synchronize because the partition is not in the Running state
◊ 6: Resource will not synchronize because the partition is unable to remove any more memory dynamically
◊ 7: Resource will not synchronize because the synchronization command failed to run for an unknown reason
◊ 8: Resource is not synchronized because the RMC command failed. The system will retry. If the partition is in the Running state with an active network connection, check the return code and contact your support representative.
◊ 9: Resource is not synchronized because the drmgr command on the partition failed. The system will retry. Check the return code, and the command output.
◊ 10: Resource will not synchronize because the requested assigned value is less than the current minimum. Restart your partition in order to complete the synchronization.
◊ 11: Resource will not synchronize because the requested assigned value is greater than the current maximum. Restart your partition in order to complete the synchronization.
◊ 12: Resource will not synchronize because the pending and current processing modes do not match. Restart your partition in order to complete the synchronization.
◊ 255: Resource synchronization has not yet been attempted. It may take a few seconds depending on your system utilization before synchronization is attempted.

subsystem_id
: The subsystem causing the event. Valid values include:
◊ 10 - 1F: Processor subsystem including internal cache
◊ 20 - 2F: Memory subsystem including external cache
◊ 30 - 3F: I/O subsystem (hub, bridge, bus)
◊ 40 - 4F: I/O adapter, device and peripheral
◊ 50 - 5F: CEC hardware
◊ 60 - 6F: Power/Cooling subsystem
◊ 70 - 79: Other subsystem
◊ 7A - 7F: Surveillance error
◊ 80 - 8F: Platform firmware
◊ 90 - 9F: Software
◊ A0 - AF: External environment

symptom
: Symptom of the event

sys_log_id
: Unique ID of the system log for the event

sys_mtms
: MTMS of the system

text
: Text of the event

time
: Time of the console or dlpar event

userid
: User ID of the user that ran the command that caused the console event

--header
: Display a header record, which is a delimiter separated list of attribute names for the attribute values that will be displayed. This header record will be the first record displayed. This option is only valid when used with the **-F** option.

Exit Status

This command has a return code of zero on success.

Security

This command is accessible by all users, except for certain types and attributes as noted, which require SR User or DE User roles.

Examples

1. To list the serviceable events that occurred today, type:

   ```
   lssvcevents -t hardware -d 0
   ```
2. To list the console events that occurred within the past 3 days, type:

   ```
   lssvcevents -t console -d 3
   ```
3. To list all of the open serviceable events for the system, type:

   ```
   lssvcevents -t hardware --filter "status=open"
   ```
4. To list the associated FRUs for a specific serviceable event, type:

   ```
   lssvcevents -t fru --filter problem_nums=6013EFFF-205E9F22-4CC931E5-F892358-A0F6C1D6
   ```

Related Information

The **chsvcevent** command, and the **mksvcevent** command.

lssw Command

Purpose

Lists installed software products.

Syntax

lssw -hist

Description

The **lssw** command displays information about installed file sets or file set updates. If the **-hist** parameter is not specified, the name, most recent level, state, and description of all file sets is displayed. Part information (usr, root, and share) is consolidated into the same listing. For formatted file sets, it displays the most recent maintenance level. Any interim fixes on the system are also displayed.

If the **-hist** flag is specified, installation and update history information is displayed.

Output Values

The following sections define terms used in several of the output fields. Note that not all output values are defined here. Only the ones that require explanation are defined.

State Values

The **state** field in the **lssw** output gives the state of the fileset on your system. It can have the following values:

State	Description
APPLIED	The specified fileset is installed on the system. The **APPLIED** state means that the fileset can be removed with the **updateios** command and the previous level of the fileset restored.
APPLYING	An attempt was made to apply the specified fileset, but it did not complete successfully, and cleanup was not performed.
BROKEN	The specified fileset or fileset update is broken and should be reinstalled before being used.
COMMITTED	The specified fileset is installed on the system. The **COMMITTED** state means that a commitment has been made to this level of the software. A committed fileset update cannot be rejected, but a committed fileset base level and its updates (regardless of state) can be removed by the **updateios** command.
EFIX LOCKED	The specified fileset was installed successfully and locked.
OBSOLETE	The specified fileset was installed with an earlier version of the operating system but has been replaced by a repackaged (renamed) newer version. Some of the files that belonged to this fileset have been replaced by versions from the repackaged fileset.
COMMITTING	An attempt was made to commit the specified fileset, but it did not complete successfully, and cleanup was not performed.
REJECTING	An attempt was made to reject the specified fileset, but it did not complete successfully, and cleanup was not performed.

Action Values

The **action** field in the **lssw** output identifies the installation action that was taken for the fileset. The following values may be found in this field:

Action	Definition

APPLY	An attempt was made to apply the specified fileset.
CLEANUP	An attempt was made to perform cleanup for the specified fileset.
COMMIT	An attempt was made to commit the specified fileset.
REJECT	An attempt was made to reject the specified fileset.

Status Values

The **status** field in the **lssw** output identifies the resultant status in the history of installation actions. The following values may be found in this field:

Status	**Description**
BROKEN	The fileset was left in a broken state after the specified action.
CANCELED	The specified action was canceled before it completed.
COMPLETE	The commitment of the fileset has completed successfully.

Flags

-hist Displays the installation and update history information.

Exit Status

See Virtual I/O Server command exit status

Examples

1. To list all installed software, type:

   ```
   lssw
   ```
2. To list installation and update history, type:

   ```
   lssw -hist
   ```

Related Information

The **updateios** command, the **ioslevel** command, the **remote_management** command, the **oem_setup_env** command, and the **oem_platform_level** command.

IVM lssyscfg Command

Purpose

List attributes of partitions, partition profiles, or the managed system. This command is valid only in an Integrated Virtualization Manager environment.

Syntax

To list partition attributes:

lssyscfg -r lpar **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list partition profile attributes:

lssyscfg -r prof **--filter** "*FilterData*" **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

To list system attributes:

lssyscfg -r sys **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

Description

The **lssyscfg** command lists attributes of partitions, partition profiles, or the managed system.

Flags

-r *ResourceType*
The type of resources to list: `lpar`: Logical partition resources
`prof`: Logical partition profile resources
`sys`: Managed system resources

-m *ManagedSystem*
The name of the managed system. This attribute is optional because there is only one system to ma The name may either be the user-defined name for the managed system, or be in the form tttt-mmm*ssssssss, where tttt is the machine type, mmm is the model, and ssssssss is the serial nur of the managed system.

--filter *FilterData*
The filters to apply to the resources to be listed. Filters are used to select which resources of the spe resource type are to be listed. If no filters are used, then all of the resources of the specified resourc will be listed. For example, specific partitions can be listed by using a filter to specify the names or ID the partitions to list. Otherwise, if no filter is used, then all the partitions in the managed system will b listed.

The filter data consists of filter name/value pairs, which are in comma separated value (CSV) format filter data must be enclosed in double quotes.

The format of the filter data is as follows:

```
"filter-name=value,filter-name=value,..."
```

Note that certain filters accept a comma separated list of values, as follows:

```
""filter-name=value,value,...",..."
```

When a list of values is specified, the filter name/value pair must be enclosed in double quotes. Depe on the shell being used, nested double quote characters may need to be preceded by an escape character, which is usually a '\' character.

Unless otherwise indicated, multiple values can be specified for each filter.

Valid filter names for partitions:

`lpar_names | lpar_ids` - name or ID of the partitions to view
`work_groups` - work groups to which the partitions belong

Valid filter names for partition profiles:

`lpar_names | lpar_ids`: name or ID of the partition profiles to view
`profile_names`: profile names for the partitions
Note: This option is not valid when listing managed systems.

-F *AttributeNames*

A delimiter separated list of attribute names for the desired attribute values to be displayed for each resource. If no attribute names are specified, then values for all of the attributes for the resource will be displayed.

When this option is specified, only attribute values will be displayed. No attribute names will be displayed. The attribute values displayed will be separated by the delimiter which was specified with this option.

This option is useful when only attribute values are desired to be displayed, or when the values of only selected attributes are desired to be displayed.

Attribute names for partitions:

name
    Name of the partition
lpar_id
    unique integer identifier for the partition
lpar_env
    The operating environment for this partition Valid values:
    ◊ aixlinux: a partition type that supports AIX or Linux
    ◊ vioserver: a Virtual I/O Server partition
state
    The current runtime state of the partition Valid values:
    ◊ Not Activated
    ◊ Starting
    ◊ Running
    ◊ Shutting Down
    ◊ Error
    ◊ Open Firmware
    ◊ Not Available
resource_config
    Valid values:
    ◊ 0 - resources are not available to power on with system
    ◊ 1 - resources are available to power on with system
os_version
    The operating system's version identifier
logical_serial_num
    A globally unique string for this partition
default_profile
    The default profile is always equal to the partition name
curr_profile
    The current profile is always equal to the partition name
work_group_id
    Valid values:
    ◊ none: do not participate in the workload management group
    ◊ 1: participate in the workload management group
shared_proc_pool_util_auth
    Permission for the partition to retrieve shared processor pool utilization information Valid values:
    ◊ 0: do not allow authority
    ◊ 1: allow authority

power_ctrl_lpar_ids
    A list of partitions which have power control over this partition. Valid values:
    ◊ none: No partitions
boot_mode
    Partition power on mode. Valid values:
    ◊ norm: normal
    ◊ dd: diagnostic with default boot list
    ◊ ds: diagnostic with stored boot list

◊ of: Open Firmware OK prompt
◊ sms: System Management Services

lpar_keylock
Partition keylock position. Valid values:
◊ norm: normal keylock
◊ manual: manual keylock

auto_start
Valid values:
◊ 0 - do not automatically start with system power on
◊ 1 - automatically start with system power on

uptime
Partition uptime in seconds.

mem_synchronized
The current and pending memory values for this partition are synchronized.

proc_synchronized
The current and pending processing values for this partition are synchronized.

rmc_state
The state of the RMC connection between the management partition and the client partition. RMC connection is used primarily for Dynamic Logical Partitioning (DLPAR). Valid values:
◊ inactive
◊ active
◊ unknown
◊ none - RMC not configured. This partition has never been registered with RMC.

rmc_ipaddr
The IP address of the client partition. This IP address is used by RMC to connect to the clier partition for Dynamic Logical Partitioning (DLPAR).

rmc_osshutdown_capable
Indicates if the partition supports shutdown via the RMC connection. This allows the manage partition to safely shutdown the client partition via chsysstate -o osshutdown Valid values:
◊ 0 - no
◊ 1 - yes

dlpar_mem_capable
Indicates if the partition supports Dynamic Logical Partitioning (DLPAR) of memory. Valid va
◊ 0 - no
◊ 1 - yes

dlpar_proc_capable
Indicates if the partition supports Dynamic Logical Partitioning (DLPAR) of processing resou Valid values:
◊ 0 - no
◊ 1 - yes

Attribute names for partition profiles:

name
Name of the profile

lpar_name
Name of the partition

lpar_id
Unique integer identifier for the partition

os_type
The operating system environment for this partition. Valid values:
◊ aixlinux: an RPA partition type which supports AIX or Linux
◊ vioserver: a Virtual I/O Server partition

all_resources
Valid values:
◊ 0: this partition will not own all physical resources on the system

min_mem
The minimum megabytes of memory for this partition

desired_mem
The assigned megabytes of memory for this partition

max_mem
The maximum megabytes of memory for this partition

proc_mode
Valid values:

◊ ded: dedicated processor mode
◊ shared: shared processor mode

min_proc_units
: The minimum number of processing units for this partition

desired_proc_units
: The assigned number of processing units for this partition

max_proc_units
: The maximum number of processing units for this partition

min_procs
: The minimum number of processors for this partition. In shared processing mode, this refers to virtual processors.

desired_procs
: The assigned number of processors for this partition. In shared processing mode, this refers to virtual processors.

max_procs
: The maximum number of processors for this partition. In shared processing mode, this refers to virtual processors.

sharing_mode
: Valid values:

◊ keep_idle_procs - valid with dedicated processor mode
◊ share_idle_procs - valid with dedicated processor mode
◊ cap - capped mode. valid with shared processor mode
◊ uncap - uncapped mode. valid with shared processor mode

uncap_weight
: A weighted average of processing priority when in uncapped sharing mode. The smaller the value, the lower the weight. Valid values are: 0 - 255

io_slots
: Comma separated list of I/O slots for the partition. Each item in this list has the format: `drc_index/slot_io_pool_id/is_required`

The attribute names are not present in the list, just their values are present. For example, `21010003/none/1`, specifies an I/O slot with a DRC index of 0x21010003 which is not assigned to an I/O pool, and it is a required slot.

Valid values for `is_required`:

◊ 0 - no
◊ 1 - yes

lpar_io_pool_ids
: Valid values are:

◊ none - this partition is not part of an I/O pool

max_virtual_slots
: Maximum number of virtual I/O adapter slots

virtual_serial_adapters
: Comma separated list of virtual serial adapters. Each item in this list has the format:

```
slot_num/adapter_type/supports_hmc/remote_lpar_id/remote_lpar_name/remote_slot_num/is_req
```

The attribute names are not present in the list, just their values are present. If an attribute is optional and is not to be included, then no value would be specified for that attribute. For example, `0/server/1/any//any/1` specifies a virtual server serial device that has a virtual slot number of 0, supports HMC, supports any remote partition, supports any remote slot, and is required.

A value of `none` indicates that there are no virtual serial adapters.

Valid values for `adapter_type`:

◊ client: client adapter
◊ server: server adapter, valid for Virtual I/O Server partitions only

Valid values for `supports_hmc`:

◊ 0 - no
◊ 1 - yes

Valid values for `is_required`:

◊ 0 - no
◊ 1 - yes

virtual_scsi_adapters

Comma separated list of virtual SCSI adapters. Each item in this list has the format:

```
slot_num/adapter_type/remote_lpar_id/remote_lpar_name/remote_slot_num/is_required
```

The attribute names are not present in the list, just their values are present. If an attribute is optional and is not to be included, then no value would be specified for that attribute. For exa

```
4/client//lpar2/3/0
```

specifies a virtual client SCSI adapter with a virtual slot number of 4, a server partition name lpar2, a server slot number of 3, and is not required. The server partition ID was omitted.

A value of none indicates that there are no virtual SCSI adapters.

Valid values for `adapter_type`:

◊ client: client adapter
◊ server: server adapter, valid for Virtual I/O Server partitions only.

Valid values for `is_required`:

◊ 0 - no
◊ 1 - yes

virtual_eth_adapters

Comma separated list of virtual ethernet adapters, with each adapter having the following fo

```
slot_number/is_ieee/port_vlan_id/additional_vlan_ids/is_trunk/is_required
```

All 5 '/' characters must be present, but optional values may be omitted. Optional values are `additional-vlan-IDs`, and `is-trunk`. Valid values for `is_ieee`, `is_trunk`, and `is_required`:

◊ 0 - no
◊ 1 - yes

For example, `4/0/2//0/0` specifies a virtual ethernet adapter with a virtual slot number of 4 not IEEE 802.1Q enabled, has a port virtual LAN ID of 2, no additional virtual LAN IDs, it is n trunk adapter, and is not required.

A value of none indicates that there are no virtual Ethernet adapters.

boot_mode

Partition power on mode. Valid values:
◊ norm - normal
◊ dd - diagnostic with default boot list
◊ ds - diagnostic with stored boot list
◊ of - Open Firmware OK prompt
◊ sms - System Management Services

conn_monitoring

Valid values:
◊ 0 - connection monitoring is disabled
◊ 1 - connection monitoring is enabled

auto_start

Valid values:
◊ 0 - do not automatically start with system power on
◊ 1 - automatically start with system power on

power_ctrl_lpar_ids

A list of partitions which have power control over this partition Valid values are:
◊ none - No partitions

power_ctrl_lpar_names

A list of partitions which have power control over this partition Valid values are:
◊ none - No partitions

Attribute names for the managed system:

name

Name for the managed system

type_model
: Type and model for the managed system

serial_num
: Serial number for the managed system

state
: Valid values:
  - ◊ Operating - the managed system is running

sys_time
: The UTC time of system firmware in the format *month/day/year hour:minute:second*.

power_off_policy
: Valid values:
  - ◊ 0- power off the managed system after all partitions are powered off

cod_mem_capable
: Valid values:
  - ◊ 0- not capable of memory Power On Demand
  - ◊ 1 - capable of memory Power On Demand

cod_proc_capable
: Valid values:
  - ◊ 0- not capable of processor Power On Demand
  - ◊ 1 - capable of processor Power On Demand

micro_lpar_capable
: Valid values: `0`: not capable of creating shared processor partitions
  `1`: capable of creating shared processor partitions

dlpar_mem_capable
: Valid values:
  - ◊ `0`: Changes do not take effect until the next reboot of the partition or platform.
  - ◊ `1`: Changes take effect immediately.

max_lpars
: Maximum number of partitions supported by firmware

max_power_ctrl_lpars
: Maximum number of power controlling partitions per controlled partition

service_lpar_id
: ID of the partition with platform service authority

service_lpar_name
: Name of the partition with platform service authority

mfg_default_config
: Valid values: `0`: the system is not in the manufacturing default partition configuration
  `1`: the system is in the manufacturing default partition configuration

curr_configured_max_lpars
: The current maximum number of partitions supported by the management partition

pend_configured_max_lpars
: The maximum number of partitions supported by the management partition after the next restart

config_version
: The version of the current partition configuration data in platform firmware

pend_lpar_config_state
: Valid values:
  - ◊ `enabled`: the partition configuration data will be enabled during the next restart
  - ◊ `disabled`: the partition configuration data will be disabled during the next restart
  - ◊ `cleared`: the partition configuration data will be cleared to manufacturing defaults during the next restart

lpar_comm_ipaddr
: The IP address through which client partitions will communicate with the management partition. This is used primarily for Dynamic Logical Partitioning (DLPAR). It is defaulted to the first IP address configured on your system, but can be manually set if desired.

lpar_comm_default
: Indicates if the **lpar_comm_ipaddr** is using the default IP address, or if the user has manually set this using **chsyscfg**. Valid values:
  - ◊ 0: User has manually set the IP address
  - ◊ 1: Default IP address is used. This is the first IP address configured on your system as reported by lstcpip -interfaces

--header
: Display a header record, which is a delimiter separated list of attribute names for the attribute values that will be displayed. This header record will be the first record displayed. This option is only valid when used with the **-F** option.

Exit Status

This command has a return code of zero on success.

Security

This command is accessible by all users.

Examples

1. To list the attributes for the managed system, type:

   ```
   lssyscfg -r sys
   ```
2. To list only the user-defined name, machine type and model, and serial number for the managed system, type:

   ```
   lssyscfg -r sys -F name,type_model,serial_num
   ```
3. To list all partitions and only display attribute values for each partition following a header of attribute names, type:

   ```
   lssyscfg -r lpar -F --header
   ```
4. To list the partitions named lpar1, lpar2, and lpar3, type:

   ```
   lssyscfg -r lpar --filter \"lpar_names=lpar1,lpar2,lpar3\"
   ```
5. To list the partition profile for partition lpar2, type:

   ```
   lssyscfg -r prof --filter lpar_names=lpar2
   ```

Related Information

The **chsyscfg** command, the **mksyscfg** command, and the **rmsyscfg** command.

IVM lssysconn Command

Purpose

List connection information for systems. This command is valid only in an Integrated Virtualization Manager environment.

Syntax

To list the service processor network connection information:

lssysconn -r all **-F** "*AttributeNames*" **--header -m** *ManagedSystem*

Description

The **lssysconn** command lists the service processor network connection information.

Flags

-r *ResourceType*

The type of resources to list:

`all` - Lists all connections

- Attributes: resource_type, type_model_serial_num, sp, side, ipaddr, alt_ipaddr, state, eth_loc_code, alt_eth_loc_code
- Filters: None

-m *ManagedSystem*

The name of the managed system. This attribute is optional because there is only one system to manage. The name may either be the user-defined name for the managed system, or be in the form tttt-mmm*ssssssss, where tttt is the machine type, mmm is the model, and ssssssss is the serial number of the managed system.

-F *AttributeNames*

A delimiter separated list of attribute names for the desired attribute values to be displayed for each resource. If no attribute names are specified, then values for all of the attributes for the resource will be displayed.

When this option is specified, only attribute values will be displayed. No attribute names will be displayed. The attribute values displayed will be separated by the delimiter which was specified with this option.

This option is useful when only attribute values are desired to be displayed, or when the values of only selected attributes are desired to be displayed.

The following attributes are available:

resource_type
: Indicates the resource type. This attribute always reads `0`.

type_model_serial_num
: The type-model and serial number of the system in the form ttt-mmm*ssssssss, where tttt is the machine type, mmm is the model, and ssssssss is the serial number of the managed system.

sp
: The type of service processor. This attribute always reads `unavailable`.

side
: The current side of the service processor. This attribute always reads `unavailable`.

ipaddr
: The IP Address of the first Ethernet device on the service processor.

alt_ipaddr
: The IP Address of the second Ethernet device on the service processor.

state
: The connection state to the service processor. This attribute always reads `No Connection`.

eth_loc_code
: The physical location code of the first Ethernet device on the service processor.

alt_eth_loc_code
: The physical location code of the second Ethernet device on the service processor.

--header
: Display a header record, which is a delimiter separated list of attribute names for the attribute values that will be displayed. This header record will be the first record displayed. This option is only valid when used with the -F option.

Exit Status

This command has a return code of zero on success.

Security

This command is accessible by all users.

Examples

1. To list all system connections, type:

```
lssysconn -r all
```

Related Information

The **lssyscfg** command.

lstcpip Command

Purpose

Displays the Virtual I/O Server TCP/IP settings and parameters.

Syntax

lstcpip -num -routtable -routinfo -state -arp

lstcpip -stored

lstcpip -adapters

lstcpip -sockets -family {inet | inet6 | unix}

lstcpip -namesrv | **-interfaces -fmt** *delimiter*

lstcpip -state -field *FieldName* ... | **-routetable -field** *FieldName* ... **-fmt** *delimter*

lstcpip -hostname

Description

The **lstcpip** command displays the current and stored TCP/IP setting such as IP address, routing table, sockets, name server settings, and so forth.

Flags

-adapters Lists Ethernet adapters on the system.

-arp Displays the current ARP table entries.

-famliy Specifies the INET, INET6, or UNIX socket family.

-hostname Shows the system hostname

-namesrv Lists DNS name servers in search order and domain name.

-num Shows numeric output, rather than trying to resolve host names.

-routtable Shows the routing tables.

-routinfo Shows the routing tables, including the user-configured and current costs of each route.

-sockets Displays information about currently open sockets

-state Shows the current state of all configured interfaces

-stored Displays stored TCP/IP configuration, which will be applied at boot time. It will list interfaces' IP addresses, any defined static routes, hostname, DNS info.

Examples

1. To list the Virtual I/O Server TCP/IP configuration, type:

   ```
   lstcpip -stored
   ```
2. To list current routing table, type:

   ```
   lstcpip -routtable
   ```
3. To list open inet sockets, type:

   ```
   lstcpip -sockets -family inet
   ```

lsuser Command

Purpose

Displays user account attributes.

Syntax

lsuser ALL | *Name*, *Name* ...

Description

The **lsuser** command displays the user account attributes. You can use this command to list all attributes of all the system users or all the attributes of specific users. If more than one user is specified, each user must be separated by a comma. If no users are specified the attributes of all users are displayed.

The **lsuser** command lists each user's attributes on one line. It displays attribute information as Attribute=Value definitions, each separated by a blank space.

Exit Status

See Virtual I/O Server command exit status.

Security

This command can be run by any users. However, user attributes are only displayed for the **padmin** user.

Examples

1. To list all users on the system, type:

   ```
   lsuser
   ```

   The system displays output similar to the following for the padmin user:

   ```
   padmin roles=PAdmin account_locked=false expires=0 histexpire=0 histsize=0 loginretries=0 maxage=0
   sally roles=DEUser account_locked=false expires=0 histexpire=0 histsize=0 loginretries=0 maxage=0
   henry roles=DEUser account_locked=false expires=0 histexpire=0 histsize=0 loginretries=0 maxage=0
   admin1 roles=Admin account_locked=false expires=0 histexpire=0 histsize=0 loginretries=0 maxage=0
   deuser1 roles=DEUser account_locked=false expires=0 histexpire=0 histsize=0 loginretries=0 maxage=
   sruser1 roles=SRUser,RunDiagnostics account_locked=false expires=0 histexpire=0 histsize=0 loginre
   view1 roles=ViewOnly account_locked=false expires=0 histexpire=0 histsize=0 loginretries=0 maxage=
   ```

   The system displays the following information for other users:

   ```
   padmin roles=PAdmin
   sally roles=DEUser
   henry roles=DEUser
   admin1 roles=Admin
   deuser1 roles=DEUser
   sruser1 roles=SRUser
   view1 roles=ViewOnly
   ```
2. To display the attributes of user delft, type:

   ```
   lsuser delft
   ```
3. To display the attributes of user delft and user gouda, type:

   ```
   lsuser delft, gouda
   ```

Related Information

The **chuser** command, the **mkuser** command, the **rmuser** command, and the **passwd** command.

lsvg Command

Purpose

Displays information about volume groups.

Syntax

lsvg -map | **-lv** | **-pv -field** *FieldName* **-fmt** *Delimiter VolumeGroup*...

Description

The **lsvg** command displays information about volume groups. If you use the *VolumeGroup* parameter, only the information for that volume group is displayed. If you do not use the *VolumeGroup* parameter, a list of the names of all defined volume groups is displayed.

When information from the Device Configuration database is unavailable, some of the fields will contain a question mark (?) in place of the missing data. The **lsvg** command attempts to obtain as much information as possible from the description area when the command is given a logical volume identifier.

Full scripting support is provided to the **lsvg** command by using the **-field** *FieldNames* and **-fmt** *Delimiter* flags. The **-field** flag will allow the user to select which output fields to display and in what order, while the **-fmt** flag provides scriptable output. The output fields will be displayed in the order they appear on the command line.

If you do not specify any flags, the following information will be displayed:

Volume group	Name of the volume group. Volume group names must be unique and can range from 1 to 15 characters.
Volume group state	State of the volume group. If the volume group is active, the state is either `active/complete` (indicating all physical volumes are active) or `active/partial` (indicating some physical volumes are not active). If the volume group is not active, the state is `inactive`.
Permission	Access permission: `read-only` or `read-write`.
Max LVs	Maximum number of logical volumes allowed in the volume group.
LVs	Number of logical volumes currently in the volume group.
Open LVs	Number of logical volumes within the volume group that are currently open.
Total PVs	Total number of physical volumes within the volume group.
Active PVs	Number of physical volumes that are currently active.
VG identifier	The volume group identifier.
PP size	Size of each physical partition.
Total PPs	Total number of physical partitions within the volume group.
Free PPs	Number of physical partitions not allocated.
Alloc PPs	Number of physical partitions currently allocated to logical volumes.
Quorum	Number of physical volumes needed for a majority.
VGDS	Number of volume group descriptor areas within the volume group.
Auto-on	Automatic activation at IPL (yes or no).
Concurrent	States whether the volume group is Concurrent Capable or Non-Concurrent Capable.
Auto-Concurrent	States whether you should auto activate the Concurrent Capable volume group in concurrent or non-concurrent mode. For volume groups that are Non-Concurrent Capable, this value defaults to Disabled.
VG Mode	The mode of the volume group: Concurrent or Non-Concurrent.
Node ID	Node id of this node if volume group is in concurrent node.
Active Nodes	Node ids of other concurrent nodes that have this volume group active.
Max PPs Per PV	Maximum number of physical partitions per physical volume allowed for this volume group.

Max PVs Maximum number of physical volumes allowed in this volume group.

LTG size Logical track group size, in number of kilobytes, of the volume group.

BB POLICY Bad block relocation policy of the volume group.

SNAPSHOT VG Snapshot volume group name if the snapshot volume group is active else snapshot volume group identifier.

PRIMARY VG Original volume group name of a snapshot volume group if the original volume group is active else original volume group identifier.

Flags

-field Specifies the list of fields to display. The following fields are supported if no flags are specified:

vgname
: Name of the volume group. Volume group names must be unique systemwide and can range from 1 to 15 characters.

vgstate
: State of the volume group. If the volume group is activated with the **activatevg** command, the state is either `active/complete` (indicating all physical volumes are active) or `active/partial` (indicating all physical volumes are not active). If the volume group is de-activated with the **deactivatevg** command, the state is `inactive`.

access
: Access permission: read-only or read-write.

maxlvs
: Maximum number of logical volumes allowed in the volume group.

numlvs
: Number of logical volumes currently in the volume group.

openlvs
: Number of logical volumes within the volume group that are currently open.

totalpvs
: Total number of physical volumes within the volume group.

stalepvs
: Number of PVs which are not current. The data is stale.

stalepps
: Number of PPs which are not current. The data is stale.

activepvs
: Number of physical volumes that are currently active.

vgid
: The volume group identifier.

ppsize
: Size of each physical partition.

totalpps
: Total number of physical partitions within the volume group.

freepps
: Number of physical partitions not allocated.

usedpps
: Number of physical partitions currently allocated to logical volumes.

quorum
: Number of physical volumes needed for a majority.

vgds
: Number of volume group descriptor areas within the volume group.

auton
: Automatic activation at IPL (yes or no).

ppppervg
: Maximum number of physical partitions allowed in this volume group.

ppperpv
: Maximum number of physical partitions per physical volume allowed for this volume group.

maxpvs
: Maximum number of physical volumes allowed in this volume group. This information is displayed only for 32 and 128 PV volume groups.

ltgsize
: Logical track group size of the volume group. The maximum amount of data that can be transferred in one I/O request to the disks of the volume group. The LTG size will be displayed in kilobytes unless the LTG size is greater than 1 MB, in which case megabytes will be used. it is capable of dynamically determining the LTG size based-on the disk topology and it is listed as Dynamic. If that capability is disabled by the user with the option, then it will be listed as

Static.

bbpolicy
Bad block relocation policy of the volume group.

hotspare

autosync

The following fields are supported if the **-lv** flag is specified:

lvname
A logical volume within the volume group.

type
Logical volume type.

lps
Number of logical partitions in the logical volume.

pvs
Number of physical partitions used by the logical volume.

lvstate
State of the logical volume. `Opened/stale` indicates the logical volume is open but contains partitions that are not current. `Opened/syncd` indicates the logical volume is open and synchronized. `Closed` indicates the logical volume has not been opened.

mount
File system mount point for the logical volume, if applicable.

The following fields are supported if the **-pv** flag is specified:

pvname
A physical volume within the volume group.

pvstate
State of the physical volume.

totalpps
Number of physical partitions on the physical volume.

freepps
Number of free physical partitions on the physical volume.

dist
The number of physical partitions allocated within each section of the physical volume: outer edge, outer middle, center, inner middle, and inner edge of the physical volume.

-fmt Specifies a delimiter character to separate output fields.

-pv Lists the following information for each physical volume within the group specified by the *VolumeGroup* parameter:

Physical volume
A physical volume within the group.

PVstate
State of the physical volume.

Total PPs
Total number of physical partitions on the physical volume.

Free PPs
Number of free physical partitions on the physical volume.

Distribution
The number of physical partitions allocated within each section of the physical volume: outer edge, outer middle, center, inner middle, and inner edge of the physical volume.

-lv Lists the following information for each logical volume within the group specified by the VolumeGroup parameter:

LV
A logical volume within the volume group.

Type
Logical volume type.

LPs
Number of logical partitions in the logical volume.

PPs
Number of physical partitions used by the logical volume.

PVs
Number of physical volumes used by the logical volume.

Logical volume state

State of the logical volume. `Opened/stale` indicates the logical volume is open but contains partitions that are not current. `Opened/syncd` indicates the logical volume is open and synchronized. `Closed` indicates the logical volume has not been opened.

Mount Point
File system mount point for the logical volume, if applicable.

-map

Lists the following fields for each logical volume on the physical volume: *PVname*:*PPnum LVname*: *LPnum* :*Copynum PPstate*

PVname
Name of the physical volume as specified by the system.

PPnum
Physical partition number. Physical partition numbers can range from 1 to 1016.

LVname
Name of the logical volume to which the physical partitions are allocated. Logical volume names must be system-wide unique names, and can range from 1 to 64 characters.

LPnum
Logical partition number. Logical partition numbers can range from 1 to 64,000.

Copynum
Mirror number.

PPstate
Only the physical partitions on the physical volume that are not current are shown as stale.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To display the names of all volume groups within the system, type:

   ```
   lsvg
   ```
2. To display information about volume group **vg02**, type:

   ```
   lsvg vg02
   ```

   The characteristics and status of both the logical and physical partitions of volume group **vg02** are displayed.
3. To display the names, characteristics, and status of all the logical volumes in volume group **vg02**, type:

   ```
   lsvg -lv vg02
   ```

Related Information

The **mkvg** command, the **chvg** command, the **extendvg** command, the **reducevg** command, the **mirrorios** command, the **unmirrorios** command, the **activatevg** command, the **deactivatevg** command, the **importvg** command, the **exportvg** command, and the **syncvg** command.

man Command

Purpose

Displays manual entries online.

Syntax

man *Command*

Description

The **man** command provides reference information on commands specified by name.

Exit Status

This command returns the following exit values:

0	Successful completion.
>0	An error occurred.

Examples

1. To display information about the **grep** command, enter:

   ```
   man grep
   ```

Related Information

The **more** command.

migratepv Command

Purpose

Moves allocated physical partitions from one physical volume to one or more other physical volumes.

Syntax

migratepv **-lv** *LogicalVolume SourcePhysicalVolume DestinationPhysicalVolume* ...

Description

The **migratepv** command moves allocated physical partitions and the data they contain from the *SourcePhysicalVolume* to one or more other physical volumes, *DestinationPhysicalVolume*. All physical volumes must be within the same volume group. The specified source physical volume cannot be included in the list of *DestinationPhysicalVolume* parameters.

The allocation of the new physical partitions follows the policies defined for the logical volumes that contain the physical partitions being moved.

If you specify a logical volume that contains the boot image, the **migratepv -lv** command attempts to find enough contiguous partitions on one of the target physical volumes. If the migration is successful, the **migratepv** command will indicate a change in the boot device as well as the new boot physical volume. The attempted migration fails if the **migratepv -lv** command is unable to find enough contiguous space to satisfy the request.

