15.9 cpio: A BACKUP PROGRAM

The importance of performing regular backups isn't usually appreciated until a crash has occurred and a lot of data has been lost. As administrator, you are partly responsible for the safety of the data that resides on the system. It is part of your duties to decide which files should be backed up and also to determine the periodicity of such backups. The effectiveness of the backup is determined by your ability to easily restore lost or corrupted data files. For reasons of security, backup media of sensitive data are often kept at distant locations.

We'll consider two backup programs in this chapter—**cpio** and **tar**. Both combine a group of files into an archive *(5.14)*, with suitable headers preceding the contents of each file. The backup device can be a magnetic or a cartridge tape, a floppy diskette, or even a disk file. Small systems, especially workstations, may not have the tape facility, so the floppy drive will be used here to illustrate the features of both commands.

The **cpio** command (copy input-output) copies files to and from a backup device. It uses standard input to take the list of filenames. It then copies them with their contents and headers to the standard output which can be redirected to a file or a device. This means that **cpio** can be (and is) used with redirection and piping.

cpio uses two *key* options, `-o` (output) and `-i` (input), either of which (but not both) must be there in the command line. All other options have to be used with either of these key options. The **cpio** options are shown in Table 15.3. The examples in this section and the next use System V device names. Linux users should use `/dev/fd0h1440`, and Solaris users should use `/dev/rdiskette` as the device names.

15.9.1 Backing Up Files (-o)

Since **cpio** uses only standard input, you can use **ls** to generate a list of filenames to serve as its input. The `-o` key option creates the archive on the standard output, which you need to redirect to a device file. This is how you copy files in the current directory to a 1.44 MB floppy:

```
# ls | cpio -ov > /dev/rdsk/f0q18dt                Use /dev/fd0 in Linux
array.pl
calendar
cent2fah.pl
convert.sh
xinitrc.sam
276 blocks                                         Total size of the archive
```

The `-v` (verbose) option displays each filename on the terminal while it's being copied. **cpio** needs as input a list of files, and if this list is available in a file, redirection can be used too:

```
cpio -o >/dev/rdsk/f0q18dt < flist
```

Incremental Backups **find** can also produce a file list, so any files that satisfy its selection criteria can also be backed up. You'll frequently need to use **find** and **cpio** in combination to back up selected files—for instance, those that have been modified in the last two days:

```
find . -type f -mtime -2 -print | cpio -ovB >/dev/rdsk/f0q18dt
```

Since the path list of **find** is a dot, the files are backed up with their relative pathnames. However, if it is a /, **find** will use absolute pathnames.

The `-B` option sets the block size to 5120 bytes for input and output, which is 10 times the default size. For higher (or lower) sizes, the `-C` option has to be used:

```
ls *.pl | cpio -ovC51200 >/dev/rdsk/f0q18dt              100 times the default
```

Multivolume Backups When the created archive in the backup device is larger than the capacity of the device, **cpio** prompts for inserting a new diskette into the drive:

```
# find . -type f -print | cpio -ocB >/dev/rdsk/f0q18dt
Reached end of medium on output.
If you want to go on, type device/filename when ready
/dev/fd0                                                  Device name entered
3672 blocks
```

Enter the device name when **cpio** pauses to take input. In this way, an archive can be split into several extents (volumes).

15.9.2 Restoring Files (-i)

A complete archive or selected files can be restored with the -i key option. To restore files, use redirection to take input from the device:

```
# cpio -iv < /dev/rdsk/f0q18dt
array.pl
calendar
cent2fah.pl
convert.sh
xinitrc.sam
276 blocks
```

When restoring subdirectories, **cpio** assumes that the subdirectory structures are also maintained on the hard disk; it can't create them in case they are not. However, the -d (directory) option overrides that.

cpio also accepts a quoted wild-card pattern, so multiple files fitting the pattern can be restored. Restoring only the shell scripts becomes quite easy:

```
cpio -i "*.sh" < /dev/rdsk/f0q18dt
```

Tip: A file is restored in that directory that matches its pathname. In other words, if a file has been backed up with the absolute pathname (e.g., /home/romeo/unit13), then it will be restored in the same directory (/home/romeo). However, when relative pathnames are used, files can be restored anywhere. The "relative filename" method is normally recommended because the administrator often likes to back up files from one directory and restore them in another. Make sure you use **find** with a dot, rather than a /, to specify the path list when you are using it with **cpio**. This applies to the **tar** command also.

Handling Modification Times (-m) By default, when a file is extracted, its modification time is set to the time of extraction. This could lead to problems as this file will participate in future incremental backups even though it has actually not been modified after restoration. Instead of using **touch** (*11.6.1*) to change the modification times (an impractical solution), you can use the -m option to tell cpio that the modification time has to be retained.

cpio compares the modification time of a file on the media with the one on disk (if any). If the disk file is newer than the copy, or of the same age, then it won't be restored; **cpio** then echoes this message: