AIX Questions

Compiled by Jeff Simon

The AIX Solution Provider Technical Support Group in Austin, Texas, supports software vendors who are developing or porting applications to AIX. This article is a compilation of questions that are frequently asked by vendors. The name of the responding Technical Support Group staff member appears after each response.

How do I configure an anonymous ftp site in AIX?

AIX® provides a sample script to (minimally) configure the RISC System/6000® as an anonymous ftp server. You can find the script in `/usr/lpp/tcpip/samples/anon.ftp` (AIX 3.2.5) or `/usr/samples/tcpip/anon.ftp` (AIX 4.x). Follow the steps below to configure the RS/6000™ as an anonymous ftp server.

1. Execute the `anon.ftp` script, which will create the ftp and anonymous users. These users will have different User IDs (UIDs) but the same home directory. In addition, the script will create five subdirectories (`etc`, `pub`, `bin`, `lib`, and `usr`) under the ftp user's home directory, each owned by root. This prevents anonymous ftp users from making any changes. The `etc`, `bin`, `lib`, and `usr` directories are designed for ftp use; therefore, they have world-readable (`r`) and world-searchable (`x`) permission only. The `pub` directory also has world-writeable (`w`) permission to allow file uploads. Executing this script is sufficient to provide a minimal ftp server.

   Now, when a user ftps to the RS/6000 and specifies the anonymous or ftp user name, the ftp daemon automatically accepts any password. However, the ftp daemon also automatically executes the `chroot` command on the ftp user's home directory. This restricts an anonymous ftp user to the ftp user's home directory structure. Thus, anonymous ftp users can only access (that is, `cd` and `ls`) a limited directory structure, not the entire directory structure of the machine. Although the `anon.ftp` script provides a minimal setup, you may wish to perform some additional configuration.

2. Remove the `~ftp/.profile` created by the system for the ftp user.

3. Create a `passwd` and `group` file in the ftp user's `etc` directory. This will allow anonymous ftp users to see names instead of numbers for file permissions. The `passwd` file should look like this (assuming 302 is the ftp user's UID):

```
ftp:*:302:1:anonymous ftp user:/u/ftp:/bin/false
root:*:0:0::/:/bin/false
```

   The group file should look like this:

```
system:*:0:
staff:*:1:
```

Then, execute the `chmod 400 *` command in the `~ftp/etc` directory.

4. Enable logging on the ftp daemon, which allows you to record the hostname and e-mail address (anonymous password) of anonymous ftp users, as well as which files they upload and download. To enable logging, do the following:

   - Execute the `smit inetdconf` command (as root).
   - Select Change/Show Characteristics of an `inetd` Subserver.
   - Select ftp.
   - Change the Service Program command-line `ARGUMENTS` field to read `ftpd -l`, then press Enter.
   - Edit the `/etc/syslog.conf` file and add the line `daemon.info /tmp/ftp.log`.
   - Execute the command `touch /tmp/ftp.log`.
   - Execute the command `kill -1 <syslogd pid>`, where `<syslogd pid>` is the Process ID (PID) of the `syslogd` daemon. This will cause the `syslogd` daemon to re-read its configuration file.

5. Create a separate directory for uploads, then execute the following commands:

   - `cd ~ftp`
   - `mkdir incoming`
   - `chmod 777 incoming` (to allow uploads)
   - `chmod 555 pub` (to protect your outgoing repository from uploads)

—Jeff Simon

Does Base Operating System (BOS) 4.2 have a mechanism that will automatically export global functions from my shared libraries without generating an export list?

Yes, this can be done by linking with the `-bexpall` flag. The `-bexpall` flag exports all global symbols, except imported symbols, unreferenced symbols defined in archive members, and symbols beginning with an underscore (_). Additional symbols can be exported by listing them in an export file.

By using this option, you can avoid using an export file. On the other hand, an export file provides explicit control over which symbols are exported. It allows you to use other global symbols within the shared object without worrying about conflict with names exported from other shared objects.

See InfoExplorer™ (or man pages) under `ld` (look under Options) for additional information about `-bexpall`.

Jeff Simon

—Jeff Simon

I built my database application on BOS 4.1.5 and get the following error messages when running on BOS 4.3:

```
Could not load program ...
Symbol kaio_rdwr in /usr/lib/libc.a is undefined
Symbol listio in /usr/lib/libc.a is undefined
Symbol acancel in /usr/lib/libc.a is undefined
Symbol iosuspend in /usr/lib/libc.a is undefined
Could not load library libc.a[aio.o]
Error was: Exec format error
```

What is causing this problem?

You need to make asynchronous I/O available on your BOS 4.3 system by doing the following (as root):

```
<smit/smitty> chgaio
 State to be configured at system restart
```

Note: You can use the tab key to toggle from defined to available. Next, you will need to reboot the system to initiate any changes.

—Jeff Simon

How can I tell which APARs were installed on my system?

You can do this by entering one of the following commands:

```
instfix -ik <APAR#>      // query individual APAR#
  i.e. instfix -ik IX71075
```

or

```
instfix -iv | grep IX    // query all APAR #'s
```

—Jeff Simon

What does firmware do on RS/6000 machines?