Note: All logical volume manager migration functions work by creating a mirror of the logical volumes involved, then resynchronizing the logical volumes. The original logical volume is then removed. If the **migratepv** command is used to move a logical volume containing the primary dump device, the system will not have an accessible primary dump device during the execution of the command. Therefore, a dump taken during this execution may fail.

Flags

-lv Moves only the physical partitions allocated to the specified logical volume and located on the specified source physical volume.

Exit Status

Return code	Description
8	The physical volume is not assigned to a volume group

Examples

1. To move physical partitions from **hdisk1** to **hdisk6** and **hdisk7**, type:

   ```
   migratepv hdisk1 hdisk6 hdisk7
   ```

   Physical partitions are moved from one physical volume to two others within the same volume group.
2. To move physical partitions in logical volume **lv02** from **hdisk1** to **hdisk6**, type:

   ```
   migratepv -lv lv02 hdisk1 hdisk6
   ```

   Only those physical partitions contained in **lv02** are moved from one physical volume to another.

Related Information

The **lspv** command.

mirrorios Command

Purpose

Mirrors all the logical volumes on rootvg.

Syntax

mirrorios **-f** *PhysicalVolume* ...

Description

The **mirrorios** subcommand takes all the logical volumes on the **rootvg** volume group and mirrors those logical volumes. The target physical drives must already be members of the volume group.

By default, the **mirrorios** subcommand attempts to mirror the logical volumes onto any of the disks in a volume group. If you wish to control which drives are used for mirroring, you must include the list of disks in the input parameters, *PhysicalVolume*. Mirror strictness is enforced. Additionally, the **mirrorios** subcommand mirrors the logical volumes, using the default settings of the logical volume being mirrored.

The user is warned that the Virtual I/O Server will reboot upon the completion of this command and is prompted to continue. If the **-f** option is specified, the command will run without prompting the user.

Note: It is recommended that the rootvg volume group be mirrored on all Virtual I/O Server partitions.

This command can only be executed by the prime administrator.

Flags

-f	Executes the command with out prompting the user to continue.

Exit Status

5	The Virtual I/O Server is already mirrored
6	Boot LV not found
7	The physical volume appears to belong to another volume group

Examples

1. To mirror the Virtual I/O Server root volume group to physical volume **hdisk8**, type:

   ```
   mirrorios hdisk8
   ```

Related Information

The **activatevg** command, the **chvg** command, the **deactivatevg** command, the **exportvg** command, the **importvg** command, the **lsvg** command, the **mkvg** command, the **syncvg** command, and the **unmirrorios** command.

mkbdsp Command

Purpose

Assign storage from a storage pool to be a backing device for a virtual SCSI adapter.

Syntax

Assign an existing logical volume as a backing device:

mkbdsp **-bd** *BackingDevice* **-vadapter** *ServerVirtualSCSIAdapter*

Create a new logical volume as a backing device:

mkbdsp **-sp** *StoragePool Size* **-bd** *BackingDevice* **-vadapter** *ServerVirtualSCSIAdapter*

Description

The **mkbdsp** command assigns a backing device to a virtual SCSI server adapter. If the **-sp** flag is not specified the default storage pool will be used. The user has the option of selecting a specific backing device, using the **-bd** flag, or simply specifying the amount of storage required. Only logical volume backing device can be specified with the **-bd** flag. If the storage size is given, the **mkvdev** command will create a backing device of at least the specified size and assign it as the backing device. Size can be given in megabytes (###M/n), gigabytes (###G/g), or physical partitions (###). The user also has the option of assigning the name for the newly created backing device using the **-bd** flag in combination with the size parameter..

Flags

-bd *BackingDevice*	Specifies the backing device.
-sp *StoragePool*	Specifies the storage pool to be used.
-vadapter	Specifies the virtual SCSI server adapter.

Exit Status

23	Specified storage pool is not a valid storage pool.
26	Specified name is already used. Choose a different name.
34	Specified name is reserved. Choose a different name.

Examples

1. To create a virtual target device that maps a 3 GB backing device from the default storage pool the virtual SCSI server adapter vhost3, type:

   ```
   mkbdsp 3g -vadapter vhost3
   ```

mkdir Command

Purpose

Creates one or more new directories.

Syntax

mkdir **-m** *Mode* **-p** *Directory ...*

Description

The **mkdir** command creates one or more new directories specified by the *Directory* parameter. Each new directory contains the standard entries **.** (dot) and **..** (dot-dot). You can specify the permissions for the new directories with the **-m** *Mode* flag.

The owner-ID and group-ID of the new directories are set to the process's effective user-ID and group-ID, respectively. The setgid bit setting is inherited from the parent directory. To change the setgid bit, you can either specify the **-m** *Mode* flag or issue the **chmod** command after the creation of the directory.

Note: To make a new directory you must have write permission in the parent directory.

Flags

-m *Mode*

Sets the permission bits for the newly-created directories to the value specified by the *Mode* variable. The *Mode* variable takes the same values as the *Mode* parameter for the **chmod** command, either in symbolic or numeric form.

When you specify the **-m** flag using symbolic format, the op characters + (plus) and - (minus) are interpreted relative to the assumed permission setting a=rwx. The + adds permissions to the default mode, and the - deletes permissions from the default mode. Refer to the **chmod**command for a complete description of permission bits and formats.

-p

Creates missing intermediate path name directories. If the **-p** flag is not specified, the parent directory of each-newly created directory must already exist.

Intermediate directories are created through the automatic invocation of the following **mkdir** commands:

```
mkdir -p -m $(umask -S),u+wx $(dirname Directory) &&
mkdir -m Mode Directory
```

where the `-m Mode` represents any option supplied with your original invocation of the **mkdir** command.

The **mkdir** command ignores any *Directory* parameter that names an existing directory. No error is issued.

Exit Status

This command returns the following exit values:

0 All the specified directories were created successfully, or the **-p** option was specified and all the specified directories now exist.

>0 An error occurred.

Examples

1. To create a new directory called `Test` in the current working directory, enter:

   ```
   mkdir Test
   ```

   The `Test` directory is created with default permissions.
2. To create a new directory called `Test` with rwxr-xr-x permissions in the previously created `/home/demo/sub1` directory, enter:

   ```
   mkdir -m 755 /home/demo/sub1/Test
   ```
3. To create a new directory called `Test` with default permissions in the `/home/demo/sub2` directory, enter:

   ```
   mkdir -p /home/demo/sub2/Test
   ```

   The **-p** flag creates the `/home`, `/home/demo`, and `/home/demo/sub2` directories if they do not already exist.

Files

/usr/bin/mkdir	Contains the **mkdir** command.

Related Information

The **chmod** command, **rm** command.

IVM mkgencfg Command

Purpose

Performs the initial logical partition configuration for the managed system. This command is valid only in an Integrated Virtualization Manager environment.

Syntax

mkgencfg -o *init* `-i` "*ConfigurationData*" **-m** *ManagedSystem*

Description

The **mkgencfg** command performs the initial logical partition configuration for the managed system. As part of the initial configuration, virtual Ethernet adapters are created in the management partition. The virtual Ethernet MAC address prefix can optionally be configured through this command.

Flags

-o *Operation*
The type of operations: `init` - Perform initial logical partition configuration actions for the managed system

-i *ConfigurationData*
The configuration data consists of attribute name/value pairs, which are in comma separated value (CSV) format. The format of a configuration record is as follows:

```
"attribute-name=value,attribute-name=value,..."
```

Note that certain attributes accept a comma separated list of values, as follows:

```
""attribute-name=value,value,...",..."
```

When a list of values is specified, the attribute name/value pair must be enclosed in double quotes. Depending on the shell being used, nested double quote characters may need to be preceded by an escape character.

The valid attribute names for configuration data are:

- `mac_prefix` - This must be specified as a 3 byte hexadecimal value. The first 2.5 bytes of the MAC address to be assiged to all virtual Ethernet adapters for this managed system. Additionally, the value must not be a multicast address (010000 bit must be off), and must be a private address (020000 bit must be on). An example MAC address prefix is 0642A0.
- `pend_configured_max_lpars` - The maximum number of partitions supported by the management partition after the next restart

-m *ManagedSystem*
The name of the managed system. This attribute is optional because there is only one system to manage. The name may either be the user-defined name for the managed system, or be in the form tttt-mmm*ssssssss, where tttt is the machine type, mmm is the model, and ssssssss is the serial number of the managed system.

Exit Status

This command has a return code of zero on success.

Security

This command is not accessible by users with the ViewOnly role.

Examples

1. To initialize the logical partition configuration for the managed system using defaults, type:

   ```
   mkgencfg -o init
   ```
2. To initialize the logical partition configuration for the managed system with support for 17 partitions and a MAC prefix of 0x06ABC0, type:

   ```
   mkgencfg -o init -i "pend_lpm_max_lpars=17,mac_prefix=06ABC0"
   ```

mklv Command

Purpose

Creates a logical volume.

Syntax

mklv **-mirror** **-lv** *NewLogicalVolume* | **-prefix** *Prefix VolumeGroup Size PhysicalVolume* ...

Description

The **mklv** command creates a new logical volume within the *VolumeGroup*. If you specify one or more physical volumes with the *PhysicalVolume* parameter, only those physical volumes are available for allocating physical partitions; otherwise, all the physical volumes within the volume group are available.

The allocation policy is to use a minimum number of physical volumes.

The *Size* parameter specifies the minimum size the logical volume should be. When specifying Size the following conventions must be used:

Size	Logical volume size
###M/m	### MB
###G/g	### GB

Flags

-lv Specifies the logical volume name to use instead of using a system-generated name. Logical volume names must be unique system wide name, and can range from 1 to 15 characters.

-mirror Activates mirroring for this logical volume.

-prefix Specifies the Prefix to use instead of the prefix in a system-generated name for the new logical volume. The prefix must be less than or equal to 13 characters. The name cannot begin with a prefix already defined in the **PdDv** class in the Device Configuration Database for other devices, nor be a name already used by another device.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To make a logical volume in volume group **vg02** with a minimum size of 1 Mb , type:

   ```
   mklv vg02 1MB
   ```
2. To make a logical volume in **vg03** with 1GB chosen from physical volumes **hdisk5**, **hdisk6**, and **hdisk9**, type:

   ```
   mklv vg03 1GB hdisk5 hdisk6 hdisk9
   ```
3. To request a logical volume with a minimum size of 10MB, type:

   ```
   mklv VGNAME 10M
   ```

   where *VGNAME* is the name of your logical volume.

Related Information

The **lslv** command, the **extendlv** command, and the **rmlv** command.

mklvcopy Command

Purpose

Creates a mirror of a logical volume.

Syntax

mklvcopy *LogicalVolume PhysicalVolume* ...

Description

The **mklvcopy** command creates a mirror (an additional copy) of a *LogicalVolume*. The *LogicalVolume* parameter can be a logical volume name or logical volume ID. You can request that the new copy of the logical volume be allocated on specific physical volumes (within the volume group) with the *PhysicalVolume* parameter; otherwise, all the physical volumes within the volume group are available for allocation. The new copy of the logical volume will be placed on a separate physical volume.

Note: Only one additional copy of a logical volume can be created.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To create a copy of the logical volume **lv01**, so that a total of two copies exist, type:

   ```
   mklvcopy lv01
   ```

Related Information

The **extendlv** command, the **lslv** command, the **mklv** command, the **rmlv** command, and the **rmlvcopy** command.

mkpath Command

Purpose

Adds to the system another path to an MPIO capable device.

Syntax

mkpath { **-dev** *Name* **-pdev** *Parent* **-conn** *Connection* } **-def**

Description

The **mkpath** command defines, and possibly configures, one or more paths to the target device (**-dev** *Name*). The paths are identified by a combination of the **-dev** *Name*, **-pdev** *Parent*, and **-conn** *Connection* flags. Both the target device and parent must be previously defined in the system to define a path. They both must be AVAILABLE to configure a path.

If the **-def** *flag* is specified, the **mkpath** command only defines the new path definition to the system. If the **-def** flag is not specified, the **mkpath** command attempts to define the path, if it does not already exist, before it attempts to configure the path. Configuring a path requires the path to already be defined and both the device and the parent device to already be configured.

The **mkpath** command displays a status message upon completion. It is possible for some paths to configure and others to fail.

Note that not all devices will be able to have paths manually defined by using the **mkpath** command. These limitations are due to the way that path information is stored for these devices. Fiber channel devices fall into this category.

The **mkpath** command provides status messages about the results of operation. Messages in one of the following formats will be generated:

path available | defined
: This message is displayed when **mkpath** is run on a single path. If the path is successfully configured the message `path available` is displayed. If the path is not successfully configured and there is no explicit error code returned by the method, the message `path defined` is displayed.

paths available
: This message is displayed if multiple paths were identified and all paths were successfully configured.

some paths available
: This message is displayed if multiple paths were identified, but only some of them were successfully configured.

no paths processed
: This message is generated if no paths were found matching the selection criteria.

Flags

-conn *Connection*	Indicates the connection information associated with the path to be added. This flag is required if the **-def** flag is specified.
-def	Defines a new path to a device by adding a path definition to the system. The new path will not automatically be configured when the **-def** flag is specified. Note that only one path may be defined at a time. The **-conn** and the **-pdev** flags are required when the **-def** flag is used.
-dev *Name*	Specifies the logical device name of the target device to which the path(s) are being added. The path(s) to be added are qualified by the **-pdev** and **-conn** flags.
-pdev *Parent*	Indicates the logical device name of the parent device associated with the path(s) to be added. This flag is required if the **-def** flag is specified.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To define and configure an already defined path between **scsi0** and the **hdisk1** device at **SCSI ID 5** and **LUN 0** (connection 5,0), enter:

   ```
   mkpath -dev hdisk1 -pdev scsi0 -conn 5,0
   ```

   The system displays a message similar to the following:

   ```
   path available
   ```
2. To configure an already defined path from **fscsi0** to fiber channel disk **hdisk1**, type:

   ```
   mkpath -dev hdisk1 -pdev fscsi0
   ```

   The system displays a message similar to the following:

   ```
   path available
   ```
3. To only add to the Customized Paths object class a path definition between **scsi0** and the **hdisk1** disk device at **SCSI ID 5** and **LUN 0**, enter:

   ```
   mkpath -def -dev hdisk1 -pdev scsi0 -conn 5,0
   ```

   The system displays a message similar to the following:

   ```
   path defined
   ```

Related Information

The **lspath** command and the **rmpath** command.

mktcpip Command

Purpose

Sets the required values for starting TCP/IP on a host.

Syntax

mktcpip -hostname *HostName* **-inetaddr** *Address* **-interface** *Interface* **-start -netmask** *SubnetMask* **-cabletype** *CableType* **-gateway** *Gateway* **-nsrvaddr** *NameServerAddress* **-nsrvdomain** *Domain*

Description

The **mktcpip** command sets the required minimal values required for using TCP/IP on a host machine. The basic functions of the **mktcpip** command include the following:

- Setting the host name
- Setting the IP address of the interface
- Setting the domain name and IP address of the nameserver, if applicable
- Setting the subnetwork mask, if applicable
- Starting the specified TCP/IP daemons

Flags

-cabletype *CableType*
Specifies cable size for Standard Ethernet or IEEE 802.3 Ethernet networks. Valid values for the *CableType* variable are `dix` for thick cable, `bnc` for thin cable, or `N/A` for not applicable. The **-cabletype** *CableType* flag should be used only for Standard Ethernet (en) and IEEE 802.3 Ethernet (et) interfaces. The default is N/A.

-gateway *Gateway*
Sets the gateway address for a static route. Specify the address in dotted decimal notation.

-hostname *Hostname*
Sets the name of the host. If using a domain naming system, the domain and any subdomains must be specified. The following is the standard format for setting the host name:

```
hostname
```

The following is the standard format for setting the host name in a domain naming system:

```
hostname.subdomain.subdomain.rootdomain
```

-inetaddr *Address*
Sets the Internet address of the host. Specify the address in dotted decimal notation. Each network interface on the host should have a unique Internet address. The following is the standard format for setting the Internet address:

```
127.10.31.2
```

-interface *Interface*
Specifies a particular network interface, for example:

```
tr0
```

-netmask *SubnetMask*
Specifies the mask the gateway should use in determining the appropriate subnetwork for routing. The subnet mask is a set of 4 bytes, as in the Internet address. The subnet mask consists of high bits (1s) corresponding to the bit positions of the network and subnetwork address, and low bits (0s) corresponding to the bit positions of the host address.

-nsrvaddr *NameserverAddress*
Specifies the Internet address of the name server the host uses for name resolution, if applicable. The address should be entered in dotted decimal notation, as follows:

```
127.1.0.1
```

-nsrvdomain *Domain*
Specifies the domain name of the name server the host should use for name resolution, if any. The domain name should be in the following format:

`subdomain.subdomain.rootdomain`

-start Starts the TCP/IP daemons.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To set the required values for starting TCP/IP type:

```
mktcpip -hostname fred.austin.century.com -inetaddr 192.9.200.4 -interface en0 \
-nsrvaddr 192.9.200.1 -nsrvdomain austin.century.com -start
```

Related Information

The **hostname** command, the **startnetsvc** command, the **stopnetsvc** command, the **cfglnagg** command, the **netstat** command, the **entstat** command, the **cfgnamesrv** command, the **hostmap** command, the **traceroute** command, the **ping** command, the **optimizenet** command.

mkuser Command

Purpose

Creates a new user account.

Syntax

mkuser **-de** | **-sr** **-attr** *Attributes=Value Attribute=Value... Name*

Description

The **mkuser** command creates a new user account. Upon completion of creating a the new account you will be prompted for set the new account's password. Users created with the `-attr pgrp=view` are designated as readonly. These users will not have the authority to change the system configuration and will not have the write permission to their home directories.

For a list of supported attributes, see chuser Command.

Flags

-attr *Attribute=Value*
Identifies the attribute to set as well as the new value for the attribute. The Attribute=Value parameter can use one attribute value pair or multiple attribute value pairs for one **-attr** flag.

-de
Creates a Development Engineer or DE user account. This type of account allows developers to log into the Virtual I/O Server and debug problems.

-sr
Creates a service representative or SR user account. This type of account enables a service representatives to run commands required to service the system without being logged in as root. This includes the following:

- Run diagnostics, including service aids (for example, hot plug tasks, certify, format, and so forth.)
- Run all commands that can be run by a group system
- Configure and unconfigure devices that are not busy
- Use the service aid to update system microcode
- Perform the shutdown and reboot operations

The recommended SR login user name is **qserv**.

Exit Status

See Virtual I/O Server command exit status.

Security

This command can only be executed by the prime administrator (padmin) user.

Examples

1. To create the **davis** user account with default values, type:

   ```
   mkuser davis
   ```
2. To create the **davis** user account and set the **maxage** attribute to a value of 52, type:

   ```
   mkuser -attr maxage=52 davis
   ```
3. To create a user with read only authority: type:

   ```
   mkuser -attr pgrp=view View1
   ```

Related Information

The **chuser** command, the **lsuser** command, the **rmuser** command, and the **passwd** command.

mkvdev Command

Purpose

Adds a virtual device to the system.

Syntax

To create a virtual target device:

mkvdev **-f** {**-vdev** *TargetDevice* | **-dplc** *TDPhysicalLocatonCode* } { **-vadapter** *VirtualServerAdapter* | **-aplc** *VSAPhysicalLocationCode*} **-dev** *DeviceName*

To create a Shared Ethernet Adapter:

mkvdev **-sea** *TargetDevice* **-vadapter** *VirtualEthernetAdapter*... **-default** *DefaultVirtualEthernetAdapter* **-defaultid** *SEADefaultPVID* **-attr** *Attribute=Value Attribute=Value*...

To create an Link Aggregation adapter:

mkvdev **-lnagg** *TargetAdapter*... **-attr** *Attribute=Value Attribute=Value*...

To create a VLAN Ethernet adapter:

mkvdev **-vlan** *TargetAdapter* **-tagid** *TagID*

Description

The **mkvdev** command creates a virtual device. The name of the virtual device will be automatically generated and assigned unless the **-dev** *DeviceName* flag is specified, in which case *DeviceName* will become the device name. If the **-lnagg** flag is specified, a Link Aggregation or IEEE 802.3 Link Aggregation (automatic Link Aggregation) device is created. To create an IEEE 802.3 Link Aggregation set the mode attribute to `8023ad`. If the **-sea** flag is specified, a Shared Ethernet Adapter is created. The *TargetDevice* may be a Link Aggregation adapter (note, however, that the *VirtualEthernetAdapter* may not be Link Aggregation adapters). The default virtual Ethernet adapter, *DefaultVirtualEthernetAapter*, must also be included as one of the virtual Ethernet adapters, *VirtualEthernetAdapter*. The **-vlan** flag is used to create a VLAN device and the **-vdev** flag creates a virtual target device which maps the *VirtualServerAdapter* to the *TargetDevice*.

If the backing device that is specified by the **-vdev** or **-dplc** flags is already in use, an error will be returned unless the **-f** flag is also specified.

The **mkvdev** command also configures virtual optical devices, where the **-vdev** or **-dplc** flags specify the physical optical device and the **-vadapter** or **-aplc** flags specify the virtual SCSI adapter. If the specified optical device is already assigned to a virtual SCSI adapter an error is returned unless the **-f** flag is also specified. If the **-f** flag is specified, the optical device will be removed from the virtual SCSI adapter it is currently assigned to before reassigning it to the new virtual SCSI adapter.

Attention: To protect the Configuration Database, the **mkvdev** command is not interruptible. Stopping this command before execution is complete could result in a corrupted database.

Flags

-aplc *VSAPhysicalLocationCode*	Specifies the virtual SCSI adapter using the physical location code
-attr *Attribute=Value*	Specifies the device attribute value pairs to be used instead of the defaults. The *Attribute=Value* variable can be used to specify one attribute value pair or multiple attribute value pairs for one **-attr** flag. If you use an **-attr** flag

with multiple attribute value pairs, the list of pairs must be enclosed in quotation marks with a blank space between the pairs. For example, entering **-attr** *Attribute=Value* lists one attribute value pair per flag, while entering **-attr** '*Attribute1=Value1 Attribute2=Value2*' lists more than one attribute value pair.

-default *DefaultVirtualEthernetAdapter*
Default virtual adapter to use for non-VLAN-tagged packets. This flag maps to the SEA device attribute pvid_adapter.

-defaultid *SEADefaultPVID*
The SEADafaultPVID is the VID used for untagged frames. All untagged packets are assigned the SEADefaultPVID value. When a tagged frame is received by a port, the tag is used. Otherwise if the frame is untagged, the value contained in the PVID is considered as a tag. This flag maps to the SEA device attribute pvid.

-dev *DeviceName*
By using the **-dev** flag, you can specify the name you want the device to be known by. If you do not use the **-dev** flag, a name will be automatically generated and assigned. Not all devices support user-supplied names.

-dplc *TDPhysicalLocatonCode*
Specifies the physical device using the physical location code

-f
Force the physical volume specified to be used as a backing device even if its already associated with a virtual SCSI adapter. If the specified backing device is an optical device, **-f** forces the optical device will be removed from the virtual SCSI adapter it is currently assigned to before reassigning it to the new virtual SCSI adapter.

-lnagg *TargetAdapter*...
Creates a Link Aggregation device.

-sea *TargetDevice*
Creates a Shared Ethernet Adapter which maps *VirtualEthernetAdapter* to the adapter *TargetDevice*. *TargetDevice* can be a physical adapter or a Link Aggregation adapter.

-tagid *TagID*
Specifies the VLAN tag ID.

-vadapter *VirtualEthernetAdapter* or *VirtualServerAdapter*
Specifies the virtual server adapter the new device will be mapped to.

-vdev *TargetDevice*
Creates a virtual device mapped to the physical/logical device *TargetDevice* and the virtual server adapter *VirtualServerAdapter*. The *TargetDevice* can be either a physical volume, logical volume, or optical device. Physical volumes assigned to volume groups cannot be used as target devices.

-vlan *TargetAdapter*
Creates a Virtual Local Area Network device.

Exit Status

13 Specified physical or logical volume is not a valid.
21 Device is already in use. Use the **-f** flag to force assignment.
22 Logical volumes cannot be assigned as backing devices more than once.

Examples

1. To create a virtual target device that maps the logical volume **lv20** as a virtual disk for a client partition hosted by the **vhost0** virtual server adapter, type:

   ```
   mkvdev -vdev lv20 -vadapter vhost0
   ```

   The system displays a message similar to the following:

   ```
   vtscsi0 available
   ```
2. To create a virtual target device that maps the physical volume **hdisk6** as a virtual disk for a client partition served by the **vhost2** virtual server adapter, type:

   ```
   mkvdev -vdev hdisk6 -vadapter vhost2
   ```

   The system displays a message similar to the following:

   ```
   vtscsi1 available
   ```
3. To create a Shared Ethernet Adapter that maps the physical Ethernet adapter **ent4** as a virtual Ethernet adapter for the client partitions served by the virtual Ethernet adapters **ent6**, **ent7**, and **ent9**, using **ent6** as the default adapter and **8** as the default ID, type:

   ```
   mkvdev -sea ent4 -vadapter ent6,ent7,ent9 -default ent6 -defaultid 8
   ```

The system displays a message similar to the following:

```
ent10 available
```

4. To create an automatic Link Aggregation with primary adapters **ent4** and **ent5** and backup adapter **ent6**, type:

```
mkvdev -lnagg ent4,ent5 -attr backup_adapter=ent6 mode=6023ad
```

The system displays a message similar to the following:

```
ent10 available
```

Related Information

The **cfgdev** command, the **chdev** command, the **chpath** command, the **lsdev** command, the **lsmap** command, and the **rmdev** command.

mkvg Command

Purpose

Creates a volume group.

Syntax

mkvg **-f** **-vg** *VolumeGroup PhysicalVolume* ...

Description

The **mkvg** command creates a new volume group using the physical volumes represented by the *PhysicalVolume* parameter. After creating the volume group, the **mkvg** command automatically activates the new volume group using the **activatevg** command.

Notes:

1. The physical volume is checked to verify that it is not already in another volume group. If the system believes the physical volume belongs to a volume group that is active, it exits. But if the system detects a description area from a volume group that is not active, it prompts the user for confirmation in continuing with the command. The previous contents of the physical volume are lost, so the user must be cautious when using the override function.
2. This command will fail to add a disk to the volume group if the disk indicates that it is managed by a third party volume manager.

Flags

-f
Forces the volume group to be created on the specified physical volume unless the physical volume is part of another volume group in the Device Configuration Database or a volume group that is active.

-vg *VolumeGroup*
Specifies the volume group name rather than having the name generated automatically. Volume group names must be unique system wide and can range from 1 to 15 characters. The name cannot begin with a prefix already defined in the PdDv class in the Device Configuration database for other devices. The volume group name created is sent to standard output.

The volume group name can only contain the following characters: "A" through "Z," "a" through "z," "0" through "9," or "_" (the underscore), "-" (the minus sign), or "." (the period). All other characters are considered invalid.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To create a volume group that contains physical disks **hdisk3**, **hdisk5**, and **hdisk6**, type:

   ```
   mkvg hdisk3 hdisk5 hdisk6
   ```

   The volume group is created with an automatically generated name, which is displayed.
2. To create the volume group **newvg** with one physical partition, type:

   ```
   mkvg -vg newvg hdisk1
   ```

Related Information

The **lsvg** command, the **chvg** command, the **extendvg** command, the **reducevg** command, the **mirrorios** command, the **unmirrorios** command, the **activatevg** command, the **deactivatevg** command, the **importvg** command, the **exportvg** command, and the **syncvg** command.

mksp Command

Purpose

Create a storage pool.

Syntax

mksp **-f** *StoragePool PhysicalVolume* ...

Description

The **mksp** command creates a new storage pool, using the physical volumes represented by the *PhysicalVolume* parameter.

If the system detects a description area from a volume group that is not varied on, it prompts the user for confirmation in continuing with the command. The previous contents of the physical volume are lost, so the user must be cautious when using the override function. Specifying the **-f** flag will create the volume group with out prompting the user.

Flags

-f	Forces the storage pool to be created on the specified physical volume unless the physical volume is part of another storage pool or volume group in the Device Configuration Database or a volume group that is active.

Examples

1. To create a new storage pool from physical volumes hdisk3 and hdisk4 and with the name client_data, type:

   ```
   mksp -f client_data hdisk3 hdisk4
   ```

   The new storage pool is created with the name client_data.

IVM mksvcevent Command

Purpose

Creates a new serviceable event. This command is valid only in an Integrated Virtualization Manager environment.

Syntax

mksvcevent -d *Description* **--reporting_mtms** *ReportingMTMS*

Description

The **mksvcevent** command creates a serviceable event with the specified description. This event will show up in the list of serviceable events obtained by the **lssvcevents** command.

Flags

-d *Description*	The description or text of the event.
-reporting_mtms *ReportingMTMS*	The type-model*serial of the reporting system. This should be in the form tttt-mmm*ssssssss, where tttt is the machine type, mmm is the model, and ssssssss is the serial number of the managed system.

Exit Status

This command has a return code of zero on success.

Security

This command is not accessible by users with the ViewOnly role.

Examples

1. To create a serviceable event, type:

```
mksvcevent -d This is a test event entry -reporting_mtms 9111-520*XXXXXXX
```

Related Information

The **lssvcevents** command, and the **chsvcevent** command.

IVM mksyscfg Command

Purpose

Creates a logical partition on the managed system. This command is usable only in an Integrated Virtualization Manager environment.

Syntax

mksyscfg -r *lpar* { **-f** *ConfigurationFile* | **-i** *ConfigurationData* } **-m** *ManagedSystem*

Description

The **mksyscfg** command creates a logical partition on the managed system.

Flags

-r *ResourceType*	The type of resources to create: **lpar** - Logical partition resources
-m *ManagedSystem*	The name of the managed system. This attribute is optional because there is only one system to manage. The name may either be the user-defined name for the managed system, or be in the form tttt-mmm*ssssssss, where tttt is the machine type, mmm is the model, and ssssssss is the serial number of the managed system.

-f *ConfigurationFile*

The name of the file containing the configuration data needed to change the resources. The configuration data consists of attribute name/value pairs, which are in comma separated value (CSV) format. These attribute name/value pairs form a configuration record. A line feed marks the end of a configuration record. The file must contain one configuration record for each resource to be changed, and each configuration record must be for the same resource type. If the resource type is the managed system or the managed frame, then the file must contain only one configuration record.

The format of a configuration record is as follows:

```
attribute-name=value,attribute-name=value,...<LF>
```

Note that certain attributes accept a comma separated list of values, as follows:

```
"attribute-name=value,value,...",...<LF>
```

When a list of values is specified, the attribute name/value pair must be enclosed in double quotes. Depending on the shell being used, nested double quote characters may need to be preceded by an escape character.

Required Attributes for Partition

name
: Name of the partition to create.

lpar_env
: `aixlinux` is the only valid attribute. This attribute creates an RPA partition type, which supports AIX or Linux.

min_mem
: The minimum megabytes of memory for this partition.

desired_mem
: The assigned megabytes of memory for this partition

max_mem
: The maximum megabytes of memory for this partition

Optional Attributes for partitions

profile_name

Name of the profile to create. This attribute is not required, but if specified, must be the same as the **name** attribute. **Note:** Both **name** and **profile_name** are required, and must be the same. When using this command on the Hardware Management Console, you must specify the profile name and partition name because it supports multiple profiles per partition.

lpar_id
: Unique integer ID for the partition. If this attribute is not specified, the lowest available partition will be assigned.

min_procs
: The minimum number of processors for this partition. In shared processing mode, this refers to virtual processors.

desired_procs
: The assigned number of processors for this partition. In shared processing mode, this refers to virtual processors.

max_procs
: The maximum number of processors for this partition. In shared processing mode, this refers to virtual processors.

min_proc_units
: The minimum number of processing units for this partition

desired_proc_units
: The assigned number of processing units for this partition

max_proc_units
: The maximum number of processing units for this partition

proc_mode
: Valid values are: `ded` - dedicated processor mode
`shared` - shared processor mode

sharing_mode
: Value values are: `keep_idle_procs` - valid with dedicated processor mode
`share_idle_procs` - valid with dedicated processor mode
`cap` - capped mode. valid with shared processor mode
`uncap` - uncapped mode. valid with shared processor mode

uncap_weight
: A weighted average of processing priority when in uncapped sharing mode. The smaller the value, the lower the weight. Valid values are: 0 - 255

boot_mode
: Partition power on mode. Valid values are: `norm` - normal
`dd` - diagnostic with default boot list
`ds` - diagnostic with stored boot list
`of` - Open Firmware OK prompt
`sms` - System Management Services

auto_start
: Valid values are: `0` - do not automatically start with system power on
`1` - automatically start with system power on

max_virtual_slots
: **Note:** The system determines this value.Maximum number of virtual I/O adapter slots.

virtual_eth_adapters
: Comma separated list of virtual ethernet adapters, with each adapter having the following format:

`slot_number/is_ieee/port_vlan_id/additional_vlan_ids/is_trunk/is_required`

All 5 '/' characters must be present, but optional values may be omitted. Optional values are `additional_vlan_ids`, and `is_trunk`.

Valid values for `is_ieee`, `is_trunk`, and `is_required`:

`0` - no
`1` - yes
For example, 4/0/2//0/0 specifies a virtual ethernet adapter with a virtual slot number of 4, is not IEEE 802.1Q enabled, has a port virtual LAN ID of 2, no additional virtual LAN IDs, it is not a trunk adapter, and is not required.

-i *ConfigurationData*
: This option allows you to enter configuration data on the command line, instead of using a file. Data entered on the command line must follow the same format as data in

a file, and must be enclosed in double quotes. When this option is used, only a single resource can be changed. The **-i** and the **-f** options are mutually exclusive.

Exit Status

This command has a return code of zero on success.

Security

This command is not accessible by users with the ViewOnly role.

