cp

The **cp** program can put a copy of a file into the same directory or into another directory. **cp** doesn't affect the original file, so it's a good way to keep an identical backup of a file.

To copy a file, use the command:

cp *old new*

where *old* is a pathname to the original file and *new* is the pathname you want for the copy. For example, to copy the */etc/passwd* file into a file called *password* in your working directory, you would enter:

```
$ cp /etc/passwd password
$
```

You can also use the form:

cp *old olddir*

This puts a copy of the original file *old* into an existing directory *olddir*. The copy will have the same filename as the original.

If there's already a file with the same name as the copy, **cp** replaces the old file with your new copy. This is handy when you want to replace an old copy with a newer version, but it can cause trouble if you accidentally overwrite a copy you wanted to keep. To be safe, use **ls** to list the directory before you make a copy there. Also, many versions of **cp** have an **-i** (interactive) option that asks you before overwriting an existing file.

You can copy more than one file at a time to a single directory by listing the pathname of each file you want copied, with the destination directory at the end of the command line. You can use relative or absolute pathnames (see "the section "The Unix Filesystem" in Chapter 3) as well as simple filenames. For example, let's say your working directory is */users/carol* (from the filesystem diagram in Figure 3-1). To copy three files called *ch1*, *ch2*, and *ch3* from */users/john* to a subdirectory called *work* (that's */users/carol/work*), enter:

```
$ cp ../john/ch1 ../john/ch2 ../john/ch3 work
```

Or, you could use wildcards and let the shell find all the appropriate files. This time, let's add the **-i** option for safety:

```
$ cp -i ../john/ch[1-3] work
cp: overwrite work/ch2? n
```