Firmware is the first code executed when the machine is powered-on. Firmware takes the system from a power-on state to a state where an operating system loader is in memory, ready for execution. Firmware is typically stored in read-only memory (ROM) or in programmable read-only memory (PROM).

The firmware on PCI-based RS/6000 systems performs the following steps:

- Initializes processor registers
- Initializes the memory controller
- Establishes an active RAM area
- Copies decompressed code to RAM
- Decompresses the compressed area into RAM
- Establishes execution environment (stacks, and so on)
- Initializes the console
- Displays graphics/logo on console
- Initializes individual subsystems
- Locates and loads the operating system's boot code

The PCI-based RS/6000 system's firmware also provides functions that were already incorporated in the firmware for the Micro Channel®-based RS/6000 systems. One of them is Power-On Self Test (POST). The POST checks basic hardware, such as processor, native I/O, and system memory, and makes a list of the working hardware it recognizes.

When the system is booted, the firmware obtains the boot device list maintained in NVRAM and tries to locate the first valid boot device that has a valid boot image on it (this is determined by reading the first 512 bytes on a device). When a valid boot image has been found, it is loaded into memory and firmware passes control to it. The firmware does not have to be aware of the type of code it loads—Software Read-Only Software (ROS) or the kernel of an operation system. For AIX, it is Software ROS.

Wade Carlin

—Wade Carlin

What hardware platforms does AIX Version 4 support and how does this affect boot images?

AIX Version 4 provides support for the following three RS/6000 hardware platforms:

- Micro Channel
- Symmetric Multiprocessor (SMP)
- RSPC (PCI-based)

This AIX boot image consists of the following components:

- AIX kernel
- RAM
- File system
- Base customized device information

The Software ROS loads the AIX boot image. To create platform-specific boot images for AIX, the software must first recognize the platform type that is specified in the IPL control block built by Software ROS.

The boot utility command, `bootinfo -T`, inspects the IPL control block and prints the platform type to the standard output. The command `bosboot -T` generates the boot image for the target platform. If the platform type is not specified, this command defaults to the current platform type on which the system runs.

—Wade Carlin

```
$ ls -al
drwxr-xr-x 2 brenda staff 8192 Aug 26 13:45 .
drwxr-x-- 9 brenda staff 2048 Aug 26 12:03 ..
-rw-r--r- 1 brenda staff  695 Aug 26 13:45 file1
-rwxr-xr-x 1 brenda staff  695 Aug 26 13:45 script.ksh
```

Figure 1. Permissions assigned per file/directory

Brenda Hagler

Nilesh Gandhi

How can I set the default permissions on files that I create to make them read-only?

When you create a file, the umask defines the read, write, and execute permissions for the file. To set the permissions on the files to be created so that the default is read-only for group and others, use the umask of 033. To see what umask you are currently using, issue the `umask -S` command.

If you put the umask in the `/etc/profile` file, users on your system will create files using that umask. They can define their own umask by putting the umask assignment in their `$HOME/.profile`. The umask in the users `.profile` will take precedence over any umask defined in `/etc/profile`. The command `ls -al` will display the permissions assigned per file or directory. Figure 1 shows an example.

Umask Number	Permissions
umask 002	Files and directories will be created WITHOUT write permission for others.
umask 077	Files and directories will be created WITHOUT read, write, or execute permission for group or others.
umask 027	Files and directories will be created WITHOUT write permission for group and WITHOUT read, write, or execute permissions for others.

Figure 2. The umask permissions

Reading the permissions from left to right, the first read, write, execute (`rwx`) permissions belong to the owner. In the above example, the owner is brenda. The second set of `rwx` permissions belong to the group staff in this case. The third set of permissions are for other users. Figure 2 shows examples of umask numbers and their permissions.

Note: The umask number defines which permissions should NOT be given.

For more information, see the `umask` command in the *AIX Commands Reference* (SC23-2639-03). "System Startup Files Overview" in InfoExplorer reviews the `/etc/profile` versus a user's `.profile`.

—Brenda Hagler

How can sed be used to insert a new line?

Figure 3 illustrates how `sed` can substitute a new line for any pattern, using either a line-feed or a character sequence.

Method One

```
% sh> sed 's/ /\ (press return)
> /g' <fileName>
```

Method Two

```
% sh> sed 's/ /\^J/g' <fileName>
i.e. ^J can be created while holding control key
     and pressing v and j keys simultaneously.
```

Figure 3. Inserting a new line using sed

—Nilesh Gandhi

Why do I get a core dump when trying to read 'sed -f <fileName>' from a command line?

This problem may occur if the input file contains more than 300 lines of text or if it has over 32,000 characters. When the buffer is overwritten, the system detects a memory fault and causes the `sed` program to core dump.

—Nilesh Gandhi

***Jeff Simon**, IBM Corporation, 11400 Burnet Road, Austin, TX 78758. Mr. Simon has worked in technical support since 1990 and currently works as a technical lead and systems administrator for a series of technical Web sites. He has a BS in Computer Science from Southwest Texas State University.*