Examples

1. To create a partition named lp3 with 128 MB, type:

   ```
   mksyscfg -r lpar -i "name=lp3,lpar_env=aixlinux,min_mem=128,desired_mem=128,max_mem=128"
   ```
2. To create a partition with 128 MB and a dedicated processor, type:

   ```
   mksyscfg -r lpar -i "name=lp4,lpar_env=aixlinux,min_mem=128,desired_mem=128,max_mem=128, \
   proc_mode=ded, sharing_mode=share_idle_procs,min_procs=1,desired_procs=1,max_procs=2"
   ```
3. To create a partition with 128 MB and 0.2 shared processing units, type:

   ```
   mksyscfg -r lpar -i "name=lp2,lpar_env=aixlinux,min_mem=128,desired_mem=128,max_mem=128, \
   proc_mode=shared, sharing_mode=uncap,min_procs=1,desired_procs=1,max_procs=2, \
   min_proc_units=0.1,desired_proc_units=0.2, max_proc_units=2"
   ```
4. To create a partition with 128 MB and 0.2 shared processing units. and a virtual Ethernet adapter on VLAN 1, type:

   ```
   mksyscfg -r lpar -i "name=lp2,lpar_env=aixlinux,min_mem=128,desired_mem=128,max_mem=128, \
   proc_mode=shared, sharing_mode=uncap,min_procs=1,desired_procs=1,max_procs=2, \
   min_proc_units=0.1,desired_proc_units=0.2, max_proc_units=2,virtual_eth_adapters=4/0/1//0/0"
   ```

Related Information

The **lssyscfg** command, the **mksyscfg** command, and the **rmsyscfg** command.

IVM mkvt Command

Purpose

Create a virtual terminal connection to a partition.

Syntax

mkvt { **-id** *lparID* }

Description

The **mkvt** command opens a virtual terminal connection to the target partition. After establishing a virtual terminal session, the ~. character sequence can be entered in the terminal window to terminate it, or the **rmvt** command can be used to force the session to be closed. A partition can only have one open virtual terminal session.

This command requires additional HMC configuration if used in an HMC environment.

Flags

-id *lparID*	The ID of the partition for which to open the virtual terminal session.

Exit Status

27	Unexpected error
28	Virtual terminal is already connected
29	Virtual terminal device is not found
30	Permission denied
31	Specified device does not exist

Security

This command is not accessible by users with the ViewOnly role.

Examples

1. Create a virtual terminal connection to the partition with ID 3:

```
mkvt -id 3
```

Related Information

The **rmvt** command.

more Command

Purpose

Displays file contents one screen at a time.

Syntax

more **-c** **-d** **-e** **-H** **-i** **-l** **-N** **-s** **-u** **-v** **-z** **-n** *Number* **-p** *Subcommand* **-t** *Tagstring* **-W** *Option* **-x** *Tabs* *File* ...

Description

The **more** command reads files and displays the text one screen at a time. The command pauses after each screen and prints the word `More` at the bottom of the screen. If you then press a carriage return, the **more** command displays an additional line. If you press the space bar, the **more** command displays another full screen of text.

> **Note:** On some terminal models, the **more** command clears the screen, instead of scrolling.

Instead of naming files to read, you can either redirect or pipe standard output, such as a long directory listing, to the **more** command. The command adds a % (percent sign) to its prompt when reading from a file rather than a pipe. This provides the percentage of the file (in characters, not lines) that the **more** command has read.

The **more** command sets the terminal to NOECHO mode so the output can be continuous. With the exception of the `/` and `!` subcommands, commands that are typed do not normally show up on the terminal. If the standard output is not a terminal, the **more** command will act just like the **cat** command, except that a header will be printed before each file in a series.

Environment Variables

Environment variables affect the way the **more** command works. You can set some environment characteristics in the /**etc**/**environment** file and system profile files, such as the **.ksh**, **.csh**, and **.profile** files.

The **more** command uses the **TERM** variable to determine terminal characteristics. If this variable is NULL or not set, the command uses the default terminal type. The /**usr**/**share**/**lib**/**terminfo** directory contains definitions for terminal characteristics.

By default, the **more** command window size is 2 lines less than what the system terminal is capable of. The command sets the default window size based on the **LINES** variable. Also, you can easily adjust the window size for each run of the command by adding the **-n** flag.

Use the **MORE** variable to customize the **more** command with your preferred configuration each time the system starts. This variable accepts **more** command flags.

Flags

-c Prevents the screen from scrolling, which makes text easier to read as the **more** command writes to the screen. The system ignores the **-c** flag if the terminal cannot clear to the end of a line.

-d Prints a message, appended to the `More` prompt at the bottom of the screen, about which keys continue, quit, and provide help for the **more** command. Displays error messages rather than ringing the terminal bell if an unrecognized command is used. This is helpful for inexperienced users.

-e Exits automatically after displaying the last line of the last file.

-H Disables the search pattern highlighting feature by default.

-i Searches for patterns without considering uppercase and lowercase.

-l Pauses after detecting a page break in the input. If the `-l` flag is not used, the **more** command pauses to accept commands after any line containing a `^L` (CTRL-L) character. Also, if a file begins with a FORMFEED, the screen is cleared before the file is printed.

-N Suppresses line numbering. The default display, with line numbers, can slow the **more** command's performance on very large input files. The line numbering feature displays the line number in the **=** subcommand and passes the line number to the editor (if it is the **vi** editor).

-n *Number* Configures the **more** command to display the specified number of lines in the window. Without the **-n** flag, the **more** command defaults to two lines less than what the terminal is capable of. For example, on a 24-line terminal, the default is 22 lines. The `-n` option overrides any values obtained from the environment.

-p *Subcommand* Starts the **more** command and specified subcommand for each *File* operand. For example, **more -p 50j text1 text2** displays the **text1** file at the fiftieth line; then does the same for the **text2** file when you finish the first. See "Subcommands" for descriptions of **more** subcommands.

If the command is a positioning command, such as a line number or a regular expression search, set the current position to represent the final results of the command, without writing any intermediate lines of the file. For example, the two commands:

```
more -p 1000j filename
```

```
more -p 1000G filename
```

are functionally the same and will start the display with the current position at line 1000, passing the lines that j would write and would scroll off the screen if it had been issued during the file examination.

If the positioning command is unsuccessful, the first line in the file will be the current position.

-s Reduces multiple blank lines in the output to only one blank line.

-t *Tagstring* Displays the portion of the file that contains the specified tag.

-u Prevents the **more** command from treating a backspace character as a printable control character (displayed as a ^H (CTRL-H)), suppressing backspacing, underlining, or creating reverse video text for underlined information in a source file. The **-u** flag also forces the **more** command to recognize a carriage-return character, if it exists, at the end of a line.

-v Suppresses the graphical translation of nonprinting characters. Without the **-v** flag, the **more** command graphically interprets all non-ASCII and most control characters, except Tab, Backspace, and Return. For example, if you do not use the **-v** flag, the **more** command displays the non-ASCII characters Ctrl-x as `^X` and x as `M-x`.

-W *Option* Provides the specified *Option* to the **more** command as an extension:

notite
: Prevents the **more** command from sending the terminal initialization string (either the **ti termcap** or the **smcup terminfo** capability) before displaying the file. This option also prevents the **more** command from sending the terminal de-initialization string (either the **te termcap** or the **rmcup terminfo** capability) before exiting.

tite
: Causes the **more** command to send the initialization and de-initialization strings. This is the default.

These options control whether the **more** command sends the initialization strings described, which, for certain terminals (such as some xterms), cause the **more** command to switch to an alternative screen. The effect of switching screens is to erase the display of the file you were viewing.

-x *Tabs* Sets tab stops at the specified *Tabs* position. The default tab setting is 8 columns.

-z Graphically displays the Tab, Backspace, and Return control characters. With the **-z** flag, the **more** command translates the Backspace character as `^H`, Return as `^M`, and Tab as `^I`.

Subcommands

The **more** command accepts subcommands when the command pauses and as parameters for the **-p** flag. Many subcommands take an optional integer, symbolized here by *K*, which you must enter before the subcommand, with no space between. The **more** command, in the paused state, processes subcommands immediately and does not require you to press the Enter key.

The **more** command uses the following subcommands:

h Displays a help screen that describes the **more** subcommands.

v Starts the **vi** editor, editing the current file in the current line.

r or **^L** Refreshes the display.

R Refreshes the display and removes buffered input.

_K_(Spacebar) Moves forward _K_ lines when you press the spacebar. If you do not give a value for _K_, pressing the spacebar displays the next full screen by default. This spacebar subcommand is the same as _K_**f** or _K_**^F** or _K_**z**.

_K_**f** or _K_**^F** or _K_**z** Moves forward _K_ lines, or one full screen if you do not give a value for _K_.

_K_**b** or _K_**^B** Moves backward _K_ lines, or one full screen if you do not give a value for _K_.

_K_**d** or _K_**^D** Moves forward _K_ lines, or half a screen if you do not give a value for _K_. If you give a value for _K_, the **more** command sets the **d** and **u** scroll size to _K_ lines for the session.

_K_**u** or _K_**^U** Moves backward _K_ lines, or half a screen if you do not give a value for _K_. If you give a value for _K_, the **more** command sets the **d** and **u** scroll size to _K_ lines for the session.

_K_**j** or _K_(Enter) or _K_**^E** Moves forward _K_ lines, or one line if you do not give a value for _K_.

_K_**k** or _K_**^Y** Moves backward _K_ lines, or one line if you do not give a value for _K_.

_K_**g** Moves to the beginning of the file, unless you give a line number for _K_. The default for _K_ is line number 1.

_K_**G** Moves to the last line in the file, unless you give a line number for _K_. The default for _K_ is the last line in the file.

_K_**p** or _K_**%** Moves to the point in the file that is _K_ percent of the total file. The default for _K_ is one percent, or the first line in the file.

m_a-z_ Marks the current position in the file with the specified letter.

'_a-z_ (Single quote) Moves to the position marked with the specified letter.

'' (Two single quotes) Returns to the position from which the last large movement (moving more than a page) command was run. If no such movements have been made, returns to the beginning of the file.

_K_**/**_pattern_ (Slash) Searches forward, from the current position, for the specified occurrence of the specified pattern of characters. The default value for _K_ is the first occurrence.

_K_**/!**_pattern_ (Slash, exclamation mark) Searches forward, from the current position, for the specified occurrence of a line that does not contain the specified pattern of characters. The default value for _K_ is the first occurrence.

_K_**?**_pattern_ (Question mark) Searches backward, from the current position, for the specified occurrence of the specified pattern of characters. The default value for _K_ is the first occurrence.

_K_**?!**_pattern_ (Question mark, exclamation mark) Searches backward, from the current position, for the specified occurrence of a line that does not contain the specified pattern of characters. The default value for _K_ is the first occurrence.

_K_**n** Repeats the last search, specifying an occurrence of the pattern (or an occurrence *not* containing the pattern if the search subcommand included **!**). The default value for _K_ is the first occurrence.

:a Lists the file or files you named in the **more** command line.

:f or **^G** or **=** Displays information about the current file:

- file name
- order of the file in the list of files
- current line number
- current position in the file, given as a percentage
- current byte number and total bytes to display.

:e_File_ or **E**_File_ Examines the specified file, provided you named it in the **more** command line.

_K_**:n** or _K_**N** Examines either the next file (if you do not give a value for _K_) or the file _K_ number of positions forward in the list of files you named in the **more** command line.

_K_**:p** or _K_**P** Examines either the previous file (if you do not give a value for _K_) or the file _K_ number of positions backward in the list of files you named in the **more** command line.

:t _Tagstring_ Displays the portion of the file that contains the specified tag. This subcommand works only on files containing tags created with the **ctags** command. The **:t** subcommand is the

	interactive version of the **-t** flag.
:q or **q** or **Q**	Exits the **more** command.
:!*command* or **!***command*	Starts the specified command in a new shell.
H	Toggles the search pattern highlighting feature on or off.

Exit Status

This command returns the following exit values:

0	Successful completion.
>0	An error occurred.

Examples

1. To view a file named `myfile`, enter:

   ```
   more myfile
   ```
2. To view output from the **nroff** command, enter:

   ```
   ls -l | more
   ```
3. To view each file starting at its last screen, enter:

   ```
   more -p G file1 file2
   ```
4. To view each file with the 100th line at the current position, enter:

   ```
   more -p 100 file1 file2
   ```

   Typically, the current position in a **more** command display is the third line on the screen. In this example, the first line on the screen is the 98th line in the file.
5. To view each file starting with the first line that contains the `foo` string, enter:

   ```
   more -p /foo file1 file2
   ```

   The **more** command displays the line in the current position, the third line on the screen.

Files

/usr/share/lib/terminfo	Indicates the terminal information database.

Related Information

The **cat** command.

motd Command

Purpose

Displays or modifies the partition's message of the day file.

Syntax

motd

motd { **-append** | **-overwrite** } { **-file** *Filename* | "*Message of the day string*" }

Description

The **motd** command writes or appends to the partitions message of the day file. The new message can be specified on the command line or in a file with the **-file** flag. If no flags are specified the current message of the day is displayed.

Flags

-append	Appends the specified message to the current message of the day
-file *FileName*	Replaces the current message of the day with the contents of *FileName*
-overwrite	Replaces the current message of the day with the specified message

mount Command

Purpose

Makes a file system available for use.

Syntax

mount *Node*:*Directory Directory*

mount -cd *DeviceDirectory*

Description

The **mount** command instructs the operating system to make a file system available for use at a specified location (the mount point). The **mount** command mounts a file system expressed as a directory using the *Node*:*Directory* parameter on the directory specified by the *Directory* parameter. After the **mount** command has finished, the directory specified becomes the root directory of the newly mounted file system.

If you enter the mount command without flags, the command displays the following information for the mounted file systems:

- the node (if the mount is remote)
- the object mounted
- the mount point
- the virtual-file-system type
- the time mounted
- any mount options

The /**mnt** directory can be used as a local mount point, or you can create a directory using the **mkdir** command. Any directories created with the **mkdir** command must be a sub-directory of your home directory.

Flags

-cd	Specifies the cd device name on which to mount.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To list the mounted file systems, type:

   ```
   mount
   ```

   This command produces output similar to the following:

   ```
   node   mounted          mounted    vfs  date           options   over
   ----   -------          ---------  ---  ------------    -------  ---------
          /dev/hd0         /          jfs  Dec 17 08:04    rw, log  =/dev/hd8
          /dev/hd3         /tmp       jfs  Dec 17 08:04    rw, log  =/dev/hd8
          /dev/hd1         /home      jfs  Dec 17 08:06    rw, log  =/dev/hd8
          /dev/hd2         /usr       jfs  Dec 17 08:06    rw, log  =/dev/hd8
   sue    /home/local/src  /usr/code  nfs  Dec 17 08:06    ro, log  =/dev/hd8
   ```

   For each file system, the **mount** command lists the node name, the device name, the name under which it is mounted, the virtual-file-system type, the date and time it was mounted, and its options.
2. To mount the remote directory on to a local directory, enter:

   ```
   mount testsys3:/test /mnt
   ```

This command mounts the /**test** directory located on **testsys3** onto the local /**mnt** directory.

Related Information

The **backupios** command.

mv Command

Purpose

Moves files.

Syntax

To Move and Rename a File

mv -E{**force**|**ignore**|**warn**} **-i** | **-f -I** *SourceFile ... TargetFile*

To Move and Rename a Directory

mv -E{**force**|**ignore**|**warn**} **-i** | **-f -I** *SourceDirectory ... TargetDirectory*

To Move Files or Directories to a Directory Maintaining Original File Names

mv -E{**force**|**ignore**|**warn**} **-i** | **-f -I** *SourceFile/SourceDirectory TargetDirectory*

Description

Attention: The **mv** command can overwrite many existing files unless you specify the **-i** flag. The **-i** flag prompts you to confirm before it overwrites a file. If both the **-f** and **-i** flags are specified in combination, the last flag specified takes precedence.

The **mv** command moves files and directories from one directory to another or renames a file or directory. If you move a file or directory to a new directory, it retains the base file name. When you move a file, all links to other files remain intact, except when you move it to a different file system. When you move a directory into an existing directory, the directory and its contents are added under the existing directory.

When you use the **mv** command to rename a file or directory, the *TargetDirectory* parameter can specify either a new file name or a new directory path name.

If moving the file would overwrite an existing file that does not have write-permission set and if standard input is a workstation, the **mv** command displays the file-permission code and reads a line from standard input. If that line begins with a `y` or the locale's equivalent of a `y`, the **mv** command moves the file. If the response is anything other than a `y`, the **mv** command does nothing to that file and continues with the next specified file. The file-permission code displayed may not fully represent the access permission if the *TargetFile* is associated with an ACL. When the parent directory of the *SourceFile* is writable and has the sticky bit set, one or more of the following conditions are true:

- The user must own the file.
- The user must own the directory
- The user must be a privileged user.
- The file must be writable by the user.

This warning message and prompt for input can be overridden by using the **-f** option.

You can use the **mv** command to move files within the same file system or between file systems. Whether you are working in one file system or across file systems, the **mv** command copies the file to the target and deletes the original file. The **mv** command preserves in the new file the time of the most recent data modification, the time of the most recent access, the user ID, the group ID, the file mode, the extended attributes, and ACLs of the original file. For symbolic links, the **mv** command preserves only the owner and group of the link itself.

If it is unable to preserve the owner and group ID, the **mv** command clears S_ISUID and S_ISGID bits in the target. The **mv** command prints a diagnostic message to stderr if it is unable to clear these bits, though the exit code is not affected.

The **mv** command modifies either the source file or the destination path if the command is prematurely terminated.

Note: The **mv** command supports the **—** (dash, dash) parameter as a delimiter that indicates the end of the flags.

The **mv** command will not move an object if the object is exported as an NFS version 4 referral. The referral object is marked as busy and remains so until it is unexported.

Flags

Attention: The **mv** command can overwrite many existing files unless you specify the **-i** flag. The **-i** flag prompts you to confirm before it overwrites a file. If both the **-f** and **-i** flags are specified in combination, the last flag specified takes precedence.

-E
The **-E** option requires one of the following arguments. If you omit the **-E** option, **warn** is the default behavior.

force
Fails the **mv** operation on a file if the fixed extent size or space reservation of the file cannot be preserved.

ignore
Ignores any errors in preserving extent attributes.

warn
Issues a warning if the space reservation or the fixed extent size of the file cannot be preserved.

-f Does not prompt you before overwriting an existing file.

-i Prompts you before moving a file or directory to an existing path name by displaying the name of the file followed by a question mark. If you answer with a line starting with `y` or the locale's equivalent of a `y`, the move continues. Any other reply prevents the move from occurring.

-I Suppresses the warning message during ACL conversion.

Examples

1. To rename a file, enter:

   ```
   mv appendix apndx.a
   ```

   This command renames `appendix` to `apndx.a`. If a file named `apndx.a` already exists, its old contents are replaced with those of `appendix`.
2. To move a directory, enter:

   ```
   mv book manual
   ```

   This command moves all files and directories under `book` to the directory named `manual`, if `manual` exists. Otherwise, the directory `book` is renamed `manual`.
3. To move a file to another directory and give it a new name, enter:

   ```
   mv intro manual/chap1
   ```

This command moves `intro` to `manual/chap1`. The name `intro` is removed from the current directory, and the same file appears as `chap1` in the directory `manual`.

4. To move a file to another directory, keeping the same name, enter:

```
mv chap3 manual
```

This command moves `chap3` to `manual/chap3`

> **Note:** Examples 1 and 3 name two files, example 2 names two existing directories, and example 4 names a file and a directory.

5. To move several files into another directory, enter:

```
mv chap4 jim/chap5 /home/manual
```

This command moves the `chap4` file to the `/home/manual/chap4` file directory and the `jim/chap5` file to the `/home/manual/chap5` file.

6. To use the **mv** command with pattern-matching characters, enter:

```
mv manual/* .
```

This command moves all files in the `manual` directory into the current directory . (period), retaining the names they had in `manual`. This move also empties `manual`. You must type a space between the asterisk and the period.

> **Note:** Pattern-matching characters expand names of existing files only. For example, the command `mv intro man*/chap1` does not work if the file `manual/chap1` does not exist.

Exit Status

0	All input files were moved successfully.
>0	An error occurred.

Files

/usr/bin/mv	Contains the **mv** command.

Related Information

The **chmod** command and the **rm** command.

netstat Command

Purpose

Shows network status.

Syntax

To display active sockets for each protocol or routing table information

netstat -num -protocol *protocol* **-routtable -routinfo -state** *Interval*

To display the contents of a network data structure

netstat -stats | **-cdlistats -protocol** *protocol Interval*

To display the address resolution protocol (arp)

netstat -arp

To clear all statistics

netstat -clear

Description

The **netstat** command symbolically displays the contents of various network-related data structures for active connections. The *Interval* parameter, specified in seconds, continuously displays information regarding packet traffic on the configured network interfaces.

Flags

-arp Displays address resolution interfaces.

-cdlistats Shows statistics for CDLI-based communications adapters.

-clear Clears all statistics

-num Shows network addresses as numbers. When this flag is not specified, the **netstat** command interprets addresses where possible and displays them symbolically. This flag can be used with any of the display formats.

-protocol *Protocol* Shows statistics about the value specified for the *Protocol* variable, which is either a well-known name for a protocol or an alias for it. A null response means that there are no numbers to report. The program report of the value specified for the *Protocol* variable is unknown if there is no statistics routine for it.

-routinfo Shows the routing tables, including the user-configured and current costs of each route.

-routtable Shows the routing tables. When used with the **-stats** flag, the **-routtable** flag shows routing stistics. See "Routing Table Display."

-state Shows the state of all configured interfaces.

The interface display format provides a table of cumulative statistics for the following items:

- Errors
- Collisions **Note:** The collision count for Ethernet interfaces is not supported.
- Packets transferred

The interface display also provides the interface name, number, and address as well as the maximum transmission units (MTUs).

-stats Shows statistics for each protocol.

Default Display

The default display for active sockets shows the following items:

- Local and remote addresses
- Send and receive queue sizes (in bytes)
- Protocol
- Internal state of the protocol

Internet address formats are of the form *host.port* or *network.port* if a socket's address specifies a network but no specific host address. The host address is displayed symbolically if the address can be resolved to a symbolic host name, while network addresses are displayed symbolically.

NS addresses are 12-byte quantities, consisting of a 4-byte network number, a 6-byte host number and a 2-byte port number, all stored in network standard format. For VAX architecture, these are word and byte reversed; for the Sun systems, they are not reversed.

If a symbolic name for a host is not known or if the **-num** flag is used, the address is printed numerically, according to the address family. Unspecified addresses and ports appear as an * (asterisk).

Interface Display

The interface display format provides a table of cumulative statistics for the following items:

- Errors
- Collisions **Note:** The collision count for Ethernet interfaces is not supported.
- Packets transferred

The interface display also provides the interface name, number, and address as well as the maximum transmission units (MTUs).

Routing table display

The routing table display indicates the available routes and their statuses. Each route consists of a destination host or network and a gateway to use in forwarding packets.

A route is given in the format A.B.C.D/XX, which presents two pieces of information. A.B.C.D indicates the destination address and XX indicates the netmask associated with the route. The netmask is represented by the number of bits set. For example, the route 9.3.252.192/26 has a netmask of 255.255.255.192, which has 26 bits set.

The routing table contains the following ten fields:

Flags
The flags field of the routing table shows the state of the route:

A
An Active Dead Gateway Detection is enabled on the route

U
Up

H
The route is to a host rather than to a network

G
The route is to a gateway

D
The route was created dynamically by a redirect

M
The route has been modified by a redirect

L
The link-level address is present in the route entry

c
Access to this route creates a cloned route

W
The route is a cloned route

1

Protocol-specific routing flag #1

2
Protocol-specific routing flag #2

3
Protocol-specific routing flag #3

b
The route represents a broadcast address

e
Has a binding cache entry

l
The route represents a local address

m
The route represents a multicast address

P
Pinned route

R
Host or net unreachable

S
Manually added

u
Route usable

s
The group routing stopsearch option is enabled on the route

Gateway
Direct routes are created for each interface attached to the local host. The gateway field for these entries shows the address of the outgoing interface.

Refs
Gives the current number of active uses for the route. Connection-oriented protocols hold on to a single route for the duration of a connection, while connectionless protocols obtain a route while sending to the same destination.

Use
Provides a count of the number of packets sent using that route.

PMTU
Gives the Path Maximum Transfer Unit (PMTU).

Interface
Indicates the network interfaces utilized for the route.

Exp
Displays the time (in minutes) remaining before the route expires.

Groups
Provides a list of group IDs associated with that route.

Netmasks
Lists the netmasks applied on the system.

Route Tree for Protocol Family
Specifies the active address families for existing routes. Supported values for this field are as follows:

1
Specifies the UNIX address family

2
Specifies the Internet address family (for example, TCP and UDP)

3
Specifies the Xerox Network System (XNS) address family

When a value is specified for the *Interval* parameter, the **netstat** command displays a running count of statistics related to network interfaces. This display contains two columns: a column for the primary interface (the first interface found during autoconfiguration) and a column summarizing information for all interfaces. The first line of each screen of information contains a summary of statistics accumulated since the system was last restarted. The subsequent lines of output show values accumulated over intervals of the specified length.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To display routing table information for an Internet interface, type:

   ```
   netstat -routtable
   ```

   This produces the output similar to the following:

   ```
   Routing tables
   ```

```
Destination      Gateway           Flags   Refs     Use  If   PMTU Exp Groups

Route tree for Protocol Family 2 (Internet):
default          129.3.141.1       UGc       0        0  en0     -   -
129.33.140/23    127.0.0.1         U         6       53  en0     -   -
129.33.41.2      localhost         UGHS      6      115  lo0     -   -
129.45.41.2      129.3.41.1        UGHW      1      602  en0  1500   -
dcefs100         129.31.41.1       UGHW      1        2  en0     -   -
192.100.61       localhost         U         7    14446  lo0     -   -

Route tree for Protocol Family 24 (Internet v6):
::1              ::1               UH        0        0  lo0 16896   -
```

2. To display interface information for an Internet interface, type:

   ```
   netstat -state
   ```

   This produces the output similar to the following:

   ```
   Name  Mtu   Network     Address             Ipkts Ierrs    Opkts Oerrs  Coll
   en0   1500  link#2      0.5.20.4.0.4e      874986     0    22494     0     0
   en0   1500  90.34.14    hostname           874986     0    22494     0     0
   lo0   16896 link#1                          14581     0    14590     0     0
   lo0   16896 129         localhost           14581     0    14590     0     0
   lo0   16896 ::1                             14581     0    14590     0     0
   ```

Related Information

The **mktcpip** command, the **hostname** command, the **startnetsvc** command, the **stopnetsvc** command, the **cfglnagg** command, the **entstat** command, the **cfgnamesrv** command, the **hostmap** command, the **traceroute** command, the **ping** command, the **optimizenet** command.

oem_platform_level Command

Purpose

Returns the operating system level of the OEM install and setup environment.

Syntax

oem_platform_level

Description

The **oem_platform_level** command displays the name and version of the underlying Virtual I/O Server operating system.

This command can only be executed by the prime administrator.

Exit Status

See Virtual I/O Server command exit status

Examples

1. To get the operating system level of the OEM install and setup environment, run the following command:

   ```
   oem_platform_level
   ```

Related Information

The **lssw** command, the **ioslevel** command, the **remote_management** command, the **oem_setup_env** command, and the **updateios** command.

oem_setup_env Command

Purpose

Initiates the OEM install and setup environment.

Syntax

oem_setup_env

Description

The **oem_setup_env** command places the user into the OEM software install and setup environment. In this environment, the user will be able to install and setup OEM software by following the installation instructions provided with each software package. After the software is installed, the user will need to create a link in the /**usr**/**ios**/**oem**/ directory to any new commands that will run from the Virtual I/O Server command line. After these links have been created, the commands will be accessible by all Virtual I/O Server users. Note however that these commands will not run with root authority.

Upon completion of installing all desired software, typing `exit` will return the user to the Virtual I/O Server prompt.

Only the prime administrator can execute this command.

Note: The **oem_setup_env** command will place the **padmin** user in a non-restricted UNIX root shell with a home directory in the /**home**/**padmin** directory. The user can then run any command available to the root user. This is not a supported Virtual I/O Server administration method. The purpose of this command is to allow installation of vendor software, such as device drivers.

Exit Status

See Virtual I/O Server command exit status

Examples

1. To initiate the OEM setup and install environment, type the following:

```
oem_setup_env
```

Related Information

The **lssw** command, the **ioslevel** command, the **remote_management** command, the **updateios** command, and the **oem_platform_level** command.

optimizenet Command

Purpose

Manages network tuning parameters.

Syntax

optimizenet -reboot | **-perm** { **-set** *Tunable=NewValue* | **-default** *Tunable* }

optimizenet -list *Tunable*

optimizenet -h *Tunable*

Description

The **optimizenet** command is used to configure network tuning parameters. The **optimizenet** command sets or displays current or next boot values for network tuning parameters. This command can also make permanent changes or defer changes until the next reboot. Whether the command sets or displays a parameter is determined by the accompanying flag. The **-set** flag performs both actions. It can either display the value of a parameter or set a new value for a parameter.

If the **-list** flag is specified without any *Tunables*, only *Tunables* modifiable by this command will be displayed.

Flags

-default *Tunable*
Resets *Tunable* to its default value. If the *Tunable* needs to be changed (not currently set to its default value) and it is of type `Reboot`, it will not be changed; a warning displays instead.

-h *Tunable*
Displays help about Tunable parameter if one is specified.

-list Tunable
Lists the characteristics of one or all Tunables, one per line, using the following format:

```
NAME                      CUR DEF BOOT MIN MAX  UNIT          TYPE
   DEPENDENCIES
-----------------------------------------------------------------
General Network Parameters
-----------------------------------------------------------------
sockthresh                85  85  85   0   100  %_of_thewall D
-----------------------------------------------------------------
fasttimo                  200 200 200  50  200  millisecond  D
-----------------------------------------------------------------
inet_stack_size           16  16  16   1        kbyte        R
-----------------------------------------------------------------
```

where

CUR
current value

DEF
default value

BOOT
reboot value

MIN
minimal value

MAX
maximum value

UNIT
tunable unit of measure

TYPE
Parameter type: D (Dynamic), S (Static), R (Reboot), B (Bosboot), M (Mount), I (Incremental), and C (Connect)

DEPENDENCIES
List of dependent tunable parameters, one per line

-perm

Makes changes apply to both current and reboot values when used in combination with **-set** or **-default**. These combinations cannot be used on Reboot type parameters because their current value can't be changed.

When used with **-set** without specifying a new value, values displays only if the current and next boot values for a parameter are the same. Otherwise NONE displays as the value.

-reboot

Makes changes apply to reboot values when used in combination with **-set** or **-default**. When used with **-set** without specifying a new value, next boot values for *tunables* display instead of the current values.

-set *Tunable =NewValue*

Displays the value or sets the Tunable to *NewValue*. If a tunable needs to be changed (the specified value is different than current value), and is of type **Reboot** it will not be changed but a warning displays instead.

When **-reboot** is used in combination without a new value, the **nextboot** value for Tunable is displayed. When **-perm** is used in combination without a new value, a value displays only if the current and next boot values for tunable are the same Otherwise NONE displays as the value.

Network tunable parameters

arptab_bsiz

Purpose:
Specifies Address Resolution Protocol (ARP) table bucket size.

Values:
◊ Default: 7
◊ Range: 1 to MAXSHORT
◊ Type: Reboot

Diagnosis
netstat -protocol arp will show the number of ARP packets sent and the number of ARP entries purged from the ARP table. If large number of entries are being purged, the ARP table size should be increased.

arptab_nb

Purpose:
Specifies the number of ARP table buckets.

Values:
◊ Default: 73
◊ Range: 1 to MAXSHORT
◊ Type: Reboot

Diagnosis:
netstat -protocol arp will show the number of ARP packets sent and the number of ARP entries purged from the ARP table. If large number of entries are being purged, the ARP table size should be increased.

Increase this value for systems that have a large number of clients or servers. The default provides for 73 x 7 = 511 ARP entries, but assumes an even hash distribution.

clean_partial_conns

Purpose:
Specifies whether or not SYN (synchronizes the sequence number) attacks are being avoided.

Values:
◊ Default: 0 (off)
◊ Range: 0 or 1
◊ Type: Dynamic

Tuning:
This option should be turned on for servers that need to protect against network attacks. If on, randomly removes partial connections to make room for new non-attack connections.

net_malloc_police

Purpose:

Specifies the size of the **net_malloc** and **net_free** trace buffers.

Values:

◊ Default: 0
◊ Range: 0 to MAXINT
◊ Type: Dynamic

Tuning:

If the value of this variable is non-zero, all **net_malloc** and **net_free** buffers will be traced in a kernel buffer and by system trace hook HKWD_NET_MALLOC. Additional error-checking will also be enabled. This includes checks for freeing a free buffer, alignment, and buffer overwrite. Enable this parameter only when investigating some network problem, because performance is affected negatively when turned on. The default value is zero (policing off). Values of **net_malloc_police** larger than 1024 will allocate that many items in the kernel buffer for tracing.

rfc1323

Purpose:

Enables window scaling and timestamps as specified by RFC 1323 (TCP Extensions for High Performance). Window scaling allows the TCP window sizes (**tcp_recvspace** and **tcp_sendspace**) to be larger than 64KB (65536) and is typically used for large MTU networks.

Values:

◊ Default: 0 (off)
◊ Range: 0 or 1
◊ Type: Connect

Tuning:

The default value of 0 disables the RFC enhancements on a systemwide scale. A value of 1 specifies that all TCP connections will attempt to negotiate the RFC enhancements. Make changes before attempting to set **tcp_sendspace** and **tcp_recvspace** to more than 64 KB.

route_expire

Purpose:

Specifies whether unused routes created by cloning, or created and modified by redirects expire.

Values:

◊ Default: 1 (on)
◊ Range: 0 or 1
◊ Type: Dynamic

Tuning:

A value of 1 allows route expiration, which is the default. Negative values are not allowed for this option.

tcp_pmtu_discover

Purpose:

Enables or disables path MTU discovery for TCP applications.

Values:

◊ Default: 1
◊ Range: 0 or 1
◊ Type: Dynamic

Tuning:

A value of 0 disables path MTU discovery for TCP applications, while a value of 1 enables it.

tcp_recvspace

Purpose:

Specifies the system default socket buffer size for receiving data. This affects the window size used by TCP.

Values:

◊ Default: 16384 bytes
◊ Range: 4096 to 1048576
◊ Type: Connect

Diagnosis:

Setting the socket buffer size to 16 KB (16,384) improves performance over standard Ethernet and Token-Ring networks. Lower bandwidth networks, such as Serial Line Internet

Protocol (SLIP), or higher bandwidth networks, such as Serial Optical Link, should have different optimum buffer sizes. The optimum buffer size is the product of the media bandwidth and the average round-trip time of a packet. For high speed networks, like gigabit Ethernet or ATM 622, a value of 65536 should be used for the minimum size for best performance.

For values larger than 65536, you must enable **rfc1323** (rfc1323=1) to enable TCP window scaling.

tcp_sendspace

Purpose:
Specifies the system default socket buffer size for sending data.

Values:
◊ Default: 16384 bytes
◊ Range: 4096 to 1048576
◊ Type: Connect

Tuning:
This affects the window size used by TCP. Setting the socket buffer size to 16 KB (16,384) improves performance over standard Ethernet networks. Lower bandwidth networks, such as Serial Line Internet Protocol (SLIP), or higher bandwidth networks, such as Serial Optical Link, should have different optimum buffer sizes. The optimum buffer size is the product of the media bandwidth and the average round-trip time of a packet: (optimum_window=bandwidth * average_round_trip_time) For high speed networks, like gigabit Ethernet or ATM 622, a value of 65536 should be used for the minimum size for best performance. For values larger than 65536, you must enable **rfc1323** (rfc1323=1) to enable TCP window scaling.

udp_recvspace

Purpose:
Specifies the system default socket-buffer size for receiving UDP data.

Values:
◊ Default: 42080 bytes
◊ Range: 4096 to 1048576
◊ Type: Connect

Diagnosis:
Nonzero n in **netstat -stats** report of udp: *n* socket buffer overflows

Tuning:
Increase size, preferably to multiple of 4096.

udp_sendspace

Purpose:
Specifies the system default socket-buffer size for sending UDP data.

Values:
◊ Default: 9216 bytes
◊ Range: 4096 to 1048576
◊ Type: Connect

Diagnosis:
Increase size, preferably to multiple of 4096.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To display the maximum size of the mbuf pool, type:

   ```
   optimizenet -set thewall
   ```
2. To change the default socket buffer sizes on your system, type:

```
optimizenet -reboot -set tcp_sendspace=32768
```

3. To use a machine as an internet work router over TCP/IP networks, type:

   ```
   optimizenet -set ipforwarding=1
   ```

4. To list the current and reboot value, range, unit, type and dependencies of the **arptab_bsiz** parameter, type:

   ```
   optimizenet -list arptab_bsiz
   ```

5. To display help information on **arptab_bsiz**, type:

   ```
    optimizenet -h arptab_bsiz
   ```

Related Information

The **entstat** command, the **lsnetsvc** command, the **mktcpip** command, the **netstat** command, and the **traceroute** command.

passwd Command

Purpose

Changes a user's password.

Syntax

passwd *User*

Description

The **passwd** command sets and changes passwords for users. Use this command to change your own password (all users) or another user's password (padmin only). To change your own password, enter the **passwd** command. The **passwd** command prompts the non-padmin user for the old password and then prompts for the new password twice. The password is never displayed on the screen. If the two entries of the new password do not match, the **passwd** command prompts for the new password again.

Password policy is checked during a password change. Construct locally-defined passwords according to the following password restrictions:

minother	Specifies the minimum number of other characters.
minlen	Specifies the minimum number of characters.
maxrepeats	Specifies the maximum number of times a single character can be used in a password.
maxage	Specifies the maximum age of a password. A password must be changed after a specified amount of time measured in weeks.
maxexpired	Specifies the maximum number of weeks beyond the maxage value that a password can be changed by the user.
histexpire	Specifies the number of weeks that a user cannot reuse a password.
histsize	Specifies the number of previous passwords that the user cannot reuse.

Exit Status

See Virtual I/O Server command exit status.

Security

Changing a password other than your own requires prime administrator authority.

Examples

1. To change the password for user account **heerlen**, type:

   ```
   passwd heerlen
   ```

   The user will then be prompted to enter the new password.

Related Information

The **chuser** command, the **lsuser** command, the **mkuser** command, and the **rmuser** command.

pdump Command

Purpose

Perform platform (hardware and firmware) dump-related actions.

Syntax

pdump -reserve *fstype* | **-enable** | **-disable** | **-ls** | **-size**

Description

The **pdump** command helps the operating system save firmware and hardware related dumps. This command also provides an estimate of the disk space required for storing these dumps. Note that platform and scan dumps are saved in order to capture the state of the firmware and the hardware for analysis.

Flags

-disable Disables platform dumps.

-enable Enables platform dumps.

-ls Lists the current configuration of platform dump.

-reserve *fstype* Reserves enough disk space on the system for platform dumps. The **-enable** option will create a file system (if one does not exist) exclusively for platform dumps. If a file system already exists and the size is not enough, the file system size will be increased. The *fstype* must be a valid file system type. If the file system already exists, any may be specified.

-size Provides an estimate of disk space required to save the platform dumps when they occur. This option will interact with the firmware to provide this estimate. It is expected that, based on this space information, the user will have enough disk space allocated for platform dumps to be saved. The value output will be the required size in bytes.

ping Command

Purpose

Sends an echo request to a network host.

Syntax

ping **-n** **-r** **-s** *PacketSize* **-src** *hostname*/*IP_addr* *Host* *Count*

Description

The **ping** command sends an Internet Control Message Protocol (ICMP) ECHO_REQUEST to obtain an ICMP ECHO_RESPONSE from a host or gateway. The **ping** command is useful for:

- Determining the status of the network and various foreign hosts.
- Tracking and isolating hardware and software problems.
- Testing, measuring, and managing networks.

If the host is operational and on the network, it responds to the echo. Each echo request contains an Internet Protocol (IP) and ICMP header, followed by a timeval structure, and enough bytes to fill out the packet. The default is to continuously send echo requests until an Interrupt is received (Ctrl-C).

The **ping** command sends one datagram per second and prints one line of output for every response received. The **ping** command calculates round-trip times and packet loss statistics, and displays a brief summary on completion. The ping command completes when the program times out or on receipt of a SIGINT signal. The Host parameter is either a valid host name or Internet address.

By default, the **ping** command will continue to send echo requests to the display until an Interrupt is received (Ctrl-C). Because of the load that continuous echo requests can place on the system, repeated requests should be used primarily for problem isolation.

Flags

-n	Specifies numeric output only. No attempt is made to look up symbolic names for host addresses.
-r	Bypasses the routing tables and sends directly to a host on an attached network. If the Host is not on a directly connected network, the ping command generates an error message. This option can be used to **ping** a local host through an interface that no longer has a route through it.
-s *PacketSize*	Specifies the number of data bytes to be sent. The default is 56, which translates into 64 ICMP data bytes when combined with the 8 bytes of ICMP header data.
-src *hostname/IP_addr*	Uses the IP address as the source address in outgoing ping packets. On hosts with more than one IP address, the **-src** flag can be used to force the source address to be something other than the IP address of the interface on which the packet is sent. If the IP address is not one of the machine's interface addresses, an error is returned and nothing is sent.

Parameters

Count	Specifies the number of echo requests to be sent (and received). This parameter is included for compatibility with previous versions of the **ping** command.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To check the network connection to host **canopus** and specify the number of echo requests to send, enter:

   ```
   ping canopus 5
   ```

   Information similar to the following is displayed:

   ```
   PING canopus.austin.century.com: (128.116.1.5): 56 data bytes
   64 bytes from 128.116.1.5: icmp_seq=0 ttl=255 time=2 ms
   64 bytes from 128.116.1.5: icmp_seq=1 ttl=255 time=2 ms
   64 bytes from 128.116.1.5: icmp_seq=2 ttl=255 time=3 ms
   64 bytes from 128.116.1.5: icmp_seq=3 ttl=255 time=2 ms
   64 bytes from 128.116.1.5: icmp_seq=4 ttl=255 time=2 ms

   ----canopus.austin.century.com PING Statistics----
   5 packets transmitted, 5 packets received, 0% packet loss
   round-trip min/avg/max = 2/2/3 ms
   ```
2. To obtain information about host **opus** and specify the number of data bytes to be sent, enter:

   ```
   ping -s 2000 opus
   ```

   Information similar to the following is displayed:

   ```
   PING opus.austin.century.com: (129.35.34.234): 2000 data bytes
   2008 bytes from 129.35.34.234: icmp_seq=0 ttl=255 time=20 ms
   2008 bytes from 129.35.34.234: icmp_seq=1 ttl=255 time=19 ms
   2008 bytes from 129.35.34.234: icmp_seq=2 ttl=255 time=20 ms
   2008 bytes from 129.35.34.234: icmp_seq=3 ttl=255 time=20 ms
   2008 bytes from 129.35.34.234: icmp_seq=4 ttl=255 time=20 ms
   2008 bytes from 129.35.34.234: icmp_seq=5 ttl=255 time=19 ms
   2008 bytes from 129.35.34.234: icmp_seq=6 ttl=255 time=19 ms
   ^C
   ----opus.austin.century.com PING Statistics----
   7 packets transmitted, 7 packets received, 0% packet loss
   round-trip min/avg/max = 19/19/20 ms
   ```

   Note: The output is repeated until an Interrupt (Ctrl-C) is received.

Related Information

The **optimizenet** command and the **traceroute** command.

redefvg Command

Purpose

Redefines the set of physical volumes of the given volume group in the device configuration database.

Syntax

redefvg { **-dev** *Device* | **-vgid** *Vgid* } *VolumeGroup*

Description

During normal operations, the device configuration database remains consistent with the Logical Volume Manager (LVM) information in the reserved area on the physical volumes. If inconsistencies occur between the device configuration database and the LVM, the **redefvg** command determines which physical volumes belong to the specified volume group and enters this information in the device configuration database. The **redefvg** command checks for inconsistencies by reading the reserved areas of all the configured physical volumes attached to the system.

Note: To use this command, you must either have root user authority or be a member of the system group.

Flags

-dev *Device*	The volume group ID, *Vgid*, is read from the specified physical volume device. You can specify the Vgid of any physical volume belonging to the volume group that you are redefining.
-vgid *Vgid*	The volume group identification number of the volume group to be redefined.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To synchronize the copies on physical volumes hdisk04, type:

   ```
   redefvg -dev hdisk04
   ```
2. To synchronize the copies on volume groups vg04 and vg05, type:

   ```
   redefvg -vgid vg04 vg05
   ```

Related Information

The **mkvg** command, the **syncvg** command, the **chvg** command, the **extendvg** command, the **lsvg** command, the **mirrorios** command, the **unmirrorios** command, the **activatevg** command, the **deactivatevg** command, the **importvg** command, the **exportvg** command, and the **reducevg** command.

reducevg Command

Purpose

Removes physical volumes from a volume group. When all physical volumes are removed from the volume group, the volume group is deleted.

Syntax

reducevg **-f** **-rmlv** *VolumeGroup PhysicalVolume* ...

Description

The **reducevg** command removes one or more physical volumes represented by the *PhysicalVolume* parameter from the *VolumeGroup*. When you remove all physical volumes in a volume group, the volume group is also removed.

Note: Sometimes a disk is removed from the system without first running the **reducevg** command. The VGDA still has this removed disk in its memory, but the *PhysicalVolume* name no longer exists or has been reassigned. To remove references to this missing disk you can still use **reducevg**, but with the Physical Volume ID (PVID) instead of the disk name: `reducevg VolumeGroup PVID`.

Flags

-f Removes the requirement for user confirmation when the **-rmlv** flag is used.
Deallocates the existing logical volume partitions and then deletes resultant empty logical volumes from the specified physical volumes. User confirmation is required unless the **-f** flag is added.

-rmlv **Attention:** The **reducevg** command with the **-rmlv** flag automatically deletes all logical volume data on the physical volume before removing the physical volume from the volume group. If a logical volume spans multiple physical volumes, the removal of any of those physical volumes may jeopardize the integrity of the entire logical volume.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To remove physical volume **hdisk1** from volume group **vg01**, type:

   ```
   reducevg vg01 hdisk1
   ```
2. To remove physical volume **hdisk1** and all residing logical volumes from volume group **vg01** without user confirmation, type:

   Attention: The **reducevg** command with the **-rmlv** flag automatically deletes all logical volume data before removing the physical volume.

   ```
   reducevg -rmlv -f vg01 hdisk1
   ```

   The physical volume **hdisk1** and all residing logical volumes are removed.

Related Information

The **mkvg** command, the **chvg** command, the **extendvg** command, the **lsvg** command, the **mirrorios** command, the **unmirrorios** command, the **activatevg** command, the **deactivatevg** command, the **importvg** command, the **exportvg** command, and the **syncvg** command.

remote_management Command

Purpose

Enables the Virtual I/O Server to be remotely managed by an AIX NIM master.

Syntax

To enable the Virtual I/O Server to be remotely managed by an AIX NIM master:

remote_management **-interface** *Interface Master*

To disable remote management:

remote_management **-disable**

Description

The **remote_management** command will setup the Virtual I/O Server to allow remote management from a NIM master. The *Master* parameter specifies the NIM master *hostname*. The *Interface* parameter specifies the network interface to be used to connect to the NIM master. If *Interface* is not specified, the default network interface used will be **en0**.

Once remote management has been enabled on the Virtual I/O Server, typical NIM functions, such as update, backup, and reinstall, can be initiated from the NIM master.

Flags

-disable	Disables the Virtual I/O Server NIM client daemon.
-interface	Specifies which network interface to use. If no network interface is specified, interface **en0** will be used.

Exit Status

See Virtual I/O Server command exit status

Examples

1. To enable remote_management using NIM master **nimsys01**, type:

   ```
   remote_management nimsys01
   ```
2. To disable remote_management, type:

   ```
   remote_management -disable
   ```

Related Information

The **lssw** command, the **ioslevel** command, the **updateios** command, the **oem_setup_env** command, and the **oem_platform_level** command.

restorevgstruct Command

Purpose

Restores the user volume group.

Syntax

restorevgstruct { **-ls** | **-vg** *VolumeGroupLabel DiskName* ... }

Description

The **restorevgstruct** command restores the structure of a previously-saved user volume group. If the **-ls** flag is specified, a list of previously-saved volume groups and the date each volume group was saved is displayed. This command does not work on rootvg.

Flags

DiskName...	Specifies the names of disk devices to be used instead of the disk devices saved in the volume group structure. Target disk devices must be defined as empty physical volumes; that is, they must contain a physical volume identifier and must not belong to a volume group. If the target disk devices belong to a volume group, they must be removed from the volume group using the **reducevg** command.
-ls	Displays a list of previously saved volume groups.
-vg	Specifies the name of the VolumeGroup to restore.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To restore the volume group **myvg**, onto the **hdisk2** and **hdisk3** disks, enter:

   ```
   restorevgstruct myvg hdisk2 hdisk3
   ```
2. To list all previously saved volume groups, enter:

   ```
   restorvgstruct -ls
   ```

   The message generated would be similar to:

   ```
   -rw-r--r--   1 root     system        51200 Jun 18 10:53 myvg.data
   -rw-r--r--   1 root     system        51200 Jun 18 10:53 myvg2.data
   ```

Related Information

The **activatevg** command, the **savevgstruct** command, the **chvg** command, the **deactivatevg** command, the **exportvg** command, the **extendvg** command, the **importvg** command, the **lsvg** command, the **mkvg** command, and the **syncvg** command.

rm Command

Purpose

Removes (unlinks) files or directories.

Syntax

rm **-f** **-r** **-R** **-i** **-e** *File ...*

Description

The **rm** command removes the entries for the specified *File* parameter from a directory. If an entry is the last link to a file, the file is then deleted. If you do not have write permission for a file and the standard input is a terminal, you are prompted with the file name and ask to confirm that you want to delete the file. If you type a `y` (for yes), the file is deleted, type any other character and the file is not deleted. You do not need read or write permission for the file you want to remove. However, you must have write permission for the directory containing the file.

If the file is a symbolic link, the link is removed, but the file or directory that the symbolic link refers to remains. You do not need write permission to delete a symbolic link, if you have write permission in the directory.

If either of the files `.` (dot) or `..` (dot, dot) are specified as the base name portion of the *File* parameter, the **rm** command writes a diagnostic message to standard error and does nothing more with such parameters.

The **rm** command writes a prompt to standard error and reads a line from standard input if the **-f** flag is not specified, and either the *File* parameter does not have write permission and the standard input is a workstation, or the **-i** flag is specified. If the response is not affirmative, the **rm** command does nothing more with the current file and proceeds to the next file.

The files owned by other users cannot be removed if the sticky bit of the directory is set and the directory is not owned by the user.

Note: The **rm** command supports the **—** (dash, dash) parameter as a delimiter that indicates the end of the flags.

An attempt to remove a file or directory that has been exported for use by the NFS version 4 server will fail with a message saying that the resource is busy. The file or directory must be unexported for NFS version 4 use before it can be removed.

Flags

-e Displays a message after each file is deleted.

-f Does not prompt before removing a write-protected file. Does not display an error message or return error status if a specified file does not exist. If both the **-f** and **-i** flags are specified, the last one specified takes affect.

-i Prompts you before deleting each file. When you use the **-i** and **-r** flags together, the **rm** command also prompts before deleting directories. If both the **-i** and **-f** flags are specified, the last one specified takes affect.

-r Permits recursive removal of directories and their contents when the *File* parameter is a directory. This flag is equivalent to the **-R** flag.

-R Permits recursive removal of directories and their contents when the *File* parameter is a directory. This flag is equivalent to the **-r** flag.

Exit Status

This command returns the following exit values:

0	If the **-f** flag was not specified, all the named directory entries were removed; otherwise, all the existing named directory entries were removed.
>0	An error occurred.

Examples

1. To delete a file, enter:

   ```
   rm myfile
   ```

   If there is another link to this file, then the file remains under that name, but the name `myfile` is removed. If `myfile` is the only link, the file itself is deleted.
2. To delete a file without first receiving a confirmation prompt, enter: `rm  -f core`
   No confirmation prompt is issued before the **rm -f** command attempts to remove the file named `core`. However, an error message displays if the `core` file is write-protected and you are not the owner of the file or you do not have root authority. No error message displays when the **rm -f** command attempts to remove nonexistent files.
3. To delete files one by one, enter: `rm  -i mydir/*`
   After each file name is displayed, enter `y` to delete the file, or press the Enter key to keep it.
4. To delete a directory tree, enter:

   ```
   rm -ir manual
   ```

   This command recursively removes the contents of all subdirectories of the `manual` directory, prompting you regarding the removal of each file, and then removes the `manual` directory itself, for example:

   ```
   You:  rm -ir manual
   System: rm: Select files in directory manual? Enter y for yes.
   You:  y
   System: rm: Select files in directory manual/draft1? Enter y for yes.
   You:  y
   System: rm: Remove manual/draft1?
   You:  y
   System: rm: Remove manual/draft1/chapter1?
   You:  y
   System: rm: Remove manual/draft1/chapter2?
   You:  y
   System: rm: Select files in directory manual/draft2? Enter y for yes.
   You:  y
   System: rm: Remove manual/draft2?
   You:  y
   System: rm: Remove manual?
   You:  y
   ```

   Here, the **rm** command first asks if you want it to search the `manual` directory. Because the `manual` directory contains directories, the **rm** command next asks for permission to search `manual/draft1` for files to delete, and then asks if you want it to delete the `manual/draft1/chapter1` and `manual/draft1/chapter2` files. The **rm** command next asks for permission to search the `manual/draft2` directory. Then asks for permission to delete the `manual/draft1`, `manual/draft2`, and `manual` directories.

   If you deny permission to remove a subdirectory (for example, `manual/draft2`), the **rm** command does not remove the `manual` directory. Instead, you see the message: `rm: Directory manual not empty.`

Files

/usr/bin/rm	Contains the **rm** command.

rmbdsp Command

Purpose

Remove a backing device and return the storage back to the storage pool.

Syntax

rmbdsp {**-bd** *LogicalVolume* | **-vtd** *VirtualTargetDevice*} **-savebd**

Description

The **rmbdsp** command removes a backing device from a virtual SCSI server adapter by removing its associated virtual target device. By default the backing device is also removed and its storage returned to the storage pool. If the **-savebd** flag is included then the backing device is not removed. The backing device can be identified by either specifying the name, **-bd**, or the virtual target device, **-vtd**. This command only works with logical volume backing devices.

Flags

-bd *BackingDevice*	Specifies the backing device that is the backing device.
-savebd	Instructs the command not to remove the backing device.
-vtd	Specifies the virtual target devices that associate the backing device with the virtual SCSI adapter.

Exit Status

25	Specified logical volume is not a backing device.

Examples

1. To remove the virtual target device vtscsi4 and not remove the backing device associated with it, type:

   ```
   rmbdsp -vtd vtscsi4 -savebd
   ```

rmdev Command

Purpose

Removes a device from the system.

Syntax

rmdev { **-dev** | **-pdev** } *Name* **-recursive** **-ucfg**

Description

The **rmdev** command unconfigures and undefines the device specified with the device logical name. If you specify the **-recursive** flag, the **rmdev** command acts on any children of the device as well. By specifying the **-ucfg** flag the device will be unconfigured but not undefined.

Use the **-pdev** flag along with the parent device's logical name to delete all of the children devices. The children are deleted in the same recursive fashion as described above for the **-recursive** flag. The only difference is that the specified device itself is not deleted. Thus, the **-recursive** flag is redundant and need not be specified with the **-pdev** flag.

Attention: To protect the Configuration database, the **rmdev** command is not interruptible. Stopping this command before execution is complete could result in a corrupted database.

Flags

-dev *Name*	Specifies the logical device, indicated by the Name parameter. This flag may not be used with the **-pdev** flag.
-pdev *Name*	Specifies the parent logical device (indicated by the Name parameter whose children need to be removed. This flag may not be used with the **-dev** flag.
-recursive	Unconfigures the device and its children.
-ucfg	Unconfigures, but does not undefine, the specified device. The device's state will be moved from `Available` to `Defined`. To move the device back to `Available` state run `cfgdev -dev` *`Name`*

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To unconfigure the **cd0** CD-ROM device, type:

```
rmdev  -dev cd0
```

2. To unconfigure the SCSI adapter **scsi1** and all of its children, type:

```
 rmdev  -recursive -dev scsi1
```

3. To unconfigure just the children of the SCSI adapter **scsi1**, but not the adapter itself, type:

```
rmdev  -pdev scsi1
```

4. To unconfigure the children of PCI bus pci1 and all other devices under them, type:

```
rmdev  -pdev pci1
```

Related Information

The **cfgdev** command, the **chdev** command, the **lsdev** command, the **mkvdev** command, and the **rmdev** command.

rmlv Command

Purpose

Removes logical volumes from a volume group.

Syntax

rmlv **-f** *LogicalVolume* ...

Description

The **rmlv** command removes a logical volume. The *LogicalVolume* parameter can be a logical volume name or logical volume ID.

Attention: This command destroys all data in the specified logical volumes.

Flags

-f	Removes the logical volumes without requesting confirmation.

Exit Status

See Virtual I/O Server command exit status.

Examples

Attention: The following command destroys all data in the logical volumes.

1. To remove logical volume **lv05** without requiring user confirmation, enter the following command:

   ```
   rmlv -f lv05
   ```

   The logical volume is removed from the volume group.

Related Information

The **mklv** command, the **extendlv** command, and the **lslv** command.

rmlvcopy Command

Purpose

Removes a copy of a logical volume.

Syntax

rmlvcopy *LogicalVolume PhysicalVolume* ...

Description

The **rmlvcopy** command removes one of the copies (disabling mirroring) of the logical volume. The *LogicalVolume* parameter can be a logical volume name or logical volume ID. The *PhysicalVolume* parameter can be the physical volume name or the physical volume ID. If the *PhysicalVolume* parameter is used, then only the copy from that physical volume will be removed.

Note: If the LVM has not recognized that a disk has failed, it is possible that the LVM will remove a different mirror. Therefore, if you know that a disk has failed and the LVM does not show those disks as missing, you should specify the failed disks on the command line.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To remove mirroring from the logical volume **lv0112**, type:

   ```
   rmlvcopy lv0112
   ```

Related Information

The **mklv** command, the **extendlv** command, the **rmlv** command, and the **lslv** command.

rmpath Command

Purpose

Removes from the system a path to an MPIO-capable device.

Syntax

rmpath { **-dev** *Name* **-pdev** *Parent* **-conn** *Connection* } **-rm**

Description

The **rmpath** command unconfigures, and possibly undefines, one or more paths associated with the specified target device (**-dev** *Name*). The set of paths that is removed is determined by the combination of the **-dev** *Name*, **-pdev** *Parent* and **-conn** *Connection* flags. If the command will result in all paths associated with the device being unconfigured or undefined, the command will exit with an error and without unconfiguring or undefining any path. In this situation, **rmdev** command must be used instead to unconfigure or undefine the target device itself.

The default action unconfigures each specified path, but does not completely remove it from the system. If the **-rm** flag is specified, the **rmpath** command unconfigures (if necessary) and removes, or deletes, the path definition(s) from the system.

When the **rmpath** command finishes, it displays a status message. When unconfiguring paths, it is possible for this command to be able to unconfigure some paths and not others (e.g., paths that are in the process of doing I/O cannot be unconfigured).

The **rmpath** command provides status messages about the results of operation. Messages in one of the following formats will be generated:

path defined | deleted
: This message is displayed when a single path was successfully unconfigured or undefined. If the path is successfully configured the message path available displays. If the path is not successfully configured and there is no explicit error code returned by the method, the message path defined displays.

paths defined | deleted
: This message is displayed if multiple paths were identified and all paths were successfully unconfigured or undefined. If the **-rm** flag is not specified, the message would be paths defined. If the **-rm** flag is specified, the message would be `paths deleted`.

some paths defined | deleted
: This message is display if multiple paths were identified, but only some of them were successfully unconfigured or undefined. If the **-rm** flag is not specified, the message would be some paths defined. If the **-rm** flag is specified, the message would be some paths deleted.

no paths processed
: This message is generated if no paths were found matching the selection criteria.

Flags

-rm	Indicates that the specified paths are to be deleted from the system.
-dev *Name*	Specifies the logical device name of the target device whose path is to be removed. The paths to be removed are qualified via the **-pdev** and **-conn** flags.
-pdev *Parent*	Indicates the logical device name of the parent device to use in qualifying the paths to be removed. Since all paths to a device cannot be removed by this command, either this flag, the **-conn** flag, or both must be specified.
-conn *Connection*	Indicates the connection information to use in qualifying the paths to be removed. Since all paths to a device cannot be removed by this command, either this flag, the **-pdev** flag, or both must be specified.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. 1. To unconfigure the path from **scsi0** to **hdisk1** at connection 5,0, type:

   ```
   rmpath -dev hdisk1 -pdev scsi0 -conn "5,0"
   ```

   The message generated would be similar to:

   ```
   path defined
   ```
2. To unconfigure all paths from **scsi0** to **hdisk1**, type:

   ```
   rmpath -dev hdisk1 -pdev scsi0
   ```

   If all paths were successfully unconfigured, the message generated would be similar to:

   ```
   paths defined
   ```

   However, if only some of the paths were successfully unconfigured, the message would be similar to:

   ```
   some paths defined
   ```
3. To undefine the path definition between **scsi0** and **hdisk1** at connection 5,0, type:

   ```
   rmpath -rm -dev hdisk1 -pdev scsi0 -conn "5,0"
   ```

   The message generated would be similar to the following:

   ```
   path deleted
   ```
4. To unconfigure all paths from **scsi0** to **hdisk1**, type:

   ```
   rmpath -rm -dev hdisk1 -pdev scsi0
   ```

   The message generated would be similar to:

   ```
   paths deleted
   ```

Related Information

The **chpath** command, the **lspath** command, and the **rmpath** command.

IVM rmsyscfg Command

Purpose

Removes a logical partition from the managed system. This command is valid only in an Integrated Virtualization Manager environment.

Syntax

rmsyscfg -r *lpar* { **-n** *ParitionName* | **--id** *PartitionID* }

Description

The **rmsyscfg** command removes a logical partition from the managed system.

Flags

-r *ResourceType* The type of resources to remove: **lpar** - Logical partition resources

-m *ManagedSystem* The name of the managed system. This attribute is optional because there is only one system to manage. The name may either be the user-defined name for the managed system, or be in the form tttt-mmm*ssssssss, where tttt is the machine type, mmm is the model, and ssssssss is the serial number of the managed system.

-n *PartitionName* The name of the partition which you want to remove.

--id *PartitionID* The partition ID of the partition to remove.

Exit Status

This command has a return code of zero on success.

Security

This command is not accessible by users with the ViewOnly role.

Examples

1. To delete a partition with an lpar ID of 3, type:

   ```
   rmsyscfg -r lpar --id 3
   ```
2. To delete a partition a name of lp3, type:

   ```
   rmsyscfg -r lpar -n lp3
   ```

Related Information

The **lssyscfg** command, the **mksyscfg** command, and the **chsyscfg** command.

rmtcpip Command

Puropse

Removes the Virtual I/O Server TCP/IP configuration.

Syntax

rmtcpip **-f** **-nextboot** {**-all** | **-hostname** **-routing** **-interface** *ifnameList*}

rmtcpip **-f** {**-all** | **-namesrv** **-hostname** **-routing** **-interface** *ifnameList*}

Description

The **rmtcpip** command removes Virtual I/O Server TCP/IP settings, such as name server information, network interface configuration, routing information, and host name.

Flags

-all	Removes all TCP/IP settings, effectively resetting it to a newly installed system.
-f	Executes the command with out prompting for user confirmation.
-interface	Removes TCP/IP configuration from listed interfaces.
-hostname	Resets the hostname to `ioserver`.
-namesrv	Removes DNS information and clears the hosts file.
-nextboot	Removes the specified information from the configuration files, leaving the current network parameters intact (all except DNS information and hosts file).
-routing	Removes static routing tables.

Examples

1. To remove all Virtual I/O Server TCP/IP configuration, type:

   ```
   rmtcpip -all
   ```

   Answer `yes` when prompted
2. To unconfigure a network interface en0 without confirmation, type:

   ```
   rmtcpip -f -interface en0
   ```
3. To cleanup the static routing table, type:

   ```
   rmtcpip -f -routing
   ```
4. To remove IP information from a network interface on the next boot, keeping the current configuration running execute:

   ```
   rmtcpip -f -interface en0 -nextboot
   ```

rmuser Command

Purpose

Removes a user account.

Syntax

rmuser **-rmdir** *Name*

Description

The **rmuser** command removes the user account identified by the *Name* parameter. This command removes a user's attributes without removing the user's home directory and files unless the **rmdir** flag is specified.

Flags

-rmdir	Removes the specified user's home directory.

Exit Status

See Virtual I/O Server command exit status.

Security

This command can only be executed by the prime administrator (padmin) user.

Examples

1. To remove user account **haarlem**, type:

   ```
   rmuser haarlem
   ```
2. To remove the user account and home directory of user account **emmen**, type:

   ```
   rmuser -rmdir emmen
   ```

   Attention: This will delete all data stored in this user account's home directory.

Related Information

The **chuser** command, the **lsuser** command, the **mkuser** command, and the **passwd** command.

rmvdev Command

Purpose

To remove the connection between a physical device and its associated virtual SCSI adapter.

Syntax

rmvdev **-f** { **-vdev** *TargetDevice* | **-vtd** *VirtualTargetDevice* } **-rmlv**

Description

The **rmdev** command removes the connection between a physical device and its associated virtual SCSI adapter. The connection can be identified by specifying the backing (physical) device or the virtual target device. If the connection is specified by the device name and there are multiple connections between the physical device and virtual SCSI adapters and error is returned unless the **-f** flag is also specified. If **-f** is included then all connections associated with the physical device are removed.

If the backing (physical) device is a logical volume and the **-rmlv** flag is specified, then logical volume will be removed as well.

Flags

-f	Forces the removal of all virtual target devices associated with the give backing device.
-vdev *TargetDevice*	Specifies the physical backing device
-rmlv	Deletes the backing device. This flag is valid only for logical volume backing devices.
-vtd *VirtualTargetDevice*	Specifies the virtual target device to remove.

IVM rmvt Command

Purpose

Closes a virtual terminal connection to a partition. This command is valid only in an Integrated Virtualization Manager environment.

Syntax

rmvt { **-id** *lparID* }

Description

The **rmvt** command closes a virtual terminal connection to the target logical partition. To close the virtual terminal session normally, enter the ~. character sequence in the terminal window.

This command requires additional HMC configuration if used in an HMC environment.

Flags

-id *lparID*	The ID of the partition for which to close the virtual terminal session.

Exit Status

29	Virtual terminal device is not found

Security

This command is not accessible by users with the ViewOnly role.

Examples

1. To close a virtual terminal connection to the partition with ID 3, type:

   ```
   rmvt -id 3
   ```

Related Information

The **mkvt** command.

IVM rstprofdata Command

Purpose

Restores profile data. This command is valid only in an Integrated Virtualization Manager environment.

Syntax

To restore logical partition configuration data from a file:

rstprofdata -l *RestoreType* **-f** *RestoreFile* **--ignoremtms --ignoremac -m** *ManagedSystem*

Description

The **rstprofdata** command performs a restore of logical partition configuration information from a file which was created with the **bkprofdata** command. Logical partition configuration must be cleared using the **lpcfgop** command before performing a restore operation. The restore operation may require a restart of the managed system, after which the **rstprofdata** command should be run again with the same parameters as were used the first time (before the system restart).

Flags

-l *RestoreType*	The type of restore to perform. Valid options are: **1** - full restore from the backup file
-f *RestoreFile*	The name of the file to read from in the current working directory. If not specified, the default file will be "/var/adm/lpm/profile.bak"
--ignoremtms	Do not fail the restore if the type, model, or serial number of the managed system does not match the values in the backup file.
--ignoremac	Do not try to restore the virtual Ethernet MAC addresses from the backup file. Default to the automatically generated MAC addresses.
-m *ManagedSystem*	The name of the managed system. This attribute is optional because there is only one system to manage. The name may either be the user-defined name for the managed system, or be in the form tttt-mmm*ssssssss, where tttt is the machine type, mmm is the model, and ssssssss is the serial number of the managed system.
--help	Display the help text for this command and exit.

Exit Status

This command has a return code of zero on success.

Security

This command is not accessible by users with the ViewOnly role.

Examples

1. To restore the partition configuration data from **/var/adm/lpm/profile.bak**, type:

   ```
   rstprofdata -l 1
   ```
2. To restore the partition configuration data from **lparData.bak** without validating that the type, model, and serial number match, type:

   ```
   rstprofdata -l 1 -f lparData.bak --ignoremtms
   ```

Related Information

The **bkprofdata** command.

savevgstruct Command

Purpose

Backs up a volume group.

Syntax

savevgstruct *VolumeGroupLabel*

Description

The **savevgstruct** command will make a backup of a volume group structure.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To save the structure of the user defined volume group **myvg**, enter:

   ```
   savevgstruct myvg
   ```

Related Information

The **activatevg** command, the **restorevgstruct** command, the **chvg** command, the **deactivatevg** command, the **exportvg** command, the **extendvg** command, the **importvg** command, the **lsvg** command, the **mkvg** command, and the **syncvg** command.

sed Command

Purpose

A stream editor.

Syntax

sed **-n** *Script* *File* ...

sed **-n** **-e** *Script* ... **-f** *ScriptFile* ... *File* ...

Description

The **sed** command modifies lines from the specified *File* parameter according to an edit script and writes them to standard output. The **sed** command includes many features for selecting lines to be modified and making changes only to the selected lines.

The **sed** command uses two work spaces for holding the line being modified: the pattern space, where the selected line is held; and the hold space, where a line can be stored temporarily.

An edit script consists of individual subcommands, each one on a separate line. The general form of **sed** subcommands is the following:

address-range functionmodifiers

The **sed** command processes each input *File* parameter by reading an input line into a pattern space, applying all **sed** subcommands in sequence whose addresses select that line, and writing the pattern space to standard output. It then clears the pattern space and repeats this process for each line specified in the input *File* parameter. Some of the **sed** subcommands use a hold space to save all or part of the pattern space for subsequent retrieval.

When a command includes an address (either a line number or a search pattern), only the addressed line or lines are affected by the command. Otherwise, the command is applied to all lines.

An address is either a decimal line number, a $ (dollar sign), which addresses the last line of input, or a context address. A context address is a regular expression similar to those used in the **ed** command except for the following differences:

- You can select the character delimiter for patterns. The general form of the expression is:

  ```
  \?pattern?
  ```

  where ? (question mark) is a selectable character delimiter. You can select any character from the current locale except for the space or new-line character. The \ (backslash) character is required only for the first occurrence of the ? (question mark).

  The default form for the pattern is the following:

  ```
  /pattern/
  ```

  A \ (backslash) character is not necessary.
- The **\n** sequence matches a new-line character in the pattern space, except the terminating new-line character.

- A **.** (period) matches any character except a terminating new-line character. That is, unlike the **ed** command, which cannot match a new-line character in the middle of a line, the **sed** command can match a new-line character in the pattern space.

Certain commands called *addressed* commands allow you to specify one line or a range of lines to which the command should be applied. The following rules apply to addressed commands:

- A command line without an address selects every line.
- A command line with one address, expressed in context form, selects each line that matches the address.
- A command line with two addresses separated by commas selects the entire range from the first line that matches the first address through the next line that matches the second. (If the second address is a number less than or equal to the line number first selected, only one line is selected.) Thereafter, the process is repeated, looking again for the first address.

Flags

-e *Script*	Uses the *Script* variable as the editing script. If you are using just one **-e** flag and no **-f** flag, the **-e** flag can be omitted.
-f *ScriptFile*	Uses the *ScriptFile* variable as the source of the edit script. The *ScriptFile* variable is a prepared set of editing commands applied to the *File* parameter.
-n	Suppresses all information normally written to standard output.

Note: You can specify multiple **-e** and **-f** flags. All subcommands are added to the script in the order specified, regardless of their origin.

sed Subcommands

The **sed** command contains the following **sed** script subcommands. The number in parentheses preceding a subcommand indicates the maximum number of permissible addresses for the subcommand.

Notes:

1. The *Text* variable accompanying the **a**\, **c**\, and **i**\ subcommands can continue onto more than one line, provided all lines but the last end with a \ (backslash) to quote the new-line character. Backslashes in text are treated like backslashes in the replacement string of an **s** command and can be used to protect initial blanks and tabs against the stripping that is done on every script line. The *RFile* and *WFile* variables must end the command line and must be preceded by exactly one blank. Each *WFile* variable is created before processing begins.
2. The **sed** command can process up to 999 subcommands in a pattern file.

Subcommand	Description
(1) **a**\ *Text*	Places the *Text* variable in output before reading the next input line.
(2)**b***label*	Branches to the **:** command bearing the *label* variable. If the *label* variable is empty, it branches to the end of the script.
(2)**c**\ *Text*	Deletes the pattern space. With 0 or 1 address or at the end of a 2-address range, places the *Text* variable in output and then starts the next cycle.
(2)**d**	Deletes the pattern space and then starts the next cycle.
(2)**D**	Deletes the initial segment of the pattern space through the first new-line character and then starts the next cycle.
(2)**g**	Replaces the contents of the pattern space with the contents of the hold space.
(2)**G**	Appends the contents of the hold space to the pattern space.
(2)**h**	Replaces the contents of the hold space with the contents of the pattern space.
(2)**H**	Appends the contents of the pattern space to the hold space.
(1)**i**\ *Text*	Writes the *Text* variable to standard output before reading the next line into the pattern space.

(2)**l**
Writes the pattern space to standard output showing nondisplayable characters as 4-digit hexadecimal values. Long lines are folded.

(2)**l**
Writes the pattern space to standard output in a visually unambiguous form. The characters \\\, \\a, \\b, \\f, \\r, \\t, and \\v are written as the corresponding escape sequence. Non-printable characters are written as 1 three-digit octal number (with a preceding backslash character) for each byte in the character (most significant byte first). This format is also used for multibyte characters. This subcommand folds long lines. A backslash followed by a new-line character indicates the point of folding. Folding occurs at the 72nd column position. A $ (dollar sign) marks the end of each line.

(2)**n**
Writes the pattern space to standard output if the default output is not suppressed. It replaces the pattern space with the next line of input.

(2)**N**
Appends the next line of input to the pattern space with an embedded new-line character (the current line number changes). You can use this to search for patterns that are split onto two lines.

(2)**p**
Writes the pattern space to standard output.

(2)**P**
Writes the initial segment of the pattern space through the first new-line character to standard output.

(1)**q**
Branches to the end of the script. It does not start a new cycle.

(2)**r** *RFile*
Reads the contents of the *RFile* variable. It places contents in output before reading the next input line.

(2)**s**/*pattern*/*replacement*/*flags*
Substitutes the *replacement* string for the first occurrence of the *pattern* parameter in the pattern space. Any character that is displayed after the **s** subcommand can substitute for the / (slash) separator except for the space or new-line character.

The value of the *flags* variable must be zero or more of:

g
Substitutes all non-overlapping instances of the *pattern* parameter rather than just the first one.

n
Substitutes for the *n-th* occurrence only of the *pattern* parameter.

p
Writes the pattern space to standard output if a replacement was made.

w *WFile*
Writes the pattern space to the *WFile* variable if a replacement was made. Appends the pattern space to the *WFile* variable. If the *WFile* variable was not already created by a previous write by this **sed** script, the **sed** command creates it.

(2)**t***label*
Branches to the **:***label* variable in the script file if any substitutions were made since the most recent reading of an input line execution of a **t** subcommand. If you do not specify the *label* variable, control transfers to the end of the script.

(2)**w***WFile*
Appends the pattern space to the *WFile* variable.

(2)**x**
Exchanges the contents of the pattern space and the hold space.

(2)**y**/*pattern1*/*pattern2*/
Replaces all occurrences of characters in the *pattern1* variable with the corresponding *pattern2* characters. The number of characters in the *pattern1* and *pattern2* variables must be equal. The new-line character is represented by **n**.

(2)**!***sed-cmd*
Applies the specified **sed** subcommand only to lines not selected by the address or addresses.

(0)**:***label*
Marks a branch point to be referenced by the **b** and **t** subcommands. This label can be any sequence of eight or fewer bytes.

(1)**=**
Writes the current line number to standard output as a line.

(2)**{***subcmd* **}**
Groups subcommands enclosed in {} (braces).

(0)
Ignores an empty command.

(0)**#**
If a # (pound sign) appears as the first character on a line of a script file, that entire line is treated as a comment, with one exception. For the first line of a script file only, if the character after the # is an n, the default output will be suppressed. The rest of the line after the #n is ignored.

Exit Status

This command returns the following exit values:

0	Successful completion.
>0	An error occurred.

Examples

1. To perform a global change, enter:

   ```
   sed  "s/happy/enchanted/g" chap1
   ```

   This command sequence replaces each occurrence of the word `happy` found in the file `chap1` with the word `enchanted`. The **g** character at the end of the **s** subcommand tells the **sed** command to make as many substitutions as possible on each line. Without the **g** character, the **sed** command replaces only the first occurrence of the word `happy` on a line.

   The **sed** command operates as a filter. It reads text from standard input or from the files named on the command line (`chap1` in this example), modifies this text, and writes it to standard output. Unlike most editors, it does not replace the original file. This makes the **sed** command a powerful command when used in pipelines.
2. To use the **sed** command as a filter in a pipeline, enter:

   ```
   pr  chap2 | sed "s/Page *0-9*$/(&)/" | enq
   ```

   This command sequence encloses the page numbers in parentheses before printing the file `chap2`. The **pr** command puts a heading and page number at the top of each page, then the **sed** command puts the page numbers in parentheses, and the **enq** command prints the edited listing.

   The **sed** command pattern `/Page *0-9*$/` matches page numbers that appear at the end of a line. The **s** subcommand changes this to `(&)`, where the `&` stands for the page number that was matched.
3. To display selected lines of a file, enter:

   ```
   sed  -n  "/food/p" chap3
   ```

   The `sed -n` displays each line in the file `chap3` that contains the word `food`. Normally, the **sed** command copies every line to standard output after it is edited. The **-n** flag stops the **sed** command from doing this. You then use subcommands like **p** to write specific parts of the text. Without the **-n** flag, this example displays all the lines in the file `chap3`, and it shows each line containing `food` twice.
4. To perform complex editing, enter:

   ```
   sed  -f  script.sed  chap4
   ```

   This command sequence creates a **sed** script file when you want to do anything complex. You can then test and modify your script before using it. You can also reuse your script to edit other files. Create the script file with an interactive text editor.
5. A sample **sed** script file:

   ```
   :join
   /\\$/{N
   s/\\\n//
   b join
   }
   ```

   This **sed** script joins each line that ends with a \ (backslash) to the line that follows it. First, the pattern `/\\$/` selects a line that ends with a \ for the group of commands enclosed in `{}` (braces). The **N** subcommand then appends the next line, embedding a new-line character. The `s/\\\n//` deletes the \ and embedded new-line character. Finally, `b join` branches back to the label `:join` to check for a `\` at the end of the newly joined line. Without the branch, the **sed** command writes the joined line and reads the next one before checking for a second `\`.

Note: The **N** subcommand causes the **sed** command to stop immediately if there are no more lines of input (that is, if the **N** subcommand reads an end-of-file character). It does not copy the pattern space to standard output before stopping. This means that if the last line of the input ends with a \, it is not copied to the output.

6. To copy an existing file (`oldfile`) to a new file (`newfile`) and replace all occurrences of the `testpattern` text string with the contents of the `$REPL` shell variable, enter:

   ```
   cat oldfile | sed -e "s/testpattern/$REPL/g" | tee newfile
   ```
7. To replace all occurrences of A with a, B with b, C with c, and all occurrences of newlines with character Z in the input file, enter:

   ```
   $ sed -f command.file input.file
   ```

   where *command.file* is the script file and *input.file* is the input file.

   ```
      $cat command.file
      y/ABC\n/abcZ/
   ```

   Alternatively, the following command can also be executed for the same function:

   ```
   sed "y/ABC\n/abcZ/" input.file
   ```

Related Information

The **awk** command and the **grep** command.

showmount Command

Purpose

Displays a list of exported directories.

Syntax

showmount *Host*

Description

The **showmount** command displays a list of all exported directories from a specified machine in the *Host* parameter.

Parameters

Host	Host name of the systems to display exported directories.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To display all exported directories on the host **middelburg**, type:

   ```
   showmount middelburg
   ```

Related Information

The **mount** command and the **unmount** command.

shutdown Command

Purpose

Ends system operation.

Syntax

shutdown -restart

Description

The **shutdown** command halts the operating system. When the shutdown is complete the user receives a shutdown completion message. Do not attempt to restart the system or turn off the system before the shutdown completion message is displayed; otherwise, file system damage can result.

Flags

-restart	Restarts the system after being shutdown.

Exit Status

See Virtual I/O Server command exit status.

Security

This command can only be executed by the prime administrator (padmin).

Examples

1. To shut down the system, type:

   ```
   shutdown
   ```
2. To restart the system, type:

   ```
   shutdown -restart
   ```

snap Command

Purpose

Gathers system configuration information.

Syntax

snap **-general** **-dev** *DeviceName*

snap *script1* "*script2 arg1 arg2*" ...

Description

The **snap** command gathers system configuration information and compresses the information into a pax file (snap.pax.Z). The file can then be transmitted to a remote system. The information gathered with the **snap** command may be required to identify and resolve system problems.

Approximately 8MB of temporary disk space is required to collect all system information, including contents of the error log.

Flags

-dev *OutputDevice*	Copies the compressed image onto the specified device.
-general	Gathers general system information which is a subset of the system configuration information

Exit Status

See Virtual I/O Server command exit status.

Examples

1. Enter the following command to gather all system configuration information:

   ```
   snap
   ```

   The output of this command is written to the users home directory.
2. Enter the following command to gather general system configuration information, including the output of the **lslpp -hBc** command:

   ```
   snap -general -dev /dev/rfd0
   ```

   Output is written to the **/tmp/ibmsupt/general/lslpp.hBc** and **/tmp/ibmsupt/general/general.snap** files. This command also writes the system information to a removable diskette.
3. To run the scripts foo1, foo2 and foo3. where foo1 takes no argument, foo2 takes three arguments and foo3 takes one argument, type the following"

   ```
   snap  foo1 "foo2 -x -y 3" "foo3 6" foo4
   ```

   Output is written to **/tmp/ibmsupt/snapscripts/foo1**, **/tmp/ibmsupt/snapscripts/foo2** and **/tmp/ibmsupt/snapscripts/foo3** assuming the destination directory is the default, **/tmp/ibmsupt**.

Files

snap.pax.Z

startnetsvc Command

Purpose

Starts the **telnet** and **ftp** daemons.

Syntax

startnetsvc *NetworkService*

Description

The **startnetsvc** command can start the **telnet** and **ftp** daemons. Enabling the **telnet** daemon allows users to **telnet** into the Virtual I/O Server. Enabling the **ftp** daemon allows users to **ftp** into the Virtual I/O Server.

Parameters

NetworkService

The following values may be used:

telnet

Enables the **telnet** daemon

ftp

Enables to **ftp** daemon

ALL

Enables both the **telnet** and **ftp** daemons.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To start the **telnet** daemon, type:

   ```
   startnetsvc telnet
   ```
2. To start the **ftp** daemon, type:

   ```
   startnetsvc ftp
   ```
3. To start both the **ftp** daemon and the **telnet** daemon, type:

   ```
   startnetsvc ALL
   ```

Related Information

The **mktcpip** command, the **hostname** command, the **stopnetsvc** command, the **cfglnagg** command, the **netstat** command, the **entstat** command, the **cfgnamesrv** command, the **hostmap** command, the **traceroute** command, the **ping** command, the **optimizenet** command.

startsysdump Command

Purpose

Starts a kernel dump to the primary dump device.

Syntax

startsysdump

Description

The **startsysdump** command provides a command line interface to start a kernel dump to the primary dump device. Any previous kernel dumps will be erased before the dump is created. During a kernel dump, the following values can be displayed on the three-digit terminal display as follows. The user will be required the run the **snap** command to obtain the system dump.

0c0	Indicates that the dump completed successfully.
0c1	Indicates that an I/O occurred during the dump.
0c2	Indicates that the dump is in progress.
0c4	Indicates that the dump is too small.
0c5	Indicates a dump internal error.
0c6	Prompts you to make the secondary dump device ready.
0c7	Indicates that the dump process is waiting for a response from the remote host.
0c8	Indicates that the dump was disabled. In this case, no dump device was designated in the system configuration object for dump devices. The **startsysdump** command halts, and the system continues running.
0c9	Indicates that a dump is in progress.
0cc	Indicates that the system switched to the secondary dump device after attempting a dump to the primary device.

Note: When the dump completes, the system reboots.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To start a kernel dump, type:

```
startsysdump
```

starttrace Command

Purpose

Records selected system events.

Syntax

starttrace -event *Event*, *Event* ...

Description

The **starttrace** command starts the **trace** daemon which configures a trace session and starts the collection of system events. The data collected by the trace function is recorded in the trace log. A report from the trace log can be generated with the **cattracerpt** command.

Flags

-event *Event*,*Event*

Specifies the user-defined events for which you want to collect trace data. The Event list items should be separated by commas. **Note:** The following events are used to determine the pid, the cpuid and the exec path name in the **cattracerpt** report:

- 106 DISPATCH
- 10C DISPATCH IDLE PROCESS
- 134 EXEC SYSTEM CALL
- 139 FORK SYSTEM CALL
- 465 KTHREAD CREATE

If any of these events is missing, the information reported by the **cattracerpt** command will be incomplete. When using the **-event** flag, you should include all these events in the *Event* list.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To trace hook 234 and the hooks that will allow you to see the process names, enter:

```
starttrace -event 234,106,10C,134,139,465
```

Related Information

The **stoptrace** command and the **cattracerpt** command.

stopnetsvc Command

Purpose

Disables the **telnet** and **ftp** daemons.

Syntax

stopnetsvc *NetworkService*

Description

The **stopnetsvc** command can stop the **telnet** and **ftp** daemons. Disabling the **telnet** daemon prevents anyone from being able to **telnet** into the Virtual I/O Server. Disabling the **ftp** daemon prevents anyone from being able to **ftp** into the Virtual I/O Server.

Parameters

NetworkService	The following values may be used: telnet Disables the **telnet** daemon ftp Disables the **ftp** daemon ALL Disables both the **telnet** daemon and the **ftp** daemon

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To disable the **telnet** daemon, type:

   ```
   stopnetsvc telnet
   ```
2. To disable the **ftp** daemon, type:

   ```
   stopnetsvc ftp
   ```
3. To disable both the **telnet** daemon and the **ftp** daemon, type:

   ```
   stopnetsvc ALL
   ```

Related Information

The **mktcpip** command, the **hostname** command, the **startnetsvc** command, the **cfglnagg** command, the **netstat** command, the **entstat** command, the **cfgnamesrv** command, the **hostmap** command, the **traceroute** command, the **ping** command, the **optimizenet** command.

stoptrace Command

Purpose

Stops the trace function.

Syntax

stoptrace

Description

The **stoptrace** command ends a trace session.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To end a trace session, type:

```
stoptrace
```

Related Information

The **starttrace** command and the **cattracerpt** command.

stty Command

Purpose

Sets, resets, and reports workstation operating parameters.

Syntax

stty **-a** **-g** *Options*

Description

The **stty** command sets certain I/O options for the device that is the current standard input. This command writes output to the device that is the current standard output.

This version of the operating system uses the standard interface to control the terminals, maintaining a compatibility with POSIX and BSD interfaces. The **stty** command supports both POSIX and BSD compliant options, but the usage of POSIX options is strongly recommended. A list of obsolete BSD options, with the corresponding POSIX options, is also provided.

When you redirect standard input from a tty device by typing:

```
stty -a </dev/ttyx
```

the **stty** command (POSIX) will hang while waiting for the **open()** of that tty until the RS-232 carrier detect signal has been asserted. Exceptions to this rule occur if the **clocal** or **forcedcd** (128-port only) option is set.

Flags

-a	Writes the current state of all option settings to standard output.
-g	Writes option settings to standard output in a form usable by another **stty** command.

Options

The **stty** command supports following categories of options:

- Control modes
- Input modes
- Output modes
- Local modes
- Hardware flow control modes
- Control assignments
- Combination modes
- Window size
- Obsolete options

Control modes

Control mode	Description
clocal	Assumes a line without modem control.
-clocal	Assumes a line with modem control.
cread	Enables the receiver.
-cread	Disables the receiver.

cstopb	Selects 2 stop bits per character.
-cstopb	Selects 1 stop bit per character.
cs5, **cs6**, **cs7**, **cs8**	Selects character size.
hup, **hupcl**	Hangs up dial-up connection on the last close.
-hup, **-hupcl**	Does not hang up dial-up connection on the last close.
parenb	Enables parity generation and detection.
-parenb	Disables parity generation and detection.
parodd	Selects odd parity.
-parodd	Selects even parity.
0	Hangs up phone line immediately.
speed	Sets the workstation input and output speeds to the specified *speed* number of bits per second. All speeds are not supported by all hardware interfaces. Possible values for *speed* are: **50**, **75**, **110**, **134**, **200**, **300**, **600**, **1200**, **1800**, **2400**, **4800**, **9600**, **19200**, **19.2**, **38400**, **38.4**, **exta**, and **extb**. **Note: exta**, **19200**, and **19.2** are synonyms; **extb**, **38400**, and **38.4** are synonyms.
ispeed *speed*	Sets the workstation input speed to the specified *speed* number of bits per second. All speeds are not supported by all hardware interfaces, and all hardware interfaces do not support this option. Possible values for *speed* are the same as for the *speed* option.
ospeed *speed*	Sets the workstation output speed to the specified *speed* number of bits per second. All speeds are not supported by all hardware interfaces, and all hardware interfaces do not support this option. Possible values for *speed* are the same as for the *speed* option.

Input modes

Input mode	Description
brkint	Signals INTR on break.
-brkint	Does not signal INTR on break.
icrnl	Maps CR to NL on input.
-icrnl	Does not map CR to NL on input.
ignbrk	Ignores BREAK on input.
-ignbrk	Does not ignore BREAK on input.
igncr	Ignores CR on input.
-igncr	Does not ignore CR on input.
ignpar	Ignores parity errors.
-ignpar	Does not ignore parity errors.
inlcr	Maps NL to CR on input.
-inlcr	Does not map NL to CR on input.
inpck	Enables parity checking.
-inpck	Disables parity checking.
istrip	Strips input characters to 7 bits.
-istrip	Does not strip input characters to 7 bits.
iuclc	Maps uppercase alphabetic characters to lowercase.
-iuclc	Does not map uppercase alphabetic characters to lowercase.
ixany	Allows any character to restart output.
-ixany	Allows only the START (the Ctrl-Q key sequence) to restart output.
ixoff	Sends START/STOP characters when the input queue is nearly empty/full.
-ixoff	Does not send START/STOP characters.
ixon	Enables START/STOP output control. Once START/STOP output control has been enabled, you can pause output to the workstation by pressing the Ctrl-S key sequence and resume output by pressing the Ctrl-Q key sequence.
-ixon	Disables START/STOP output control.
imaxbel	Echoes the BEL character and discards the last input character if input overflows.
-imaxbel	Discards all input if input overflows.
parmrk	Marks parity errors.
-parmrk	Does not mark parity errors.

Output modes

Output mode	Description
bs0, **bs1**	Selects style of delay for backspaces (**bs0** siginifes no delay).
cr0, **cr1**, **cr2**, **cr3**	Selects style of delay for CR characters (**cr0** siginifes no delay).
ff0, **ff1**	Selects style of delay for form feeds (**ff0** siginifes no delay).
nl0, **nl1**	Selects style of delay for NL characters (**nl0** siginifes no delay).
ofill	Uses fill characters for delays.
-ofill	Uses timing for delays.
ocrnl	Maps CR characters to NL characters.
-ocrnl	Does not map CR characters to NL characters.
olcuc	Maps lowercase alphabetic characters to uppercase on output.
-olcuc	Does not map lowercase alphabetic characters to uppercase on output.
onlcr	Maps NL characters to CR-NL characters.
-onlcr	Does not map NL characters to CR-NL characters.
onlret	On the terminal, NL performs the CR function.
-onlret	On the terminal, NL does not perform the CR function.
onocr	Does not output CR characters at column zero.
-onocr	Outputs CR characters at column zero.
opost	Processes output.
-opost	Does not process output; that is, ignores all other output options.
ofdel	Uses DEL characters for fill characters.
-ofdel	Uses NUL characters for fill characters.
tab0, **tab1**, **tab2**	Selects style of delay for horizontal tabs (**tab0** siginifes no delay).
tab3	Expands tab character to variable number of spaces.
vt0, **vt1**	Selects style of delay for vertical tabs (**vt0** siginifes no delay).

Local modes

Local mode	Description
echo	Echoes every character typed.
-echo	Does not echo characters.
echoctl	Echoes control characters as ^X (Ctrl-X), where X is the character given by adding 100 octal to the code of the control character.
-echoctl	Does not echo control characters as ^X (Ctrl-X).
echoe	Echoes the ERASE character as the "backspace space backspace" string. **Note:** This mode does not keep track of column position, so you can get unexpected results when erasing such things as tabs and escape sequences.
-echoe	Does not echo the ERASE character, just backspace.
echok	Echoes a NL character after a KILL character.
-echok	Does not echo a NL character after a KILL character.
echoke	Echoes the KILL character by erasing each character on the output line.
-echoke	Just echoes the KILL character.
echonl	Echoes the NL character.
-echonl	Does not echo the NL character.
echoprt	Echoes erased characters backwards with / (slash) and \ (backslash).
-echoprt	Does not echo erased characters backwards with / (slash) and \ (backslash).
icanon	Enables canonical input (canonical input allows input-line editing with the ERASE and KILL characters).
-icanon	Disables canonical input.
iexten	Specifies that implementation-defined functions shall be recognized from the input data. Recognition of the following control characters requires **iexten** to be set: **eol2, dsusp, reprint, discard, werase, lnext.** The functions associated with these modes also require **iexten** to be set: **imaxbel**, **echoke**, **echoprt**, and **echoctl**.
-iexten	

	Specifies that implementation-defined functions shall not be recognized from the input data.
isig	Enables the checking of characters against the special control characters INTR, SUSP and QUIT.
-isig	Disables the checking of characters against the special control characters INTR, SUSP and QUIT.
noflsh	Does not clear buffers after INTR, SUSP, or QUIT control characters.
-noflsh	Clears buffers after INTR, SUSP, or QUIT control characters.
pending	Causes any input that is pending after a switch from raw to canonical mode to be re-input the next time a read operation becomes pending or the next time input arrives. Pending is an internal state bit.
-pending	No text is pending.
tostop	Signals **SIGTOU** for background output.
-tostop	Does not signal **SIGTOU** for background output.
xcase	Echoes uppercase characters on input, and displays uppercase characters on output with a preceding \ (backslash).
-xcase	Does not echo uppercase characters on input.

Hardware flow control modes

These options are extensions to the X/Open Portability Guide Issue 4 standard.

Extension	Description
cdxon	Enables CD hardware flow control mode on output.
-cdxon	Disables CD hardware flow control mode on output.
ctsxon	Enables CTS hardware flow control mode on output.
-ctsxon	Disables CTS hardware flow control mode on output.
dtrxoff	Enables DTR hardware flow control mode on input.
-dtrxoff	Disables DTR hardware flow control mode on input.
rtsxoff	Enables RTS hardware flow control mode on input.
-rtsxoff	Disables RTS hardware flow control mode on input.

Control assignments

To assign a control character to a character string, type:

```
stty Control String
```

where the *Control* parameter may be the INTR, QUIT, ERASE, KILL, EOF, EOL, EOL2, START, STOP, SUSP, DSUSP, REPRINT, DISCARD, WERASE, LNEXT, MIN, or TIME character. (Use the MIN and TIME characters with the **-icanon** option.)

Note: The values for MIN and TIME are interpreted as integer values, not as character values.

The *String* parameter may be any single character such as `c`. An example of this control assignment is:

```
stty STOP c
```

Another way of assigning control characters is to enter a character sequence composed of a \^ (backslash, caret) followed by a single character. If the single character after the ^ (caret) is one of the characters listed in the ^c (caret c) column of the following table, the corresponding control character value will be set. For example, to assign the DEL control character by using the ? (question mark) character, type the string `\^?` (backslash, caret, question mark), as in:

```
stty ERASE \^?
```

^c	Value
a, A	<SOH>
b, B	<STX>
c, C	<ETX>
d, D	<EOT>
e, E	<ENQ>

f, F	<ACK>
g, G	<BEL>
h, H	<BS>
i, I	<HT>
j, J	<LF>
k, K	<VT>
l, L	<FF>
m, M	<CR>
n, N	<SO>
o, O	<SI>
p, P	<DLE>
q, Q	<DC1>
r, R	<DC2>
s, S	<DC3>
t, T	<DC4>
u, U	<NAK>
v, V	<SYN>
w, W	<ETB>
x, X	<CAN>
y, Y	<EM>
z, Z	<SUB>
	<ESC>
\	<FS>
	<GS>
^	<RS>
_	<US>
?	<DEL>
@	<NUL>

Combination modes

Combination mode	Description
cooked	See the **-raw** option.
ek	Sets ERASE and KILL characters to the Ctrl-H and Ctrl-U key sequences, respectively.
evenp	Enables **parenb** and **cs7**.
-evenp	Disables **parenb** and sets **cs8**.
lcase, **LCASE**	Sets **xcase**, **iuclc**, and **olcuc**. Used for workstations with uppercase characters only.
-lcase, **-LCASE**	Sets **-xcase**, **-iuclc**, and **-olcuc**.
nl	Sets **-icrnl** and **-onlcr**.
-nl	Sets **icrnl**, **onlcr**, **-inlcr**, **-igncr**, **-ocrnl**, and **-onlret**.
oddp	Enables **parenb**, **cs7**, and **parodd**.
-oddp	Disables **parenb** and sets **cs8**.
parity	See the **evenp** option.
-parity	See the **-evenp** option.
sane	Resets parameters to reasonable values.
raw	Allows raw mode input (no input processing, such as erase, kill, or interrupt); parity bit passed back.
-raw	Allows canonical input mode.
tabs	Preserves tabs.
-tabs, **tab3**	Replaces tabs with spaces when printing.

Window size

Window size	Description
cols *n*, **columns** *n*	The terminal (window) size is recorded as having *n* columns.
rows *n*	The terminal (window) size is recorded as having *n* rows.
size	Prints the terminal (window) sizes to standard output (first rows and then columns).

Obsolete options

The following BSD options are supported by the **stty** command. For each of them, the recommended POSIX option is given.

Option	Description
all	Use the **stty** **-a** command to display all current settings.
crt	Use the **sane** option to reset parameters to reasonable values.
crtbs	Use the **-echoe** option.
crterase	Use the **echoe** option.
-crterase	Use the **-echoe** option.
crtkill	Use the **echoke** option.
-crtkill	Use the **echok** and **-echoke** options.
ctlecho	Use the **echoctl** option.
-ctlecho	Use the **-echoctl** option.
decctlq	Use the **-ixany** option.
-decctlq	Use the **ixany** option.
even	Use the **evenp** option.
-even	Use the **-evenp** option.
everything	Use the **stty** **-a** command to display all current settings.
litout	Use the **-opost** option.
-litout	Use the **opost** option.
odd	Use the **oddp** option.
-odd	Use the **-oddp** option.
pass8	Use the **-istrip** option.
-pass8	Use the **istrip** option.
prterase	Use the **echoprt** option.
speed	Use the **stty** command to display current settings.
tandem	Use the **ixoff** option.
-tandem	Use the **-ixoff** option.

Examples

1. To display a short listing of your workstation configuration, type:

   ```
   stty
   ```

   This lists settings that differ from the defaults.
2. To display a full listing of your workstation configuration, type: `stty  -a`
3. To enable a key sequence that stops listings from scrolling off the screen, type:

   ```
   stty ixon ixany
   ```

   This sets **ixon** mode, which lets you stop runaway listing by pressing the Ctrl-S key sequence. The **ixany** flag allows you to resume the listing by pressing any key. The normal workstation configuration includes the **ixon** and **ixany** flags, which allows you to stop a listing with the Ctrl-S key sequence that only the Ctrl-Q key sequence will restart.
4. To reset the configuration after it has been messed up, type: `Ctrl-J stty  sane Ctrl-J` Press the Ctrl-J key sequence before and after the command instead of the Enter key. The system usually recognizes the Ctrl-J key sequence when the parameters that control Enter key processing are messed up.

   Sometimes the information displayed on the screen may look strange, or the system will not respond when you press the Enter key. This can happen when you use the **stty** command with parameters that are incompatible or that do things you don't understand. It can also happen when a screen-oriented application ends abnormally and does not have a chance to reset the workstation configuration.

Entering the **stty sane** command sets a reasonable configuration, but it may differ slightly from your normal configuration.

5. To save and restore the terminal's configuration:

```
OLDCONFIG=`stty -g`          # save configuration
stty -echo                   # do not display password
echo "Enter password: \c"
read PASSWD                  # get the password
stty $OLDCONFIG              # restore configuration
```

This command saves the workstation's configuration, turns off echoing, reads a password, and restores the original configuration.

Entering the **stty -echo** command turns off echoing, which means that the password does not appear on the screen when you type it at the keyboard. This action has nothing to do with the **echo** command, which displays a message on the screen.

File

/**usr**/**bin**/**stty**	Contains the **stty** command.

syncvg Command

Purpose

Synchronizes logical volume copies that are not current.

Syntax

syncvg { **-lv** | **-pv** | **-vg** } *Name* ...

Description

The **syncvg** command synchronizes the physical partitions, which are copies of the original physical partition, that are not current. The **syncvg** command can be used with logical volumes, physical volumes, or volume groups, with the *Name* parameter representing the logical volume name, physical volume name, or volume group name.

Unless disabled, the copies within a volume group are synchronized automatically when the volume group is activated by the **activatevg** command.

Note: For the **sycnvg** command to be successful, at least one good copy of the logical volume should be accessible, and the physical volumes that contains this copy should be in ACTIVE state.

The **syncvg** command will check for the *NUM_PARALLEL_LPS* environment variable. The value of *NUM_PARALLEL_LPS* will be used to set the number of logical partitions to be synchronized in parallel.

Flags

-lv	Specifies that the *Name* parameter represents a logical volume device name.
-pv	Specifies that the *Name* parameter represents a physical volume device name.
-vg	Specifies that the *Name* parameter represents a volume group device name.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To synchronize the copies on physical volumes **hdisk04** and **hdisk05**, type:

   ```
   syncvg -pv hdisk04 hdisk05
   ```
2. To synchronize the copies on volume groups **vg04** and **vg05**, type:

   ```
   syncvg -vg vg04 vg05
   ```

Related Information

The **mkvg** command, the **chvg** command, the **extendvg** command, the **lsvg** command, the **mirrorios** command, the **unmirrorios** command, the **activatevg** command, the **deactivatevg** command, the **importvg** command, the **exportvg** command, and the **reducevg** command.

sysstat Command

Purpose

Displays a summary of current system activity.

Syntax

sysstat **-long** | **-short** *User*

Description

The **sysstat** command provides the following information: time of day, amount of time since last system startup, number of users logged in and number of processes running.

Flags

-long	Prints the summary in long form. This is the default
-short	Prints the time of day, amount of time since last system startup, number of Users logged in and number of processes running.

Parameters

User	Specify an existing user name.

tail Command

Purpose

Displays the last few lines of a file.

Syntax

Standard Syntax

tail **-f** **-c** *Number* | **-n** *Number* | **-m** *Number* | **-b** *Number* | **-k** *Number* *File*

To Display Lines in Reverse Order

tail **-r** **-n** *Number* *File*

Description

The **tail** command writes the file specified by the *File* parameter to standard output beginning at a specified point. If no file is specified, standard input is used. The *Number* variable specifies how many units to write to standard output. The value for the *Number* variable can be a positive or negative integer. If the value is preceded by + (plus sign), the file is written to standard output starting at the specified number of units from the beginning of the file. If the value is preceded by − (minus sign), the file is written to standard output starting at the specified number of units from the end of the file. If the value is not preceded by + (plus sign) or − (minus sign), the file is read starting at the specified number of units from the end of the file.

The type of unit used by the *Number* variable to determine the starting point for the count is determined by the **-b**, **-c**, **-k**, **-m**, or **-n** flag. If one of these flags is not specified, the **tail** command reads the last ten lines of the specified file and writes them to standard output. This is the same as entering **-n 10** at the command line.

The **-m** flag provides consistent results in both single- and double-byte character environments. The **-c** flag should be used with caution when the input is a text file containing multibyte characters, because output can be produced that does not start on a character boundary.

Flags

-b *Number*	Reads the specified file beginning at the 512-byte block location indicated by the *Number* variable.
-c *Number*	Reads the specified file beginning at the byte location indicated by the *Number* variable.
-f	If the input file is a regular file or if the *File* parameter specifies a FIFO (first-in-first-out), the **tail** command does not terminate after the last specified unit of the input file has been copied, but continues to read and copy additional units from the input file as they become available. If no *File* parameter is specified and standard input is a pipe, the **-f** flag is ignored. The **tail -f** command can be used to monitor the growth of a file being written by another process.
-k *Number*	Reads the specified file beginning at the 1KB block location indicated by the *Number* variable.
-m *Number*	Reads the specified file beginning at the multibyte character location indicated by the *Number* variable. Using this flag provides consistent results in both single- and double-byte character-code-set environments.
-n *Number*	Reads the specified file from the first or last line location as indicated by the sign (+ or - or none) of the *Number* variable and offset by the number of lines *Number*.
-r	Displays the output from the end of the file in reverse order. The default for the **-r** flag prints the entire file in reverse order. If the file is larger than 20,480 bytes, the **-r** flag displays only the last 20,480 bytes.

The **-r** flag is valid only with the **-n** flag. Otherwise, it is ignored.

Exit Status

This command returns the following exit values:

0	Successful completion.
>0	An error occurred.

Examples

1. To display the last 10 lines of the `notes` file, enter:

   ```
   tail notes
   ```
2. To specify the number of lines to start reading from the end of the `notes` file, enter: `tail -n 20 notes`
3. To display the `notes` file a page at a time, beginning with the 200th byte, enter: `tail -c +200 notes | pg`
4. To follow the growth of a file, enter: `tail -f accounts`
   This displays the last 10 lines of the `accounts` file. The **tail** command continues to display lines as they are added to the `accounts` file. The display continues until you press the Ctrl-C key sequence to stop it.

File

/usr/bin/tail	Contains the **tail** command.

Related Information

The **head** command and the **more** command

tee Command

Purpose

Displays the output of a program and copies it into a file.

Syntax

tee **-a** **-i** *File* ...

Description

The **tee** command reads standard input, then writes the output of a program to standard output and simultaneously copies it into the specified file or files.

Flags

-a	Adds the output to the end of *File* instead of writing over it.
-i	Ignores interrupts.

Exit Status

This command returns the following exit values:

0	The standard input was successfully copied to all output files.
>0	An error occurred.

> **Note:** If a write to any successfully opened *File* operand is not successful, writes to other successfully opened *File* operands and standard output will continue, but the exit value will be **>0**.

Examples

1. To view and save the output from a command at the same time:

   ```
   lint program.c | tee program.lint
   ```

   This displays the standard output of the command **lint program.c** at the workstation, and at the same time saves a copy of it in the file `program.lint`. If a file named `program.lint` already exists, it is deleted and replaced.
2. To view and save the output from a command to an existing file: `lint program.c | tee -a program.lint`
   This displays the standard output of the **lint program.c** command at the workstation and at the same time appends a copy of it to the end of the `program.lint` file. If the `program.lint` file does not exist, it is created.

Files

/usr/bin/tee	Contains the **tee** command.

topas Command

Purpose

Reports selected local system statistics.

Syntax

topas **-cpus** *number_of_monitored_hot_CPUs* **-disks** *number_of_monitored_hot_disks* **-interval** *monitoring_interval_in_seconds* **-nets** *number_of_monitored_hot_network_interfaces* **-procs** *number_of_monitored_hot_processes* **-wlms** *number_of_monitored_hot_WLM_classes* **-procsdisp** | **-wlmdisp** | **-cecdisp**

Description

The **topas** command reports selected statistics about the activity on the local system. The command displays its output in a format suitable for viewing on an 80x25 character-based display.

If the **topas** command is invoked without flags, it runs as if invoked with the following command line:

```
topas -disks 20 -interval 2 -nets 20 -procs 20 -wlms 20 -cpus 20
```

The program extracts statistics from the system with an interval specified by the *monitoring_interval_in_seconds* argument. The default output, as shown below, consists of two fixed parts and a variable section. The top two lines at the left of the display show the name of the system the **topas** command runs on, the date and time of the last observation, and the monitoring interval.

The second fixed part fills the rightmost 25 positions of the display. It contains five subsections of statistics, as follows:

EVENTS/QUEUES
Displays the per-second frequency of selected system-global events and the average size of the thread run and wait queues:

Cswitch
The number of context switches per second over the monitoring interval.

Syscalls
The total number of system calls per second executed over the monitoring interval.

Reads
The number of read system calls per second executed over the monitoring interval.

Writes
The number of write system calls per second executed over the monitoring interval.

Forks
The number of fork system calls per second executed over the monitoring interval.

Execs
The number of exec system calls per second executed over the monitoring interval.

Runqueue
The average number of threads that were ready to run but were waiting for a processor to become available.

Waitqueue
The average number of threads that were waiting for paging to complete.

FILE/TTY
Displays the per-second frequency of selected file and tty statistics.

Readch

The number of bytes read per second through the **read** system call over the monitoring interval.

Writech

The number of bytes written per second through the **write** system call over the monitoring interval.

Rawin

The number of raw bytes read per second from TTYs over the monitoring interval.

Ttyout

The number of bytes written to TTYs per second over the monitoring interval.

Igets

The number of calls per second to the inode lookup routines over the monitoring interval.

Namei

The number of calls per second to the pathname lookup routines over the monitoring interval.

Dirblk

The number of directory blocks scanned per second by the directory search routine over the monitoring interval.

PAGING

Displays the per-second frequency of paging statistics.

Faults

Total number of page faults taken per second over the monitoring interval. This includes page faults that do not cause paging activity.

Steals

Physical memory 4K frames stolen per second by the virtual memory manager over the monitoring interval.

PgspIn

Number of 4K pages read from paging space per second over the monitoring interval.

PgspOut

Number of 4K pages written to paging space per second over the monitoring interval.

PageIn

Number of 4K pages read per second over the monitoring interval. This includes paging activity associated with reading from file systems. Subtract **PgspIn** from this value to get the number of 4K pages read from file systems per second over the monitoring interval.

PageOut

Number of 4K pages written per second over the monitoring interval. This includes paging activity associated with writing to file systems. Subtract **PgspOut** from this value to get the number of 4K pages written to file systems per second over the monitoring interval.

Sios

The number of I/O requests per second issued by the virtual memory manager over the monitoring interval.

MEMORY

Displays the real memory size and the distribution of memory in use.

Real,MB

The size of real memory in megabytes.

% Comp

The percentage of real memory currently allocated to computational page frames. Computational page frames are generally those that are backed by paging space.

% Noncomp

The percentage of real memory currently allocated to non-computational frames. Non-computational page frames are generally those that are backed by file space, either data files, executable files, or shared library files.

% Client

The percentage of real memory currently allocated to cache remotely mounted files.

PAGING SPACE

Display size and utilization of paging space.

Size,MB

The sum of all paging spaces on the system, in megabytes.

% Used
The percentage of total paging space currently in use.

% Free
The percentage of total paging space currently free.

NFS

Display NFS stats in calls per second

- Server V2 calls/sec
- Client V2 calls/sec
- Server V3 calls/sec
- Client V3 calls/sec

The variable part of the **topas** display can have one, two, three, four, or five subsections. If more than one subsection displays, they are always shown in the following order:

- CPU
- Network Interfaces
- Physical Disks
- WorkLoad Management Classes
- Processes

When the **topas** command is started, it displays all subsections for which hot entities are monitored. The exception to this is the WorkLoad Management (WLM) Classes subsection, which is displayed only when WLM is active.

CPU Utilization

This subsection displays a bar chart showing cumulative CPU usage. If more than one CPU exists, a list of CPUs can be displayed by pressing the **c** key *twice*. Pressing the **c** key only once will turn this subsection off. The following fields are displayed by both formats:

User%
This shows the percent of CPU used by programs executing in user mode. (Default sorted by User%)

Kern%
This shows the percent of CPU used by programs executing in kernel mode.

Wait%
This shows the percent of time spent waiting for IO.

Idle%
This shows the percent of time the CPU(s) is idle.

When this subsection first displays the list of hot CPUs, the list is sorted by the User% field. However, the list can be sorted by the other fields by moving the cursor to the top of the desired column.

Network Interfaces

This subsection displays a list of hot network interfaces. The maximum number of interfaces displayed is the number of hot interfaces being monitored, as specified with the **-nets** flag. A smaller number of interfaces will be displayed if other subsections are also being displayed. Pressing the **n** key turns off this subsection. Pressing the **n** key again shows a one-line report summary of the activity for all network interfaces. Both reports display the following fields:

Interf
The name of the network interface.

KBPS
The total throughput in megabytes per second over the monitoring interval. This field is the sum of kilobytes received and kilobytes sent per second.

I-Pack
The number of data packets received per second over the monitoring interval.

O-Pack
The number of data packets sent per second over the monitoring interval.

KB-In
The number of kilobytes received per second over the monitoring interval.

KB-Out
The number of kilobytes sent per second over the monitoring interval.

When this subsection first displays the list of hot network interfaces, the list is sorted by the KBPS field. However, the list can be sorted by the other fields by moving the cursor to the top of the desired column. Sorting is only valid for up to 16 network adapters.

Physical Disks

This subsection displays a list of hot physical disks. The maximum number of physical disks displayed is the number of hot physical disks being monitored as specified with the **-disks** flag. A smaller number of physical disks will be displayed if other subsections are also being displayed. Pressing the **d** key turns off this subsection. Pressing the **d** key again shows a one-line report summary of the activity for all physical disks. Both reports display the following fields:

Disk
: The name of the physical disk.

Busy%
: Indicates the percentage of time the physical disk was active (bandwidth utilization for the drive).

KBPS
: The number of kilobytes read and written per second over the monitoring interval. This field is the sum of **KB-Read** and **KB-Read**.

TPS
: The number of transfers per second that were issued to the physical disk. A transfer is an I/O request to the physical disk. Multiple logical requests can be combined into a single I/O request to the disk. A transfer is of indeterminate size.

KB-Read
: The number of kilobytes read per second from the physical disk.

K -Writ
: The number of kilobytes written per second to the physical disk.

When this subsection first displays the list of hot physical disks, the list is sorted by the KBPS field. However, the list can be sorted by the other fields by moving the cursor to the top of the desired column. Sorting is only valid for up to 128 physical disks.

WLM Classes

This subsection displays a list of hot WorkLoad Management (WLM) Classes. The maximum number of WLM classes displayed is the number of hot WLM classes being monitored as specified with the **-wlmdisp** flag. A smaller number of classes will be displayed if other subsections are also being displayed. Pressing the **w** key turns off this subsection. The following fields are displayed for each class:

% CPU Utilization
: The average CPU utilization of the WLM class over the monitoring interval.

% Mem Utilization
: The average memory utilization of the WLM class over the monitoring interval.

% Blk I/O
: The average percent of Block I/O of the WLM class over the monitoring interval.

When this subsection first displays the list of hot WLM classes, the list will be sorted by the CPU% field. However, the list can be sorted by the other fields by moving the cursor to the top of the desired column.

Processes

This subsection displays a list of hot processes. The maximum number of processes displayed is the number of hot processes being monitored as specified with the **-procs** flag. A smaller number of processes will be displayed if other subsections are also being displayed. Pressing the **p** key turns off this subsection. The process are sorted by their CPU usage over the monitoring interval. The following fields are displayed for each process:

Name
: The name of the executable program executing in the process. The name is stripped of any pathname and argument information and truncated to 9 characters in length.

Process ID
: The process ID of the process.

% CPU Utilization
: The average CPU utilization of the process over the monitoring interval. The first time a process is shown, this value is the average CPU utilization over the lifetime of the process.

Paging Space Used
: The size of the paging space allocated to this process. This can be considered an expression of the footprint of the process but does not include the memory used to keep the executable program and any shared libraries it may depend on.

Process Owner (if the WLM section is off)
: The user name of the user who owns the process.

WorkLoad Management (WLM) Class (if the WLM section is on)
: The WLM class to which the process belongs.

Implementation Specifics

Changes to WLM that are shown by **topas** (like adding new classes, or changing existing class names) will not be reflected after starting **topas**. You must stop **topas** and all clients which use **Spmi**, then restart after the WLM changes are made. This is also the case for Disks and Network Adapters added after **topas** or any other **Spmi** consumer is started.

Sample Default Output

The following is an example of the display generated by the **topas** command:

```
Topas Monitor for host:    niller               EVENTS/QUEUES    FILE/TTY
Mon Mar 13 15:56:32 2000   Interval:  2         Cswitch     113  Readch  1853576
                                                Syscall    2510  Writech   49883
CPU   User%   Kern%  Wait%  Idle%               Reads       466  Rawin         0
cpu0    7.0     4.0    0.0   89.0               Writes       12  Ttyout      706
cpu1    1.0     8.0    0.0   91.0               Forks         0  Igets         0
cpu2    0.0     0.0    0.0  100.0               Execs         0  Namei         0
                                                Runqueue    0.0  Dirblk        0
                                                Waitqueue   0.0
Interf   KBPS   I-Pack  O-Pack   KB-In  KB-Out
lo0      100.4    45.7    45.7    50.2    50.2  PAGING           MEMORY
tr0        2.0     4.4     3.4     1.4     0.6  Faults         1  Real,MB     255
                                                Steals         0  % Comp     81.0
Disk    Busy%     KBPS     TPS KB-Read KB-Writ  PgspIn         0  % Noncomp  19.0
hdisk0    0.0      0.0     0.0     0.0     0.0  PgspOut        0  % Client    3.0
hdisk1    0.0      0.0     0.0     0.0     0.0  PageIn         0
                                                PageOut        0  PAGING SPACE
WLM-Class (Active)  CPU%    Mem%     Disk%      Sios           0  Size,MB       0
System               8      41        12                        % Used
Shared               1      24         9                        % Free
                                                NFS      calls/sec
Name     PID      CPU%   PgSP  Class            ServerV2    0   Press:
topas    (35242)  3.0     0.3  System           ClientV2    0   "h" for help.
X        (3622)   1.4    44.4  System           ServerV3    0   "q" to quit.
notes    (25306)  1.3   123.3  System           ClientV3    0
```

Sample Full Screen Process Output

```
Topas Monitor for host:    mothra      Interval:   2    Wed Nov  8 12:27:34 2000
                             DATA  TEXT  PAGE               PGFAULTS
USER       PID  PPID PRI NI   RES   RES SPACE    TIME CPU%  I/O  OTH COMMAND
root      1806     0  37 41    16  3374    16   13:25  1.0    0    0 gil
root      1032     0  16 41     3  3374     3    0:00  0.0    0    0 lrud
root      1290     0  60 41     4  3374     4    0:02  0.0    0    0 xmgc
root      1548     0  36 41     4  3374     4    0:26  0.0    0    0 netm
root         1     0  60 20   197     9   180    0:24  0.0    0    0 init
root      2064     0  16 41     4  3374     4    0:04  0.0    0    0 wlmsched
root      2698     1  60 20    14     2    14    0:00  0.0    0    0 shlap
root      3144     1  60 20    40     1    36    5:19  0.0    0    0 syncd
root      3362     0  60 20     4  3374     4    0:00  0.0    0    0 lvmbb
root      3666     1  60 20   135    23   123    0:00  0.0    0    0 errdemon
root      3982     0  60 20     4  3374     4    0:01  0.0    0    0 rtcmd
root      4644     1  17 20     6  3374     6    0:00  0.0    0    0 dog
root      4912     1  60 20   106    13    85    0:00  0.0    0    0 srcmstr
root      5202  4912  60 20    94     8    84    0:01  0.0    0    0 syslogd
root      5426  4912  60 20   195    76   181    0:12  0.0    0    0 sendmail
root      5678  4912  60 20   161    11   147    0:01  0.0    0    0 portmap
root      5934  4912  60 20   103    11    88    0:00  0.0    0    0 inetd
root      6192  4912  60 20   217    61   188    0:21  0.0    0    0 snmpd
root      6450  4912  60 20   137    10   116    0:00  0.0    0    0 dpid2
root      6708  4912  60 20   157    29   139    0:06  0.0    0    0 hostmibd
root         0     0  16 41     3  3374     3    7:08  0.0    0    0
root      6990     1  60 20   106    10    86    0:06  0.0    0    0 cron
```

Sample Full-Screen WorkLoad Management Classes Output

```
Topas Monitor for host:    mothra      Interval:   2    Wed Nov  8 12:30:54 2000
WLM-Class (Active)              CPU%      Mem%     Disk-I/O%
System                           0         0            0
Shared                           0         0            0
```

```
Default                         0         0            0
Unmanaged                       0         0            0
Unclassified                    0         0            0



==============================================================================
                            DATA  TEXT  PAGE               PGFAULTS
USER        PID  PPID PRI NI   RES   RES SPACE    TIME CPU%  I/O  OTH COMMAND
root          1     0 108 20   197     9   180    0:24  0.0    0    0 init
root       1032     0  16 41     3  3374     3    0:00  0.0    0    0 lrud
root       1290     0  60 41     4  3374     4    0:02  0.0    0    0 xmgc
root       1548     0  36 41     4  3374     4    0:26  0.0    0    0 netm
root       1806     0  37 41    16  3374    16   13:25  0.0    0    0 gil
root       2064     0  16 41     4  3374     4    0:04  0.0    0    0 wlmsched
root       2698     1 108 20    14     2    14    0:00  0.0    0    0 shlap
root       3144     1 108 20    40     1    36    5:19  0.0    0    0 syncd
root       3362     0 108 20     4  3374     4    0:00  0.0    0    0 lvmbb
root       3666     1 108 20   135    23   123    0:00  0.0    0    0 errdemon
root       3982     0 108 20     4  3374     4    0:01  0.0    0    0 rtcmd
```

Flags

-cecdisp Displays the cross-partition panel.

-cpus Specifies the number of hot CPUs to be monitored. This is also the maximum number of CPUs displayed when enough room is available on the screen. If this number exceeds the number of CPUs available, only the installed CPUs will be monitored and displayed. If this argument is omitted, a default of 2 is assumed. If a value of 0 (zero) is specified, no CPU information is monitored.

-disks Specifies the number of disks to be monitored. This is also the maximum number of disks displayed when enough room is available on the screen. When this number exceeds the number of disks installed, only the installed disks will be monitored and displayed. If this argument is omitted, a default of 2 is assumed. If a value of 0 (zero) is specified, no disk information is monitored.

-interval Sets the monitoring interval in seconds. The default is 2 seconds.

-nets Specifies the number of hot network interfaces to be monitored. This is also the maximum number of network interfaces displayed when enough room is available on the screen. When this number exceeds the number of network interfaces installed, only the installed network interfaces will be monitored and displayed. If this argument is omitted, a default of 2 is assumed. If a value of 0 (zero) is specified, no network information is monitored.

-procsdisp Displays the full-screen process display. This display shows a list of the busiest processes, similar to the process subsection on the default display, only with more columns showing more metrics per process. This list can be sorted by any column.

-procs Specifies the number of hot processes to be monitored. This is also the maximum number of processes shown when enough room is available on the screen. If this argument is omitted, a default of 20 is assumed. If a value of 0 is specified, no process information will be monitored. Retrieval of process information constitutes the majority of the **topas** overhead. If process information is not required, always use this option to specify that you do not want process information.

-wlmdisp Displays the full-screen WLM class display, which is a split display. The top part of the display shows a list of hot WLM classes, similar to the WLM classes subsection on the default display, but with enough space available to display the full class names. This list can be sorted on any column.

The bottom part of the display shows a list of busiest processes, similar to the full screen process display, but only displays processes belonging to one WLM class (selected with the **f** key).

-wlms Specifies the number of hot WorkLoad Management (WLM) classes to be monitored. This is also the maximum number of WLM classes displayed when enough room is available on the screen. If this number exceeds the number of WLM classes installed, only the installed WLM classes will be monitored and displayed. If this argument is omitted, a default of 2 is assumed. If a value of 0 (zero) is specified, no WLM class information is monitored.

Subcommands

While **topas** is running, it accepts one-character subcommands. Each time the monitoring interval elapses, the program checks for one of the following subcommands and responds to the action requested.

a
The **a** key shows all of the variable subsections being monitored (CPU, network, disk, WLM, and process). Pressing the **a** key always returns the **topas** command to the initial main display.

c
The **c** key toggles the CPU subsection between the cumulative report, off, and a list of the busiest CPUs. The number of busiest CPUs displayed will depend upon the space available on the screen.

d
The **d** key toggles the disk subsection between a list of busiest disks, off, and the report on the total disk activity of the system. The number of busiest disks displayed will depend upon the space available on the screen.

h
Show the help screen.

n
The **n** key toggles the network interfaces subsection between a list of busiest interfaces, off, and the report on the total network activity of the system. The number of busiest interfaces displayed will depend upon the space available on the screen.

w
The **w** key toggles the WorkLoad Management (WLM) classes subsection on and off. The number of busiest WLM classes displayed will depend upon the space available on the screen.

p
The **p** key toggles the hot processes subsection on and off. The number of busiest processes displayed will depend upon the space available on the screen.

P
The uppercase **P** key replaces the default display with the full-screen process display. This display provides more detailed information about processes running on the system than the process section of the main display. When the **P** key is pressed again, it toggles back to the default main display.

W
The uppercase **W** key replaces the default display with the full-screen WLM class display. This display provides more detailed information about WLM classes and processes assigned to classes. When the **W** key is pressed again, it toggles back to the default main display.

L
The uppercase L key replaces the current display with the logical partition display; Micro-Partitioning and SMT metrics similar to what **lparstat** and **mpstat** provide are displayed.

f
Moving the cursor over a WLM class and pressing the **f** key displays the list of top processes in the class at the bottom of the WLM screen. This key is valid only when topas is in the full-screen WLM display (by using the **W** key or the **-wlms** flag).

q
Quit the program.

r
Refresh the display.

Arrow and Tab keys
Subsections from the main display such as the CPU, Network, Disk, WLM Classes, and the full-screen WLM and Process displays can be sorted by different criteria. Positioning the cursor over a column activates sorting on that column. The entries are always sorted from highest to lowest value. The cursor can be moved by using the **Tab** key or the arrow keys. Sorting is only valid for 128 disks and 16 network adapters.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To display up to twenty "hot" disks every five seconds and omit network interface, WLM classes, and process information, type:

   ```
   topas -interval 5 -nets 0 -procs 0 -wlms 0
   ```
2. To display the five most active processes and up to twenty most active WLM classes (which is the default when omitting the **-w** flag) but no network or disk information, type:

   ```
   topas -procs 5 -nets 0 -disks 0
   ```
3. To run the program with default options, type:

   ```
   topas
   ```
4. To go directly to the process display, type:

   ```
   topas -procdisp
   ```
5. To go directly to the WLM classes display, type:

   ```
   topas -wlmdisp
   ```

traceroute Command

Purpose

Prints the route that IP packets take to a network host.

Syntax

traceroute **-hops** *Hops* **-num** **-port** *Port* **-src** *Address Host PacketSize*

Description

The **traceroute** command attempts to trace the route an IP packet follows to an Internet host by launching UDP probe packets with a small maximum time-to-live (*Hops* parameter), then listening for an **ICMP TIME_EXCEEDED** response from gateways along the way. Probes are started with a Hops value of one hop, which is increased one hop at a time until an **ICMP PORT_UNREACHABLE** message is returned. The **ICMP PORT_UNREACHABLE** message indicates either that the host has been located or the command has reached the maximum number of hops allowed for the trace.

The **traceroute** command sends three probes at each Hops setting to record the following:

- Hops value
- Address of the gateway
- Round-trip time of each successful probe

If the probe answers come from different gateways, the command prints the address of each responding system. If there is no response from a probe within a 3-second time-out interval, an * (asterisk) is printed for that probe.

Note: The **traceroute** command is intended for use in network testing, measurement, and management. It should be used primarily for manual fault isolation. Because of the load it imposes on the network, the traceroute command should not be used during normal operations or from automated scripts.

The **traceroute** command prints an ! (exclamation mark) after the round-trip time if the Hops value is one hop or less. A maximum time-to-live value of one hop or less generally indicates an incompatibility in the way ICMP replies are handled by different network software. The incompatibility can usually be resolved by doubling the last Hops value used and trying again.

Other possible annotations after the round-trip notation are as follows:

!H	Host unreachable
!N	Network unreachable
!P	Protocol unreachable
!S	Source route failed
!F	Fragmentation needed

If the majority of probes result in an error, the **traceroute** command exits.

The only mandatory parameter for the **traceroute** command is the destination host name or IP number. The **traceroute** command will determine the length of the probe packet based on the Maximum Transmission Unit (MTU) of the outgoing interface. The UDP probe packets are set to an unlikely value so as to prevent processing by the destination host.

Flags

-hops *Max_ttl* Sets the maximum time-to-live (maximum number of hops) used in outgoing probe packets. The default is 30 hops (the same default used for TCP connections).

-num Prints hop addresses numerically rather than symbolically and numerically. This flag saves a name-server address-to-name lookup for each gateway found on the path.

-port *Port* Sets the base UDP port number used in probes. The default is 33434. The **traceroute** command depends on an open UDP port range of base to base + nhops - 1 at the destination host. If a UDP port is not available, this option can be used to pick an unused port range.

-src *SRC_Addr* Uses the next IP address in numerical form as the source address in outgoing probe packets. On hosts with more than one IP address, the **-src** flag can be used to force the source address to be something other than the IP address of the interface on which the probe packet is sent. If the next IP address is not one of the machine's interface addresses, an error is returned and nothing is sent.

Parameters

Host Specifies the destination host, either by host name or IP number. This parameter is required.

PacketSize Specifies the probe datagram length. The default packet size is determined by the **traceroute** command based on the MTU of the outgoing interface.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To print the route to host **nis.nfs.net**, type:

   ```
   traceroute nis.nsf.net
   ```

   The output will look similar to the following:

   ```
   traceroute to rotterdam (35.1.1.48), 30 hops max, 56 byte packet
   1 helios.ee.lbl.gov (128.3.112.1) 19 ms 19 ms 0 ms
   2 lilac-dmc.Berkeley.EDU (128.32.216.1) 39 ms 39 ms 19 ms
   3 lilac-dmc.Berkeley.EDU (128.32.216.1) 39 ms 39 ms 19 ms
   4 ccngw-ner-cc.Berkeley.EDU (128.32.136.23) 39 ms 40 ms 39 ms
   5 ccn-nerif22.Berkeley.EDU (128.32.168.22) 39 ms 39 ms 39 ms
   6 128.32.197.4 (128.32.197.4) 40 ms 59 ms 59 ms
   7 131.119.2.5 (131.119.2.5) 59 ms 59 ms 59 ms
   8 129.140.70.13 (129.140.70.13) 99 ms 99 ms 80 ms
   9 129.140.71.6 (129.140.71.6) 139 ms 239 ms 319 ms
   10 129.140.81.7 (129.140.81.7) 220 ms 199 ms 199 ms
   11 nic.merit.edu (35.1.1.48) 239 ms 239 ms 239 ms
   ```

Related Information

The **ping** command and the **optimizenet** command.

unmirrorios Command

Purpose

Removes the mirrors that exist on the rootvg volume group.

Syntax

unmirrorios *PhysicalVolume* ...

Description

The **unmirrorios** command unmirrors all the logical volumes detected on the rootvg volume group. By default, **unmirrorios** will pick the set of mirrors to remove from a mirrored volume group. To control which drives no longer are to contain mirrors, you must include the list of disks in the input parameters, *PhysicalVolume*.

At the completion of this command, Quorum will be disabled until the system is rebooted.

Note: If LVM has not recognized that a disk has failed, it is possible that it will remove a different mirror. If you know that a disk has failed, and LVM does not show those disks as missing, you should specify the failed disks on the command line or you should use the **reducevg** command to remove the disk.

Exit Status

See Virtual I/O Server command exit status.

Security

This command can only be executed by the prime administrator.

Related Information

The **activatevg** command, the **chvg** command, the **deactivatevg** command, the **exportvg** command, the **importvg** command, the **lsvg** command, the **mkvg** command, the **syncvg** command, and the **mirrorios** command.

unmount Command

Purpose

Unmounts a previously mounted file system, directory, or file.

Syntax

unmount { *Directory* | *File* | *FileSystem* }

Description

The **unmount** command unmounts a previously mounted directory, file, or file system. Processing on the file system, directory, or file completes and it is unmounted.

To unmount local mounts you can specify the device, directory, file, or file system on which it is mounted.

Exit Status

See Virtual I/O Server command exit status.

Examples

1. To unmount files and directories, type the following command:

   ```
   unmount /home/user/test
   ```

   This unmounts the filesystem mounted at /**home**/**user**/**test**.

Related Information

The **mount** command.

updateios Command

Purpose

Updates the Virtual I/O Server to latest maintenance level.

Syntax

updateios **-dev** *Media* **-f** **-install** **-accept**

updateios **-commit** | **-reject** **-f**

updateios **-cleanup**

updateios **-remove** { **-file** *RemoveListFile* | *RemoveList* }

Description

The **updateios** command is used to install fixes, or updates the Virtual I/O Server to the latest maintenance level. Before installing a fix or maintenance level, the **updateios** command will first run a preview installation and display the results. Upon completion of the preview, the user will then be prompted to continue or exit. If the preview fails for any reason, the updates should not be installed.

The **-install** flag is used to install new file sets onto the Virtual I/O Server. This flag should not be used to install fixes or maintenance levels.

The **-cleanup** flag cleans up after an interrupted installation and attempts to remove all incomplete pieces of the previous installation. Cleanup should be performed whenever any software product or update is in a state of either applying or committing and can be run manually as needed.

The **-commit** flag will commit all uncommitted updates to the Virtual I/O Server.

The **-reject** flag will reject all uncommitted updates to the Virtual I/O Server.

If the **-remove** flag is specified, the listed file sets will be removed from the system. The file sets to be removed must be listed on the command line or in the **RemoveListFile** file.

The log file, **install.log** in the user's home directory, will be overwritten with a list of all file sets that were installed.

Flags

-accept	Agrees to required software license agreements for software to be installed.
-cleanup	Cleans up after an interrupted installation or update.
-commit	Commits all specified updates.
-dev *Media*	Specifies the device or directory containing the images to install.
-f	Forces all uncommitted updates to be committed prior to applying the new updates. When combined with the **-dev** flag, commits all updates prior to applying any new ones. When combined with the **-reject** flag, rejects all uncommitted updates with out prompting for confirmation.
-file *file*	Specifies the file containing a list of entries to uninstall.
-install	Installs new file sets onto the Virtual I/O Server.
-reject	Rejects all specified uncommitted updates.
-remove	Performs an uninstall of the specified software.

Exit Status

19	All uncommitted updates must be committed
20	There are no uncommitted updates

Examples

1. To update the Virtual I/O Server to the latest level, where the updates are located on the mounted filesystem /**home**/**padmin**/**update**, type:

   ```
   updateios -dev /home/padmin/update
   ```
2. To update the Virtual I/O Server to the latest level, when previous levels are not committed, type:

   ```
   updateios -f -dev /home/padmin/update
   ```
3. To reject installed updates, type

   ```
   updateios -reject
   ```
4. To cleanup partial installed updates, type

   ```
   updateios -cleanup
   ```
5. To commit the installed updates, type

   ```
   updateios -commit
   ```

Related Information

The **lssw** command, the **ioslevel** command, the **remote_management** command, the **oem_setup_env** command, and the **oem_platform_level** command.

vi Command

Purpose

Edits files with a full-screen display.

Syntax

{ **vi** | **vedit** } **-l** **-R** **-t***Tag* **-v** **-w***Number* **-y***Number* **-r** *File* { **+** | **-c** } { *Subcommand* } *File* ...

Description

The **vi** command starts a full-screen editor based on the underlying ex editor. Therefore, ex subcommands can be used within the vi editor. The **vedit** command starts a version of the vi editor intended for beginners. In the vedit editor, the **report** option is set to 1, the **showmode** option is set, and the **novice** option is set, making it a line editor.

You start the vi editor by specifying the name of the file or files to be edited. If you supply more than one *File* parameter on the command line, the vi editor edits each file in the specified order. The vi editor on an existing file displays the name of the file, the number of lines, and the number of characters at the bottom of the screen. In case of multibyte locales the number of characters need to be interpreted as the number of bytes.

Since the vi editor is a full-screen editor, you can edit text on a screen-by-screen basis. The vi editor makes a copy of the file you are editing in an edit buffer, and the contents of the file are not changed until you save the changes. The position of the cursor on the display screen indicates its position within the file, and the subcommands affect the file at the cursor position.

vi Editor Limitations

The following list provides the maximum limits of the vi editor. These counts assume single-byte characters.

- 8192 characters per line
- 256 characters per global command list
- 128 characters in the previously inserted and deleted text
- 128 characters in a shell escape command
- 128 characters in a string-valued option
- 30 characters in a tag name
- 128 map macros with 2048 characters total
- 1,048,560 lines of 8192 characters per line silently enforced
- The macro name and the macro text are limited to 100 characters.

Note: Running the vi editor on a file larger than 64 MB may cause the following error message to display:

```
0602-103 file too large to place in /tmp
```

vi Editing Modes

The vi editor operates in the following modes:

command mode
When you start the vi editor, it is in command mode. You can enter any subcommand except those designated for use only in the text input mode. The vi editor returns to command mode when subcommands and other modes end. Press the Esc key to cancel a subcommand.

text-input mode
You use the vi editor in this mode to add text. Enter text input mode with any of the following subcommands: the **a** subcommand, **A** subcommand, **i** subcommand, **I** subcommand, **o** subcommand, **O** subcommand, **c***x* subcommands (where the *x* represents the scope of the subcommand), **C** subcommand, **s** subcommand, **S** subcommand, and **R** subcommand. After entering one of these subcommands, you can

enter text into the editing buffer. To return to command mode, press the Esc key for normal exit or press Interrupt (the Ctrl-C key sequence) to end abnormally.

Subcommands with the prefix **:** (colon), / (slash), **?** (question mark), **!** (exclamation point), or **!!** (two exclamation points) read input on a line displayed at the bottom of the screen. When you enter the initial character, the vi editor places the cursor at the bottom of the screen, where you enter the remaining characters of the command. Press the Enter key to run the subcommand, or press Interrupt (the Ctrl-C key sequence) to cancel it. When the **!!** prefix is used, the cursor moves only after both exclamation points are entered.

last-line mode
When you use the **:** prefix to enter the last-line mode, the vi editor gives special meaning to the following characters when they are used before commands that specify counts:

%
All lines regardless of cursor position

$
Last line

.
Current line

Customizing the vi Editor

You can customize the vi editor by:

- Setting vi editor options
- Defining macros
- Mapping keys
- Setting abbreviations

Setting vi Editor Options

The following list describes the vi editor options you can change with the **set** command. The default setting for these options is **off**. If you turn on one of these toggle options, you can turn it off again by entering the word **no** before the option. If you want to discontinue the **autowrite** vi option, enter **noaw**, where **no** turns off the option and **aw** specifies the **autowrite** option.

Note: Do not include parentheses when entering vi options.

vi Option (Abbreviation)	Description
autoindent (ai)	Indents automatically in text input mode to the indentation of the previous line by using the spacing between tab stops specified by the **shiftwidth** option. The default is **noai**. To back the cursor up to the previous tab stop, press the Ctrl-D key sequence. This option is not in effect for global commands.
autoprin (ap)	Prints the current line after any command that changes the editing buffer. The default is **ap**. This option applies only to the last command in a sequence of commands on a single line and is not in effect for global commands.
autowrite (aw)	Writes the editing buffer to the file automatically before the **:n** subcommand, the **:ta** subcommand, the Ctrl-A, Ctrl -, and Ctrl -T key sequences, and the **!** subcommand if the editing buffer changed since the last **write** subcommand. The default is **noaw**.
backtags (bt)	Allows the Ctrl-T subcommand to return the file editing position to the location where the previous Ctrl- subcommand was issued. If **nobacktags** is set, then Ctrl-T is the same as Ctrl-. The default is **backtags**.
beautifying text (bf)	Prevents the user from entering control characters in the editing buffer during text entry (except for tab, new-line, and form-feed indicators). The default is **nobf**. This option applies to command input.
closepunct (cp=)	Handles a list of closing punctuation, especially when wrapping text (**wraptype** option). Precedes multicharacter punctuation with the number of characters; for example, `cp=3..;)}`. The **vi** command does not split closing punctuation when wrapping.
directory (dir=)	Displays the directory that contains the editing buffer. The default is **dir = /var/tmp**.

edcompatible (ed)
Retains **g** (global) and **c** (confirm) subcommand suffixes during multiple substitutions and causes the **r** (read) suffix to work like the **r** subcommand. The default is **noed.**

exrc (exrc)
If not set, ignores any **.exrc** file in the current directory during initialization, unless the current directory is that named by the **HOME** environment variable. The default is **noexrc**.

hardtabs (ht=)
Tells the vi editor the distance between the hardware tab stops on your display screen. (This option must match the tab setting of the underlying terminal or terminal emulator.) The default is **ht=8**.

ignorecase (ic)
Ignores distinction between uppercase and lowercase while searching for regular expressions. The default is **noic**.

linelimit (ll=)
Sets the maximum number of lines, as per the **-y** command-line option. This option only is effective if used with the **.exrc** file or the **EXINIT** environment variable.

lisp (lisp)
Removes the special meaning of ( ), { }, , and and enables the = (formatted print) operator for s-expressions, so you can edit list processing (LISP) programs. The default is **nolisp.**

list (list)
Displays text with tabs (^**I**) and the marked end of lines ($). The default is **nolist.**

magic (magic)
Treats the . (period), (left bracket), and * (asterisk) characters as special characters when searching for a pattern. In off mode, only the ( ) (parentheses) and $ (dollar sign) retain special meanings. However, you can evoke special meaning in other characters by preceding them with a \ (backslash). The default is **magic**.

mesg (mesg)
Turns on write permission to the terminal if set while in visual mode. This option only is effective if used with the **.exrc** file or the **EXINIT** environment variable. The default is **on**.

modeline (modeline)
Runs a vi editor command line if found in the first five or the last five lines of the file. A vi editor command line can be anywhere in a line. For the vi editor to recognize a command line, the line must contain a space or a tab followed by the `ex:` or `vi:` string. The command line is ended by a second **:** (colon). The vi editor tries to interpret any data between the first and second colon as vi editor commands. The default is **nomodeline**.

novice
Indicates whether you are in **novice** mode. You cannot change the value by using the **set** command.

number (nu)
Displays lines prefixed with their line numbers. The default is **nonu**.

optimize (opt)
Speeds the operation of terminals that lack cursor addressing. The default is **noopt**.

paragraphs (para=)
Defines vi macro names that start paragraphs. The default is **para=IPLPPPQPP\ LIpplpipnpbp**. Single-letter **nroff** macros, such as the **.P** macro, must include the space as a quoted character if respecifying a paragraph.

partialchar (pc=)
Appears in the last display column where a double-wide character would not be displayed completely. The default character is **-** (minus sign).

prompt
Prompts for a new vi editor command when in command mode by printing a **:** (colon). The default is **on**.

readonly (ro)
Sets permanent read-only mode. The default is **noreadonly**.

redraw (redraw)
Simulates a smart workstation on a dumb workstation. The default is **nore**.

remap
Allows defining macros in terms of other macros. The default is **on**.

report (re=)
Sets the number of times you can repeat a command before a message is displayed. For subcommands that produce many messages, such as global subcommands, the messages are displayed when the command sequence completes. The default is **report=5.**

scroll (scr=)
Sets the number of lines to be scrolled when the user scrolls up

or down. The default is 1/2 of the window size, rounded down.

sections (sect=)
Defines vi macro names that start sections. The default is **sect=NHSHHH**\ **HUuhsh+c**. Single-letter **nroff** macros, such as the **.P** macro, must include the space as a quoted character if respecifying a paragraph.

shell (sh=)
Defines the shell for the **!** subcommand or the **:!** subcommand. The default is the login shell.

shiftwidth (sw=)
Sets the distance for the software tab stops used by the **autoindent** option, the shift commands **(>** and **<)**, and the text input commands (the Ctrl-D and Ctrl-T key sequences). This vi option only affects the indentation at the beginning of a line. The default is **sw=8.**

showmatch (sm)
Shows the ((matching left parenthesis) or { (left bracket) as you type the) (right parenthesis) or } (right bracket). The default is **nosm**.

showmode (smd)
Displays a message to indicate when the vi editor is in input mode. The default is **nosmd**.

slowopen (slow)
Postpones updating the display screen during inserts. The default is **noslow.**

tabstop (ts=)
Sets the distance between tab stops in a displayed file. The default is **ts=8**.

tags (tags =)
Defines the search path for the database file of function names created using the **ctags** command. The default is **tags=tags**\ **/usr/lib/tags**.

term (term=)
Sets the type of workstation you are using. The default is **term=$TERM**, where **$TERM** is the value of the **TERM** shell variable.

terse (terse)
Allows the vi editor to display the short form of messages. The default is **noterse**.

timeout (to)
Sets a time limit of two seconds on an entry of characters. This limit allows the characters in a macro to be entered and processed as separate characters when the **timeout** option is set. To resume use of the macro, set the **notimeout** option. The default is **to**.

ttytype
Indicates the tty type for the terminal being used. You cannot change this value from the vi editor.

warn (warn)
Displays a warning message before the **!** subcommand executes a shell command if it is the first time you issued a shell command after changes were made in the editing buffer but not written to a file. The default is **warn**.

window (wi=)
Sets the number of lines displayed in one window of text. The default depends on the baud rate at which you are operating: 600 baud or less, 8 lines; 1200 baud, 16 lines; higher speeds, full screen minus 1 line.

wrapmargin (wm=)
Sets the margin for automatic word wrapping from one line to the next. The default is **wm=0**. A value of 0 turns off word wrapping.

wrapscan (ws)
Allows string searches to wrap from the end of the editing buffer to the beginning. The default is **ws**.

wraptype (wt=)
Indicates the method used to wrap words at the end of a line. The default value is **general**. You can specify one of the following four values:

general
Allows wraps on word breaks as white space between two characters. This setting is the default.

word
Allows wraps on words.

rigid
Allows wraps on columns and before closing punctuation.

flexible
Allows wraps on columns, but one character of punctuation can extend past the margin.

writeany (wa)	Turns off the checks usually made before a **write** subcommand. The default is **nowa**.

To see a list of the vi editor settings that have changed from the default settings, enter `set` and press the spacebar. Press the Enter key to return to the command mode.

To see a complete list of the vi editor settings, enter `set all`. Press the Enter key to return to the command mode.

To turn on a vi editor option, enter `set Option`. This command automatically returns you to the command mode.

To turn on multiple vi editor options, enter `set Option Option Option`. This command turns on the three designated vi editor options and returns you to the command mode.

To turn off a vi editor option, enter `set noOption`. This command automatically returns you to the command mode.

To change the value of a vi editor option, enter `set Option=Value`. This command automatically returns you to the command mode.

You can use the **:set** subcommand of the vi editor to set options for this editing session only, or to set options for this editing session and all future editing sessions.

To set or change vi editor options *for this editing session only*, enter the **:set** subcommand from the command line.

To set vi options for *all editing sessions*, put the **:set** subcommand in the **EXINIT** environment variable in the **.profile** file (read by the shell on login) or put the **set** subcommand into a **.exrc** file. The vi editor first looks for the **EXINIT** environment variable and runs its commands. If the **EXINIT** environment variable does not exist, the vi editor then looks for the **$HOME/.exrc** file and runs its commands. Last, and regardless of any previous results, the vi editor looks for the local **.exrc** file and runs its commands.

Note: This process is true except with the **tvi** command (trusted vi). In this instance, the vi editor looks for and runs only the /**etc/.exrc** file.

For information about changing an option by setting the **EXINIT** environment variable, see the description of environment variables in the **environment** file.

The **.exrc** file can contain subcommands of the form **set** *Option***=***Value*; for example:

```
set cp=3 . . ;
```

To include a comment in the **.exrc** file, use a **"** (double quotation mark) as the first character in the line.

Defining Macros

If you use a subcommand or sequence of subcommands frequently, you can use the vi editor to define a macro that issues that subcommand or sequence.

To define a macro, enter the sequence of subcommands into a buffer named with a letter of the alphabet. The lowercase letters a through z overlay the contents of the buffer, and the uppercase letters A through Z append text to the previous contents of the buffer, allowing you to build a macro piece by piece.

For example, to define a buffer macro named c that searches for the word corner and makes the third line after the word corner the current line, enter the following command:

```
o /corner/+3
```

Then press the Esc key and enter the following command:

```
"c
```

where c is the name of the buffer macro.

To add text to the previous contents of the defined buffer, enter the `o viSubcommand`, press the Esc key, and enter `"CapitalLetter`, where the *CapitalLetter* variable specifies an uppercase letter A through Z. For example, to build a buffer macro named T that searches for the word corner and allows you to add more commands, enter the following command:

```
o corner
```

Then press the Esc key and enter the following command:

```
"T
```

where T is the name of the buffer macro. You can repeat this process at any time to add more vi subcommands to the same buffer.

For example, to add commands that move the cursor to the previous line and delete that line, enter the following command:

```
o -dd
```

where - (minus sign) means to move the cursor up one line, and dd means to delete the current line. Press the Esc key and enter the following command:

```
"Tdd
```

To start the macro, enter `@Letter`, where the *Letter* variable specifies the letter name of the buffer macro you want to use. To use the same macro again, enter `@@` (two at symbols). For example, enter `@T` to start the T buffer macro and run the **search**, **move cursor**, and **delete line** commands. Enter `@@T` to start the T buffer macro again.

The character set used by your system is defined by the collation table. This table affects the performance of vi macros.

Mapping Keys

You can use the **:map**, **:map!**, and **:ab** subcommands to map a keystroke to a command or a sequence of commands. The **:map** subcommand is used in the command mode. The **:map!** and **:ab** subcommands are used in the text input mode. You can map keys for this editing session and all future editing sessions or only for the current editing session from either mode.

To map keys *for all future editing sessions*, put the subcommand into a **$HOME/.exrc** file. Each time you start the vi editor, it reads this file. The mapping remains in effect for every editing session.

To map keys *for the current editing session only* from the *command mode*, start the subcommand during the vi editor session. To map keys for the current editing session only from the *text input mode*, enter the subcommand on the command line during the vi editor session. The mapping remains in effect only for the current editing session.

Attention: If you use an IBM 3161 ASCII display station, IBM 3163 ASCII display station, or IBM 3101 ASCII display station, the default key-mapping of the vi editor can cause you to lose data. To see the default mapping, issue a **:map** subcommand. Specific problems arise with the Esc-J or Shift-J key sequence. These key sequences delete all information from the current position of the cursor to the end of the file. To avoid problems, change this key sequence using a **.exrc** file.

The **:map**, **:map!**, and **:ab** subcommands are defined and used as follows:

:map Defines macros in the command mode. The **:map** subcommand allows you to run a specified command or sequence of commands by pressing a single key while in the vi editor.

To map keys in the command mode, start the vi editor with an empty editing buffer and do not name a vi file using the **vi** command or type anything into the buffer after the vi editor starts. You can use the **:map** subcommand to do the following:

- To map a character to a sequence of editing commands, enter:

```
:map Letter viSubcommand
```

- To unmap a character previously mapped in command mode, enter:

```
:unmap Letter
```

- To display a list of current mappings for the command mode, enter

```
:map
```

The following keys are not used by the vi editor, but are available for use with the **:map** subcommand in the command mode:

- Letters g, K, q, V, and v
- Control key sequences Ctrl-A, Ctrl-K, Ctrl-O, Ctrl-W, and Ctrl-X
- Symbols _ (underscore), * (asterisk), \ (backslash), and = (equal sign)

Although you can map a key that is already used by the vi editor, the key's usual function is not available as long as the map is in effect. Some terminals allow you to map command sequences to function keys. If you are in LISP mode, the = (equal sign) cannot be used because it is used by the vi editor.

To map the letter v to the sequence of commands that would locate the next occurrence of the word `map` and change it to the word `MAP`, enter the following command:

```
:map v /map<Ctrl-V><Enter>cwMAP<Ctrl-V><Esc><Ctrl-V><Enter>
```

The previous example instructs the vi editor to locate the next occurrence of `map` (`/map<Ctrl-V><Enter>`), change `map` to `MAP` (`cwMAP`), end the change-word subcommand (`<Ctrl-V><Esc>`), and enter the command (`<Ctrl-V><Enter>`).

Note: To prevent the vi editor from interpreting the Enter key, it must be preceded by the Ctrl-V key sequence when being mapped. This condition is also true of the Esc, Backspace, and Delete keys.

To map the control characters Ctrl-A, Ctrl-K, and Ctrl-O, simultaneously press the Ctrl key and the letter. For example, to map the Ctrl-A key sequence to the sequence of commands that saves a file and edits the next one in a series, enter the following command:

```
:map <Ctrl-A> :w<Ctrl-V><Enter>:n<Ctrl-V><Enter>
```

To map the control characters Ctrl-T, Ctrl-W, and Ctrl-X, you must first escape them with the Ctrl-V key sequence.

To map the | (pipe symbol), you must first escape it with the two Ctrl-V key sequences, as illustrated by the following example that maps the character g to the sequence of commands that escapes to the shell, concatenates the file **/etc/motd**, and pipes the output to the **wc** command:

```
:map g :!cat /etc/motd <Ctrl-V><Ctrl-V>| wc<Ctrl-V><Enter>
```

If your terminal permits you to map function keys, you must reference them with the #*number* key sequence to designate the number of the function key that you want to map. In the following example, the F1 function key is mapped to the sequence of commands that deletes a word and moves the cursor three words down:

```
:map #1 dwwww
```

In order for function key mapping to work, the output of the function key for your terminal type must match the output defined in the **terminfo** file. These definitions are denoted by the kf*number* entries, where kf1 represents the F1 function key, kf2 represents the F2 function key, and so on. If the output that you get when you press the function key does not match this entry, you must use the terminal's setup mode to correct the settings to match these terminal database entries before any mapping can occur.

You can also map certain keyboard special keys, such as the Home, End, Page Up, and Page Down keys. For most terminals, these keys are already mapped in the vi editor. You can verify this mapping by using the **:map** subcommand. If these keys are not already mapped, you can use the **:map** subcommand as follows:

```
:map <Ctrl-V><End> G
:map <Ctrl-V><Home> 1G
:map <Ctrl-V><PageUp> <Ctrl-F>
:map <Ctrl-V><PageDown> <Ctrl-B>
```

To get a listing of all current maps in the command mode, enter the **:map** subcommand. The preceding examples are then displayed as follows:

```
v         v          /map<Ctrl-M>cwMAP<Ctrl->Ctrl-M>
<Ctrl-A>  <Ctrl-A>   :w<Ctrl-M>:n<Ctrl-M>
g         g          :!cat /etc/motd | wc <Ctrl-M>
```

Note: The Ctrl-V and Enter key sequence is displayed as the Ctrl-M key sequence, and the Ctrl-V and Esc key sequence is displayed as the Ctrl- key sequence.

:map!

Maps character strings to single keys while in text input mode. To map keys in the text input mode, start the vi editor with an empty editing buffer and do not name a vi file using the **vi** command or type anything into the buffer after the vi editor starts. You can use the **:map!** subcommand to do the following:

- To map a letter to one or more vi strings in text input mode, enter:

  ```
  :map! Letter String
  ```
- To unmap a letter previously mapped in text input mode, enter:

  ```
  :unmap! Letter
  ```
- To display a list of existing strings that are mapped to specific keys in text input mode, enter:

  ```
  :map!
  ```

Typing the mapped key while in text input mode produces the specified string. The Ctrl-V and Esc key sequence puts you into command mode, backs up to the beginning of the current word (**bbw**), and starts the **cw** (change-word) subcommand. For example:

```
:map! % <Ctrl-V><Esc>bbwcw
```

When typing text, if you realize that you have mistyped a word, you can change it by pressing the % (percent) key and retyping the word. You are automatically returned to insert mode.

Note: Be careful when choosing keys to be used for the **:map!** subcommand. Once keys have been mapped, they can no longer be input as text without first issuing the **:unmap!** subcommand.

:ab

Maps a key or sequence of keys to a string of characters for use in the text input mode. The **:ab** subcommand is useful when inputting text that possesses several repetitive phrases, names, or titles.

The following example replaces the word `city` with the phrase `Austin, Texas 78759` whenever it is typed in text input mode and followed by a white space, period, or comma:

```
:ab city Austin, Texas 78759
```

For example, if while inputting text, you type the following:

```
My current residence is city.
```

Pressing the Tab key expands the word `city` to read:

```
My current residence is Austin, Texas 78759.
```

The abbreviation is not expanded within a word. For example, if you type `My current residence` `iscity`, the word iscity is not expanded.

If the **:map!** subcommand is used to map abbreviations for insert mode, then all occurrences of the abbreviations are expanded regardless of where it occurs. If you used the **:map!** subcommand for the preceding example (`:map! city Austin, Texas 78759`), then whenever you type the word `city`, regardless of what precedes or follows, the word will be expanded to `Austin, Texas 78759`. Therefore, the word `iscity` becomes `isAustin, Texas 78759`.

Note: Be careful when choosing the keys that are used for the **:ab** subcommand. Once keys are defined, they can no longer be input as text without first issuing the **:unab** subcommand.

Setting Abbreviations

The **set** command has behavior similar to the **map!** command except that the **set** command substitutes the string for the abbreviation only when the abbreviation is a separate word. You can use the **set** command of the vi editor to:

- List existing abbreviations
- Remove an abbreviation
- Set (define) an abbreviation **Note:** Start the vi editor with an empty editing buffer. Do not name a vi file using the **vi** command or type anything into the buffer after the vi editor starts. Press the Esc key to be sure you are in the command mode.

To list abbreviations
Enter the **:ab** command to list existing abbreviations. Press the Enter key to return to command mode.

To remove abbreviations
Enter the **:anab***Abbreviation* command to remove an abbreviation, where the *Abbreviation* variable specifies the character string you do not want abbreviated any more.

To set (define) an abbreviation
Enter the **:ab** *Abbreviation String* command to set an abbreviation, where the *Abbreviation* variable specifies the character string being defined as an abbreviation and the *String* variable specifies the character string being abbreviated. The abbreviation can be substituted for the string only when the abbreviation is a separate word.

For example, if you enter the **:ab kn upper** command and then type `acknowledge` while in the text input mode, the set abbreviation string is not started because the kn string in the word acknowledge is not a separate word.

However, if you type the **:ab kn upper** command and then type `make the kn line all kncase` while in the text input mode, the result is `make the upper line all uppercase`.

Flags

-c*Subcommand*
Carries out the ex editor subcommand before viewing with **vi** begins. The cursor moves to the line affected by the last subcommand to be carried out. When a null operand is entered, as in **-c' '**, the vi editor places the cursor on the first line of the file. The **-c** flag is incompatible with the **+** flag. Do not specify both flags at the same time.

-l
Enters the vi editor in LISP mode. In this mode, the vi editor creates indents appropriate for LISP code, and the **(**, **)**, **{**, **}**, , and subcommands are modified to act appropriately for LISP.

-r*File*
Recovers a file after a vi editor or system malfunction. If you do not specify the *File* variable, the vi editor displays a list of all saved files.

-R
Sets the **readonly** option to protect the file against overwriting.

-t*Tag*
Edits the file containing the *Tag* variable and positions the vi editor at its definition. To use this flag, you must first create a database of function names and their locations using the **ctags** command.

-v
Enters the vi editor in the verbose mode.

-w*Number*
Sets the default window size to the value specified by the *Number* variable. This flag is useful when you use the vi editor over a low-speed line.

-y*Number*
Overrides the maximum line setting of 1,048,560 with any value greater than 1024. You should request twice the number of lines that you require because the vi editor uses the extra lines for

	buffer manipulation.
+*SubcommandSubcommand*	Carries out the ex editor subcommand before editing begins. If you do not specify the *Subcommand* variable, the cursor is placed on the first line of the file. This **+** flag is incompatible with the **-c** flag. Do not specify both flags at the same time.

vi General Subcommand Syntax

Use the following general syntax to enter subcommands:

Named_Buffer Operator Number Object

Note: Square brackets indicate optional items.

Named_Buffer	Specifies a temporary text storage area.
Operator	Specifies the subcommand or action; instructs the vi editor.
Number	Specifies either the extent of the action or a line address as a whole number.
Object	Specifies what to act on, such as a text object (a character, word, sentence, paragraph, section, character string) or a text position (a line, position in the current line, screen position).

Counts before Subcommands

You can put a number in front of many subcommands. The vi editor interprets this number in one of the following ways:

- Go to the line specified by the *Number* parameter:

```
5G
10Z
```

- Go to the column specified by the *Number* parameter:

```
25|
```

- Scroll the number of lines up or down specified by the *Number* parameter:

```
10Ctrl-U
10Ctrl-D
```

vi Editor Subcommands

Use the subcommands to perform these kinds of actions:

- Moving the cursor
- Editing text
- Manipulating files
- Other actions

Moving the Cursor

Use subcommands to move the cursor within a file in these ways:

- Moving within a line
- Moving within a line by character position
- Moving to words
- Moving by line position
- Moving to sentences, paragraphs, or sections
- Moving by redrawing the screen
- Paging and scrolling
- Searching for patterns
- Marking a specific location in a file and returning

Moving within a Line

Enter the following subcommands in command mode. You can cancel an incomplete command by pressing the Esc key. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax."

Left Arrow or **h** or **Ctrl-H**	Moves the cursor one character to the left.
Down Arrow or **j** or **Ctrl-J** or **Ctrl-N**	Moves the cursor down one line (it remains in the same column).
Up Arrow or **k** or **Ctrl-P**	Moves the cursor up one line (it remains in the same column).
Right Arrow or **l**	Moves the cursor one character to the right.

Moving within a Line by Character Position

Enter the following subcommands in command mode. You can cancel an incomplete command by pressing the Esc key. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax."

^	Moves the cursor to the first nonblank character.
0	Moves the cursor to the beginning of the line.
$	Moves the cursor to the end of the line.
f*x*	Moves the cursor to the next *x* character.
F*x*	Moves the cursor to the last *x* character.
t*x*	Moves the cursor to one column before the next *x* character.
T*x*	Moves the cursor to one column after the last *x* character.
;	Repeats the last **f**, **F**, **t**, or **T** subcommand.
,	Repeats the last **f**, **F**, **t**, or **T** subcommand in the opposite direction.
Number|	Moves the cursor to the specified column.

Moving to Words

Enter the following subcommands in command mode. If you need information about the format of vi subcommands, "vi General Subcommand Syntax."

w	Moves the cursor to the next small word.
b	Moves the cursor to the previous small word.
e	Moves the cursor to the next end of a small word.
W	Moves the cursor to the next big word.
B	Moves the cursor to the previous big word.
E	Moves the cursor to the next end of a big word.

Moving by Line Position

Enter the following subcommands in command mode. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax."

H	Moves the cursor to the top line on the screen.
L	Moves the cursor to the last line on the screen.
M	Moves the cursor to the middle line on the screen.
+	Moves the cursor to the next line at its first nonblank character.
-	Moves the cursor to the previous line at its first nonblank character.
Enter	Moves the cursor to the next line at its first nonblank character.

Moving to Sentences, Paragraphs, or Sections

Enter the following subcommands in command mode. You can cancel an incomplete subcommand by pressing the Esc key. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax."

(	Places the cursor at the beginning of the previous sentence, or the previous s-expression if you are in LISP mode.
)	Places the cursor at the beginning of the next sentence, or the next s-expression if you are in LISP mode.
{	Places the cursor at the beginning of the previous paragraph, or at the next list if you are in LISP mode.
}	Places the cursor at the beginning of the next paragraph, at the next section if you are in C mode, or at the next list if you are in LISP mode.
	Places the cursor at the next section, or function if you are in LISP mode.

Places the cursor at the previous section, or function if you are in LISP mode.

Moving by Redrawing the Screen

Enter the following subcommands in command mode. You can cancel an incomplete subcommand by pressing the Esc key. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax."

z	Redraws the screen with the current line at the top of the screen.
z-	Redraws the screen with the current line at the bottom of the screen.
z.	Redraws the screen with the current line at the center of the screen.
/*Pattern*/**z-**	Redraws the screen with the line containing the character string, specified by the *Pattern* parameter, at the bottom.

Paging and Scrolling

Enter the following subcommands in command mode. You can cancel an incomplete subcommand by pressing the Esc key. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax."

Ctrl-U	Scrolls up one-half screen.
Ctrl-D	Scrolls down one-half screen.
Ctrl-F	Scrolls forward one screen.
Ctrl-B	Scrolls backward one screen.
Ctrl-E	Scrolls the window down one line.
Ctrl-Y	Scrolls the window up one line.
z+	Pages up.
z^	Pages down.

Searching for Patterns

Enter the following subcommands in command mode. You can cancel an incomplete subcommand by pressing the Esc key. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax."

*Number***G**	Places the cursor at the line number specified by the *Number* parameter or at the last line if the *Number* parameter is not specified.
/*Pattern*	Places the cursor at the next line containing the character string specified by the *Pattern* parameter.
?*Pattern*	Places the cursor at the next previous line containing the character string specified by the *Pattern* parameter.
n	Repeats the last search for the text specified by the *Pattern* parameter in the same direction.
N	Repeats the last search for the text specified by the *Pattern* parameter in the opposite direction.
/*Pattern*/**+***Number*	Places the cursor the specified number of lines after the line matching the character string specified by the *Pattern* parameter.
?*Pattern***?-***Number*	Places the cursor the specified number of lines before the line matching the character string specified by the *Pattern* parameter.
%	Finds the parenthesis or brace that matches the one at current cursor position.

Editing Text

The subcommands for editing enable you to perform the following tasks:

- Marking a specific location in a file and returning
- Adding text to a file
- Changing text while in input mode
- Changing text from command mode
- Copying and moving text
- Restoring and repeating changes

Marking a Specific Location in a File and Returning

Enter the following subcommands in command mode. You can cancel an incomplete subcommand by pressing the Esc key. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax."

"	Moves the cursor to the previous location of the current line.
"	Moves the cursor to the beginning of the line containing the previous location of the current line.
m*x*	Marks the current position with the letter specified by the *x* parameter.
`*x*	Moves the cursor to the mark specified by the *x* parameter.
'*x*	Moves the cursor to the beginning of the line containing the mark specified by the *x* parameter.

Adding Text to a File (Text Input Mode)

Enter the following subcommands in command mode to change the vi editor into text input mode. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax."

a*Text*	Inserts text specified by the *Text* parameter after the cursor. End text input mode by pressing the Esc key.
A*Text*	Adds text specified by the *Text* parameter to the end of the line. End text input mode by pressing the Esc key.
i*Text*	Inserts text specified by the *Text* parameter before the cursor. End text input mode by pressing the Esc key.
I*Text*	Inserts text specified by the *Text* parameter before the first nonblank character in the line. End text input mode by pressing the Esc key.
o	Adds an empty line below the current line. End text input mode by pressing the Esc key.
O	Adds an empty line above the current line. End text input mode by pressing the Esc key.

Changing Text While in Input Mode

Use the following subcommands only while in text input mode. These commands have different meanings in command mode. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax."

Ctrl-D	Goes back to previous autoindent stop.
^ Ctrl-D	Ends autoindent for this line only.
0Ctrl-D	Moves cursor back to left margin.
Esc	Ends insertion and returns to command state.
Ctrl-H	Erases the last character.
Ctrl-Q	Enters any character if xon is disabled.
Ctrl-V	Enters any character.
Ctrl-W	Erases the last small word.
\	Quotes the erase and kill characters.
Ctrl-?	Interrupts and ends insert or the Ctrl-D key sequence.

Changing Text from Command Mode

Use the following subcommands in command mode. An incomplete subcommand can be canceled by pressing the Esc key. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax."

C	Changes the rest of the line (same as **c$**).
cc	Changes a line.
cw	Changes a word.
cw*Text*	Changes a word to the text specified by the *Text* parameter.

D	Deletes the rest of the line (same as **d$**).
dd	Deletes a line.
dw	Deletes a word.
J	Joins lines.
r*x*	Replaces the current character with the character specified by *x*.
R *Text*	Overwrites characters with the text specified by the *Text* parameter.
s	Substitutes characters (same as **cl**).
S	Substitutes lines (same as **cc**).
u	Undoes the previous change.
x	Deletes a character at the cursor.
X	Deletes a character before the cursor (same as **dh**).
<<	Shifts one line to the left.
<L	Shifts all lines from the cursor to the end of the screen to the left.
>>	Shifts one line to the right.
>L	Shifts all lines from the cursor to the end of the screen to the right.
~	Changes letter at the cursor to the opposite case.
!	Indents for LISP.

Copying and Moving Text

Use the following subcommands in command mode. An incomplete subcommand can be canceled by pressing the Esc key. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax."

p	Puts back text from the undo buffer after the cursor.
P	Puts back text from the undo buffer before the cursor.
"*x***p**	Puts back text from the *x* buffer.
"*x***d**	Deletes text into the *x* buffer.
y	Places the object that follows (for example, **w** for word) into the undo buffer.
"*x***y**	Places the object that follows into the *x* buffer, where *x* is any letter.
Y	Places the line in the undo buffer.

Restoring and Repeating Changes

Use the following subcommands in command mode. An incomplete subcommand can be canceled by pressing the Esc key. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax."

u	Undoes the last change. **Note:** After an undo, the cursor moves to the first non-blank character on the updated current line.
U	Restores the current line if the cursor has not left the line since the last change.
.	Repeats the last change or increments the **"***n***p** command. **Notes:** 1. This subcommand will repeat the last change, including an undo. Therefore, after an undo, repeat performs an undo rather than repeat the last change. 2. This subcommand is not meant for use with a macro. Enter @@ (two at signs) to repeat a macro.
"*n* **p**	Retrieves the *n*th last delete of a complete line or block of lines.

Manipulating Files

The subcommands for manipulating files allow you to do the tasks outlined in the following sections:

- Saving changes to a file
- Editing a second file

- Editing a list of files
- Finding file information

Saving Changes to a File

Use the following subcommands in command mode. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax."

:w Writes the edit buffer contents to the original file. If you are using this subcommand within the ex editor, you do not need to type the : (colon).

:w *File* Writes the edit buffer contents to the file specified by the *File* parameter. If you are using this subcommand within the ex editor, you do not need to type the : (colon).

:w! *File* Overwrites the file specified by the *File* parameter with the edit buffer contents. If you are using this subcommand within the ex editor, you do not need to type the : (colon).

Editing a Second File

Enter the following subcommands in command mode. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax."

:e *File* Edits the specified file. If you are using this subcommand from the ex editor, you do not need to type the : (colon).

:e! Re-edits the current file and discards all changes.

:e + *File* Edits the specified file starting at the end.

:e + *Number File* Edits the specified file starting at the specified line number.

:e # Edits the alternate file. The alternate file is usually the previous file name before accessing another file with a **:e** command. However, if changes are pending on the current file when a new file is called, the new file becomes the alternate file. This subcommand is the same as the **Ctrl-A** subcommand.

:r *File* Reads the file into the editing buffer by adding new lines below the current line. If you are using this subcommand from the ex editor, you do not need to type the **:** (colon).

:r !*Command* Runs the specified command and places its output into the file by adding new lines below the current cursor position.

:ta *Tag* Edits a file containing the *Tag* tag starting at the location of the tag. To use this subcommand, you must first create a database of function names and their locations using the **ctags** command. If you are using this subcommand from the ex editor, you do not need to type the **:** (colon).

Ctrl- Edits a file containing the tag associated with the current word starting at the location of the tag. To use this subcommand, you must first create a database of function names and their locations using the **ctags** command. Ctrl-T edits a file at the editing position where the previous Ctrl- subcommand was issued. If multiple Ctrl- subcommands have been issued, then multiple Ctrl-T subcommands can be used to return to previous editing positions where Ctrl- subcommands were issued.

Ctrl-A Edits the alternate file. The alternate file is usually the previous current file name. However, if changes are pending on the current file when a new file is called, the new file becomes the alternate file. This subcommand is the same as the **:e #** subcommand.

Editing a List of Files

Enter the following subcommands in command mode. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax."

:n Edits the next file in the list entered on the command line. If you are using this subcommand from the ex editor, a : (colon) is not needed.

:n *Files* Specifies a new list of files to edit. If you are using this subcommand from the ex editor, a **:** (colon) is not needed.

Finding File Information

Enter the following subcommand in command mode. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax".

Ctrl-G	Shows the current file name, current line number, number of lines in the file, and percentage of the way through the file where the cursor is located.

Other Actions

The vi editor provides the subcommands described in the following sections:

- Adjusting the screen
- Entering shell commands
- Interrupting and ending the vi editor

Adjusting the Screen

Enter the following subcommands in command mode. An incomplete subcommand can be canceled by pressing the Esc key. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax."

Ctrl-L	Clears and redraws the screen.
Ctrl-R	Redraws the screen and eliminates blank lines marked with @ (at sign).
z*Number*	Makes the window the specified number of lines long.

Entering Shell Commands

The following subcommands allow you to run a command within the vi editor. Enter these subcommands in command mode. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax."

:sh	Enters the shell to allow you to run more than one command. You can return to the vi editor by pressing the Ctrl-D key sequence. If you are using this subcommand within the ex editor, a : (colon) is not needed.
:!*Command*	Runs the specified command and then returns to the vi editor. If you are using this subcommand within the ex editor, a : (colon) is not needed. **Note:** The # (alternate file), **%** (current file), and **!** (previous command) special characters are expanded when following a **:!** subcommand. To prevent any of these characters from being expanded, use the \ (backslash).
:!!	Repeats the last **:!***Command* subcommand**.**
*Number***!!***Command*	Runs the specified command and replaces the lines specified by *Number* with the output of the command**.** If a number is not specified, the default value is 1. If the command expects standard input, the specified lines are used as input.
!*Object Command*	Runs the specified command and replaces the object specified by the *Object* parameter with the output of the command**.** If the command expects standard input, the specified object is used as input.

Interrupting and Ending the vi Editor

Enter the following subcommands in command mode. If you need information about the format of vi subcommands, see "vi General Subcommand Syntax."

Q	Enters the ex editor in command mode.
ZZ	Exits the vi editor, saving changes.
:q	Quits the vi editor. If you have changed the contents of the editing buffer, the vi editor displays a warning message and does not quit. If you are using this subcommand from the ex editor, a : (colon) is not needed.
:q!	Quits the vi editor, discarding the editing buffer. If you are using this subcommand from the ex editor, a : (colon) is not needed.

Esc	Ends text input or ends an incomplete subcommand.
Ctrl-?	Interrupts a subcommand.

Exit Status

The following exit values are returned:

0	Indicates successful completion.
>0	Indicates an error occurred.

Input Files

Input files must be text files or files that are similar to text files except for an incomplete last line that is no longer than 8191 bytes in length and contains no null characters.

The **.exrc** files must be text files consisting of **ex** commands.

By default, the vi editor reads lines from the files to be edited without interpreting any of those lines as any form of vi editor command.

Related Information

sed command.

viosecure Command

Purpose

Activates, deactivates, and displays security hardening rules. Configures, unconfigures or displays firewall settings.

Syntax

viosecure -level *LEVEL* **-apply** | **-nonint -view**

viosecure -firewall `on` **-force** | `off`

viosecure -firewall `allow` | `deny` **-port** *number* **-interface** *ifname* **-address** *IPaddress* **-timeout** *Timeout* **-remote**

viosecure -firewall `view` **-fmt** *delimiter*

Description

The **viosecure** activates, deactivats, and displays security hardening rules. By default, none of the security hardening features are activated after installation. Upon running the **viosecure** command, the command guides the user through the proper security settings, which range from High to Medium to Low. After this initial selection, a menu is displayed itemizing the security configuration options associated with the selected security level in sets of 10. These options can be accepted in whole, individually toggled off or on, or ignored. After any changes, **viosecure** continues to apply the security settings to the computer system.

The **viosecure** command also configures, unconfigures, and displays network firewall settings. Using the **viosecure** command, you can activate and deactivate specific ports and specify the interface and IP address from which connections will be allowed.

Flags

-apply
Applies all of the LEVEL security settings to the system. There is no user-selectable option.

-firewall `allow` **-port** *Port* **-interface** *ifname* **-address** *IPaddress* **-timeout** *Timeout* **-source**
Permits IP activity per port with optional parameters according to interface, IP address, and time that it is effective. The *Port* argument can be a number or a service name from the **/etc/services** file. The *remote* option specifies that the port is a remote port. All IP activity to and from that remote port is allowed. The default is all IP activity to and from a local port is allowed. The timeout period can be specified as a number (in seconds), or with a number followed by `m`(minutes), `h`(hours), or `d`(days). The maximum timeout period is 30 days.

-firewall `deny` **-port** *Port* **-interface** *Ifname* **-address** *IPaddress* **-timeout** *Timeout* **-source**
Removes a previous firewall **-allow** setting. The *Port* argument can be a number or a service name from the **/etc/services** file. If `-port 0` is specified, then all allow settings are removed. The *remote* option specifies that the port is the remote port. The default is local port. The timeout period can be specified as a number (in seconds), or with a number followed by `m`(minutes), `h`(hours) or `d`(days). The maximum timeout period is 30 days.

-firewall `off`
Unconfigures the default firewall settings.

-firewall `on` **-force**
Configures the default firewall settings from the **/home/ios/security/viosecure.ctl** file. If the **viosecure.ctl** file does not exist, you will have to use the **-force** option to use the default firewall settings.

-level *LEVEL*
Specifies the security LEVEL settings to choose, where LEVEL is `low`, `middle`, `high`, or `default`. The default LEVEL deactivates any previous security LEVEL system settings. Except for the default LEVEL, ten security LEVEL settings are displayed at a time. The user then can choose the desired security settings by entering comma separated numbers, the word `ALL` to choose all of the settings, the word `NONE` to choose none of the settings, the letter `q` to exit, or the letter `h` for help. The security settings chosen are

then applied to the system.

-firewall *view* **-fmt** *delimiter*
Displays the current allowable ports. If the **-fmt** option is specified, then it divides output by a user-specified delimiter.

-nonint
Specifies non-interactive mode.

-view
Displays the current security level settings. All of the security setting names end with three characters `Xls` where X = `l`(low), `m`(medium), `h`(high) or `d`(default). For example, the security level name `minlenlls` is the low level security setting for minimum length of a password.

Examples

1. To display the high system security settings, and to select which of the high security settings to apply to the system, type:

   ```
   viosecure -level high
   ```
2. To apply all of the 'high' system security settings to the system, type:

   ```
   viosecure -level high -apply
   ```
3. To the display the current system security settings, type:

   ```
   viosecure -view
   ```
4. To unconfigure the previous system security settings, type:

   ```
   viosecure -level default
   ```
5. To allow IP activity on the ftp-data, ftp, ssh, www, https, rmc, and cimon ports, and to deny other IP activity, type:

   ```
   viosecure -firewall on
   ```
6. To allow IP activity on all ports, type:

   ```
   viosecure -firewall off
   ```
7. To allow users from IP address 10.10.10.10 to rlogin, type:

   ```
   viosecure -firewall allow -port login -address 10.10.10.10
   ```
8. To allow users to rlogin for seven days, type:

   ```
   viosecure -firewall allow -port login -timeout 7d
   ```
9. To allow rsh client activity through interface en0, type:

   ```
   viosecure -firewall allow -port 514 -interface en0 -remote
   ```
10. To removes the rule that allows users from IP address 10.10.10.10 to rlogin, type:

    ```
    viosecure -firewall deny -port login -address 10.10.10.10
    ```
11. To display the list of allowed ports, type:

    ```
    viosecure -firewall view
    ```

viostat Command

Purpose

Reports Central Processing Unit (CPU) statistics, asynchronous input/output (AIO) and input/output statistics for the entire system, adapters, tty devices, disks and CD-ROMs.

Syntax

viostat

viostat **-sys** **-adapter** **-tty** | **-disk** **-path** **-time** *PhysicalVolume* ... *Interval Count*

Description

The viostat command is used for monitoring system input/output device loading by observing the time the physical disks are active in relation to their average transfer rates. The viostat command generates reports that can be used to change system configuration to better balance the input/output load between physical disks and adapters.

The first report generated by the **viostat** command provides statistics concerning the time since the system was booted. Each subsequent report covers the time since the previous report. All statistics are reported each time the **viostat** command is run. The report consists of a tty and CPU header row followed by a row of tty and CPU statistics. On multiprocessor systems, CPU statistics are calculated system-wide as averages among all processors.

If the -**sys** flag is specified, a system-header row is displayed followed by a line of statistics for the entire system. The hostname of the system is printed in the system-header row.

If the -**adapter** flag is specified, an adapter-header row is displayed followed by a line of statistics for the adapter. This will be followed by a disk-header row and the statistics of all the disks/CD-ROMs connected to the adapter. Such reports are generated for all the disk adapters connected to the system.

A disks header row is displayed followed by a line of statistics for each disk that is configured. If the PhysicalVolume parameter is specified, only those names specified are displayed.

If the PhysicalVolume parameter is specified, one or more alphabetic or alphanumeric physical volumes can be specified. If the PhysicalVolume parameter is specified, the tty and CPU reports are displayed and the disk report contains statistics for the specified drives. If a specified drive name is not found, the report lists the specified name and displays the message Drive Not Found. If no Drive Names are specified, the report contains statistics for all configured disks and CD-ROMs. If no drives are configured on the system, no disk report is generated. The first character in the PhysicalVolume parameter cannot be numeric.

The Interval parameter specifies the amount of time in seconds between each report. The first report contains statistics for the time since system startup (boot). Each subsequent report contains statistics collected during the interval since the previous report. The Count parameter can be specified in conjunction with the Interval parameter. If the Count parameter is specified, the value of count determines the number of reports generated at Interval seconds apart. If the Interval parameter is specified without the Count parameter, the viostat command generates reports continuously.

The **viostat** command reports number of physical processors consumed (physc) and the percentage of entitlement consumed (% entc) in shared processor environments. These metrics will only be displayed in shared processor environments.

The **viostat** command is useful in determining whether a physical volume is becoming a performance bottleneck and if there is potential to improve the situation. The % utilization field for the physical volumes indicates how evenly the file activity is spread across the drives. A high % utilization on a physical volume is a good indication that there may be contention for this resource. Since the CPU utilization statistics are also available with the viostat report, the percentage of time the CPU is in I/O wait can be determined at the same time. Consider distributing data across drives if the I/O wait time is significant and the disk utilization is not evenly distributed across volumes.

Reports

The viostat command generates four types of reports, the tty and CPU Utilization report, the Disk Utilization report, the System throughput report and the Adapter throughput report.

tty and CPU Utilization Report

The first report generated by the viostat command is the tty and CPU Utilization Report. For multiprocessor systems, the CPU values are global averages among all processors. Also, the I/O wait state is defined system-wide and not per processor. The report has the following format:

Statistic	Description
tin	Shows the total number of characters read by the system for all ttys.
tout	Shows the total number of characters written by the system to all ttys.
% user	Shows the percentage of CPU utilization that occurred while executing at the user level (application).
% sys	Shows the percentage of CPU utilization that occurred while executing at the system level (kernel).
% idle	Shows the percentage of time that the CPU or CPUs were idle and the system did not have an outstanding disk I/O request.
% iowait	Shows the percentage of time that the CPU or CPUs were idle during which the system had an outstanding disk I/O request.

This information is updated at regular intervals by the kernel (typically sixty times per second). The tty report provides a collective account of characters per second received from all terminals on the system as well as the collective count of characters output per second to all terminals on the system.

Disk Utilization Report

The second report generated by the viostat command is the Disk Utilization Report. The disk report provides statistics on a per physical disk basis. The report has a format similar to the following:

Table 2.

Statistic	Description
% tm_act	Indicates the percentage of time the physical disk was active (bandwidth utilization for the drive).
Kbps	Indicates the amount of data transferred (read or written) to the drive in KB per second.
tps	Indicates the number of transfers per second that were issued to the physical disk. A transfer is an I/O request to the physical disk. Multiple logical requests can be combined into a single I/O request to the disk. A transfer is of indeterminate size.
Kb_read	The total number of KB read.
Kb_wrtn	The total number of KB written.

Statistics for CD-ROM devices are also reported.

For large system configurations where a large number of disks are configured, the system can be configured to avoid collecting physical disk input/output statistics when the **viostat** command is not executing. If the system is configured in the above manner, the first Disk report displays the message Disk History Since Boot Not Available instead of the disk statistics. Subsequent interval reports generated by the **viostat** command contain disk statistics collected during the report interval. Any tty and CPU statistics after boot are unaffected.

System Throughput Report

This report is generated if the -sys flag is specified. This report provides statistics for the entire system. This report has the following format:

Statistic	Description
Kbps	Indicates the amount of data transferred (read or written) in the entire system in KB per second.

`tps`	Indicates the number of transfers per second issued to the entire system.
`Kb_read`	The total number of KB read from the entire system.
`Kb_wrtn`	The total number of KB written to the entire system.

Adapter Throughput Report

This report is generated if the **-adapter** flag is specified. This report provides statistics on an adapter-by-adapter basis. This report has the following format:

Statistic	Description
`Kbps`	Indicates the amount of data transferred (read or written) in the adapter in KB per second.
`tps`	Indicates the number of transfers per second issued to the adapter.
`Kb_read`	The total number of KB read from the adapter.
`Kb_wrtn`	The total number of KB written to the adapter.

Disk Input/Output History

To improve performance, the collection of disk input/output statistics has been disabled. To enable the collection of this data, type:

```
chdev -dev sys0 -attr iostat=true
```

To display the current settings, type:

```
lsdev -dev sys0 -attr iostat
```

If the collection of disk input/output history is disabled, the first disk report of **viostat** output displays the message `Disk History Since Boot Not Available` instead of disk statistics. As before, subsequent interval reports generated by the **viostat** command contain disk statistics collected during the report interval.

Flags

-adapter
Displays the adapter throughput report.

If the **-adapter** flag is specified with the **-tty** flag, the tty and CPU report is displayed, followed by the adapter throughput report. Disk Utilization reports of the disks connected to the adapters, will not be displayed after the Adapter throughput report.

If the **-adapter** flag is specified with the **-disk** flag, tty and CPU report will not be displayed. If Physical Volume parameter is specified, the Disk Utilization Report of the specified Physical volume will be printed under the corresponding adapter to which it belongs.

-disk
The **-disk** flag is exclusive of the **-tty** flag and displays only the disk utilization report.

-path
The **-m** flag will print the path statistics for the following

- Paths to MPIO (Multi-Path I/O) enabled devices.
- Paths in the ESS machines.

The throughput is per device. The throughout for all the paths to that device follow the throughput of that device.

For ESS machines, the vpaths will be treated as disks and hdisks will be treated as Paths. Internally the vpaths are actually disks and hdisks are the paths to them. For MPIO enabled devices, the path name will be represented as Path0, Path1, Path2 and so on. The numbers 0, 1, 2, and so on are the path IDs provided by the lspath command. Since paths to a device can be attached to any adapter, the adapter report will report the path statistics under each adapter. The disk name will be a prefix to all the paths. For all MPIO enabled devices, the adapter report will print the path names as hdisk10_Path0, hdisk0_Path1, and so on. For all ESS Machines, the adapter report will print the path names as vpath0_hdisk3, vpath10_hdisk25, and so on.

-sys
Displays the system throughput report.

-time
Prints the time-stamp next to each line of output of viostat. The time-stamp displays in the HH:MM:SS format.

-tty
The **-tty** flag is exclusive of the **-disk** flag and displays only the tty and cpu usage reports.

Examples

1. To display a single history since boot report for all tty, CPU, and Disks, type:

   ```
   viostat
   ```
2. To display a continuous disk report at two second intervals for the disk with the logical name disk1, type:

   ```
   viostat -disk disk1 2
   ```
3. To display six reports at two second intervals for the disk with the logical name disk1, type:

   ```
   viostat disk1 2 6
   ```
4. To display six reports at two second intervals for all disks, type:

   ```
   viostat -disk 2 6
   ```
5. To display six reports at two second intervals for three disks named disk1, disk2, disk3, type:

   ```
   viostat disk1 disk2 disk3 2 6
   ```
6. To print the System throughput report, type:

   ```
   viostat -sys
   ```
7. To print the Adapter throughput report, type:

   ```
   viostat -adapter
   ```
8. To print the System and Adapter throughput reports, with only the tty and CPU report (no disk reports), type:

   ```
    viostat -sys -adapter -tty
   ```
9. To print the System and Adapter throughput reports with the Disk Utilization reports of hdisk0 and hdisk7, type

   ```
   viostat -sys -adapter -disk hdisk0 hdisk7
   ```
10. To display time stamp next to each line of output of viostat, type:

    ```
    viostat -time
    ```

wall Command

Purpose

Writes a message to all users that are logged in.

Syntax

wall -a -g *Group Message*

Description

The **wall** command writes a message to all users that are logged in. If the *Message* parameter is not specified, the **wall** command reads the message from standard input until it reaches an end-of-file character. The message is then sent to all logged in users. The following heading precedes the message:

```
Broadcast message from
user@node

 (tty) at hh:mm:ss ...
```

`hh:mm:ss` represents the hours, minutes, and seconds when the message was sent.

To override any protections set up by other users, you must operate with root user authority. Typically, the root user uses the **wall** command to warn all other users of an impending system shutdown.

Notes:

1. The **wall** command only sends messages to the local node.
2. Messages can contain multibyte characters.

Flags

-a	Performs the default operation. This flag is provided for System V compatibility. It broadcast messages to the console and pseudo-terminals.
-g *Group*	Broadcasts to a specified group only.

Files

/dev/tty	Specifies a device.

wc Command

Purpose

Counts the number of lines, words, bytes, or characters in a file.

Syntax

wc **-c** | **-m** **-l** **-w** *File ...*

wc **-k** **-c** **-l** **-w** *File ...*

Description

By default, the **wc** command counts the number of lines, words, and bytes in the files specified by the *File* parameter. The command writes the number of newline characters, words, and bytes to the standard output and keeps a total count for all named files.

When you use the *File* parameter, the **wc** command displays the file names as well as the requested counts. If you do not specify a file name for the *File* parameter, the **wc** command uses standard input.

The **wc** command is affected by the **LANG**, **LC_ALL**, **LC_CTYPE**, and **LC_MESSAGES** environment variables.

The **wc** command considers a word to be a string of characters of non-zero length which are delimited by a white space (for example SPACE , TAB).

Flags

-c Counts bytes unless the **-k** flag is specified. If the **-k** flag is specified, the **wc** command counts characters.

-k Counts characters. Specifying the **-k** flag is equivalent to specifying the **-klwc** flag. If you use the **-k** flag with other flags, then you must include the **-c** flag. Otherwise, the **-k** flag is ignored. For more information, see examples 4 and 5.

Note: This flag is to be withdrawn in a future release.

-l Counts lines.

-m Counts characters. This flag cannot be used with the **-c** flag.

-w Counts words. A word is defined as a string of characters delimited by spaces, tabs, or newline characters.

Note: If no flag is specified, **wc** by default counts the lines, words, bytes in a file or from standard input.

Exit Status

This command returns the following exit values:

0	The command ran successfully.
>0	An error occurred.

Examples

1. To display the line, word, and byte counts of a file, enter:

```
wc chap1
```

The **wc** command displays the number of lines, words, and bytes in the `chap1` file.

2. To display only byte and word counts, enter:

   ```
   wc -cw chap*
   ```

   The **wc** command displays the number of bytes and words in each file that begins with `chap`. The command also displays the total number of bytes and words in these files.
3. To display the line, word, and character counts of a file, enter:

   ```
   wc -k chap1
   ```

   The **wc** command displays the number of lines, words, and characters in the `chap1` file.
4. To display the word and character counts of a file, enter:

   ```
   wc -kcw chap1
   ```

   The **wc** command displays the number of characters and words in the `chap1` file.
5. To use the **wc** command on standard input, enter:

   ```
   wc -klw
   ```

   The **wc** command displays the number of lines and words in standard input. The **-k** flag is ignored.
6. To display the character counts of a file, enter:

   ```
   wc -m chap1
   ```

   The **wc** command displays the number of characters in the `chap1` file.
7. To use the **wc** command on standard input, enter:

   ```
   wc -mlw
   ```

   The **wc** command displays the number of lines, words, and characters in standard input.

Files

/usr/bin/wc, /bin/wc	Contains the **wc** command.
/usr/ucb/wc	Contains the symbolic link to the **wc** command.

who Command

Purpose

Identifies the users currently logged in.

Syntax

who -a | **-b -d -i -l -m -p -q -r -s -t -u -w -A -H -T -X** *File*

who am { **i** | **I** }

Description

The **who** command displays information about all users currently on the local system. The following information is displayed: login name, tty, date and time of login. Typing `who am i` or `who am I` displays your login name, tty, date and time you logged in. If the user is logged in from a remote machine, then the host name of that machine is displayed as well.

The **who** command can also display the elapsed time since line activity occurred, the process ID of the command interpreter (shell), logins, logoffs, restarts, and changes to the system clock, as well as other processes generated by the initialization process.

The general output format of the **who** command is as follows:

```
Name State Line Time Activity Pid Exit (Hostname)
```

where:

`Name` Identifies the user's login name.

`State` Indicates whether the line is writable by everyone (see the **-T** flag).

`Line` Identifies the line name as found in the /**dev** directory.

`Time` Represents the time when the user logged in.

`Activity` Represents the hours and minutes since activity last occurred on that user's line. A . (dot) here indicates line activity within the last minute. If the line has been quiet more than 24 hours or has not been used since the last system startup, the entry is marked as old.

`Pid` Identifies the process ID of the user's login shell.

`Term` Identifies the process termination status (see the **-d** flag).

`Exit` Identifies the exit status of ended processes (see the **-d** flag).

`Hostname` Indicates the name of the machine the user is logged in from.

To obtain information, the **who** command usually examines the /**etc**/**utmp** file. If you specify another file with the *File* parameter, the **who** command examines that file instead. This new file is usually the /**var**/**adm**/**wtmp** or /**etc**/**security**/**failedlogin** file.

If the *File* parameter specifies more than one file name, only the last file name will be used.

Note: This command only identifies users on the local node.

Flags

-a Processes the /**etc**/**utmp** file or the named file with all information. Equivalent to specifying the **-bdlprtTu** flags.

-b Indicates the most recent system startup time and date.

-d

Displays all processes that have expired without being regenerated by init. The exit field appears for dead processes and contains the termination and exit values (as returned by wait) of the dead process. (This flag is useful for determining why a process ended by looking at the error number returned by the application.)

-l Lists any login process.

-m Displays information about the current terminal only. The **who -m** command is equivalent to the **who am i** and **who am I** commands.

-p Lists any active process that is currently active and has been previously generated by init.

-q Prints a quick listing of users and the number of users on the local system.

-r Indicates the current run-level of the process.

-s Lists only the name, line, and time fields. This flag is the default; thus, the **who** and **who -s** commands are equivalent.

-t Indicates the last change to the system clock by the root user using the **date** command. If the **date** command has not been run since system installation, the **who -t** command produces no output.

-u or **-i** Displays the user name, tty, login time, line activity, and process ID of each current user.

-A Displays all accounting entries in the /**etc**/**utmp** file.

-H Displays a header (title).

-T or **-w** Displays the state of the tty and indicates who can write to that tty as follows:

+ Writable by anyone.

- Writable only by the root user or its owner.

? Bad line encountered.

-X Prints all available characters of each user name instead of truncating to the first 8 characters. The user name is also moved to the last column of the output.

Exit Status

This command returns the following exit values:

0	Successful completion.
>0	An error occurred.

Examples

1. To display information about who is using the local system node, type:

   ```
   who
   ```

   Information similar to the following is displayed:

   ```
   pts/1      Nov  9 00:20   long_username_greater_than_eight_characters  (localhost)
   ```
2. To display your user name, type:

   ```
   who am i
   ```

   Information similar to the following is displayed:

   ```
   george lft/0 Jun 8 08:34
   ```
3. To display a history of logins, logouts, system startups, and system shutdowns, type:

   ```
   who /var/adm/wtmp
   ```

   Information similar to the following is displayed:

   ```
   hank   lft/0   Jun   8   08:34  (ausnix5)
   john   lft/0   Jun   8   08:34  (JIKey)
   mary   lft/0   Jun   8   08:22  (machine.austin.ibm)
   ```

```
jan    pts4    Jun   8   09:19  (puff.wisc.edu)
```

4. To display the run-level of the local system node, type: `who -r`
   Information similar to the following is displayed:

```
. run-level 2 Jun 8 04:15 2 0 s
```

5. To display any active process that is currently actively and has been previously generated by init, type: `who -p`
   Information similar to the following is displayed:

```
srcmstr   .    Jun 8  04:15  old  2896
cron      .    Jun 8  04:15  old  4809
uprintfd  .    Jun 8  04:15  old  5158
```

6. To process the **/var/adm/wtmp** file with the **-bdlprtTu** flags specified, type: `who -a /var/adm/wtmp`
   Information similar to the following is displayed:

```
   .        system boot Jun 19 10:13
   .        run-level 2 Jun 19 10:13
   .            .       Jun 19 10:14    old
   .            .       Jun 19 10:14    old
   .            .       Jun 19 10:14    old
rc        -     .       Jun 19 10:13    old
   .            .       Jun 19 10:16    old
   .            .       Jun 19 10:14    old
srcmstr   -     .       Jun 19 10:14    old
rctcpip   -     .       Jun 19 10:14    old
rcdce     -     .       Jun 19 10:14    old
rccm      -     .       Jun 19 10:15    old
dceupdt   -     .       Jun 19 10:15    old
rcnfs     -     .       Jun 19 10:15    old
cron      -     .       Jun 19 10:16    old
piobe     -     .       Jun 19 10:16    old
qdaemon   -     .       Jun 19 10:16    old
writesrv  -     .       Jun 19 10:16    old
uprintfd  -     .       Jun 19 10:16    old
   .            .       Jun 19 10:16    old
LOGIN     - lft0        Jun 19 10:16    old
   .            .       Jun 19 10:16    old
   .            .       Jun 19 10:16    old
```

Files

/etc/utmp	Contains user and accounting information.
/etc/security/failedlogin	Contains the history of all invalid logins.
/var/adm/wtmp	Contains the history of all logins since the file was last created.
/usr/include/sys/signal.h	Contains a list of termination values.

Related Information

The **date** command.

wkldagent Command

Purpose

Starts, stops, or queries the state of the Workload Manager Agent.

Syntax

wkldagent **-start** | **-status** | **-stop**

Description

The **wkldagent** command starts, stops, and queries the state of the Workload Manager Agent. The Workload Manager Agent provides recording capability for a limited set of local system performance metrics. These include common CPU, memory, network, disk, and partition metrics typically displayed by the **topas** command.

The Workload Manager must be started using the **wkldmgr** command before the **wkldagent** command is run. Daily recordings are stored in the **/home/ios/perf/wlm** directory with filenames `xmwlm.`*YYMMDD*, where YY is the year, MM is the month, and DD is the day. The **wkldout** command can be used to process Workload Manager-related recordings. All recordings cover 24-hour periods and are retained for only two days.

Flags

-start	Starts the Workload Manager Agent.
-status	Displays the state of the Workload Manager Agent, either running or stopped.
-stop	Stops the Workload Manager Agent.

Exit Status

0	The command completed successfully
>0	An error occurred.

Examples

1. To start the Workload Manager Agent, type:

   ```
   wkldmgr -start
   ```
2. To check whether the Workload Manager Agent is currently active, type:

   ```
   wkldmgr -status
   ```
3. To stop the Workload Manager Agent, type:

   ```
   wkldmgr -stop
   ```

Related Information

The **topas** command, the **wkldmgr** command, and the **wkldout** command.

wkldmgr Command

Purpose

Starts or stops Workload Manager.

Syntax

wkldmgr **-start** | **-status** | **-stop**

Description

The **wkldmgr** command starts, stops, and queries the state of the Workload Manager. Starting the Workload Manager is necessary for the **-cecdisp** option of the **topas** command to work properly.

Flags

-start Starts the Workload Manager.

-status Displays the state of the Workload Manager, either running or stopped.

-stop Stops the Workload Manager.

Exit Status

0	The command completed successfully
>0	An error occurred.

Examples

1. To start the Workload Manager, type:

   ```
   wkldmgr -start
   ```
2. To check whether the Workload Manager is currently active, type:

   ```
   wkldmgr -status
   ```
3. To stop the Workload Manager, type:

   ```
   wkldmgr -stop
   ```

Related Information

The **topas** command, the **wkldagent** command, and the **wkldout** command.

wkldout Command

Purpose

Provides post-processing of the recordings made by the Workload Manager Agent (**wkldagent**).

Syntax

wkldout -filename *file*

Description

The **wkldout** command processes the data produced by running the Workload Manger Agent. The Workload Manager Agent writes data files to the /**home**/**ios**/**perf**/**wlm** directory.

Flags

-filename	Specifies the data file to process.

Exit Status

0	The command completed successfully
>0	An error occurred.

Examples

1. To process a data file named xmwlm.060331, type:

```
wkldout -filename /home/ios/perf/wlm/xmwlm.0600331
```

Related Information

The **topas** command, the **wkldmgr** command, and the **wkldagent** command.

Technical publication remarks form

Title :	ESCALA POWER5 Hardware Information Virtual I/O Server commands

Reference N° :	86 A1 45EW 00	**Date:**	July 2006

ERRORS IN PUBLICATION

SUGGESTIONS FOR IMPROVEMENT TO PUBLICATION

Your comments will be promptly investigated by qualified technical personnel and action will be taken as required.
If you require a written reply, please include your complete mailing address below.

NAME : ______________________________ Date : __________

COMPANY : ______________________________

ADDRESS : ______________________________

____

Please give this technical publication remarks form to your BULL representative or mail to:

Bull - Documentation D$^{ept.}$
1 Rue de Provence
BP 208
38432 ECHIROLLES CEDEX
FRANCE
info@frec.bull.fr

Technical publications ordering form

To order additional publications, please fill in a copy of this form and send it via mail to:

BULL CEDOC **357 AVENUE PATTON** **B.P.20845** **49008 ANGERS CEDEX 01** **FRANCE**	**Phone:** **FAX:** **E-Mail:**	+33 (0) 2 41 73 72 66 +33 (0) 2 41 73 70 66 srv.Duplicopy@bull.net

CEDOC Reference #	Designation	Qty
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
[_ _] : The latest revision will be provided if no revision number is given.		

NAME: ______________________________ Date:__________

COMPANY:______________________________

ADDRESS: ______________________________

____

PHONE: ______________________ FAX: ______________

E-MAIL: ______________________________

For Bull Subsidiaries:

Identification: ______________________________

For Bull Affiliated Customers:

Customer Code: ______________________________

For Bull Internal Customers:

Budgetary Section: ______________________________

For Others: Please ask your Bull representative.

BULL CEDOC
357 AVENUE PATTON
B.P.20845
49008 ANGERS CEDEX 01
FRANCE

REFERENCE
86 A1 45EW 00