Access Permissions

The *ls* command is often used with the option *-l* ("long output") to get information on a file. Try executing:

ls -l /bin/cp

This will produce the following output (somewhat shortened):

-rwxr-xr-x 1 root root ... /bin/cp

The first occurrence of the word *root* means that root is the owner of the file. The owner of a file is in most cases the user who created it, but this is not necessarily so, since files may change ownership across their lifespan (see →*chown*). The second occurrence of the word *root* in the above output means that the file belongs to a group called *root*. You'll learn what this means in a minute.

Every Linux system has a user called root. It is the system's superuser, that is, the only user who has all rights and privileges. The root user may delete any file, shut down the system, change users' passwords, and so on. On most Linux servers, only the administrator is allowed to log in as root, while everyone else has a regular user account.

When you installed Linux on your desktop computer, two user accounts were created for you: a root account and a regular user account. It's your choice when to use which account. In modern Linux distributions such as Debian, OpenSUSE or Ubuntu, both accounts are given the same password by default. This blurs the distinction between root and your regular user account somewhat, which is problematic because working as root can be very dangerous.

Here is an important piece of advice: don't log in as root. Any program that you run as root will have root privileges too. The security risks that this opens up could fill a book of their own! To give just two examples, you might accidentally delete an important system file while working as root, or you might execute a malicious program from the Internet that manipulates your system configuration. This cannot happen when you work as a normal user, because normal users (and the programs they run) are not allowed to modify system files.

The upshot is that you should only log in as a regular user, running single commands as root when necessary. Executing a single command as root is done by adding →*sudo* at the beginning of the command line, as described later in this book.

Now let's look at the strange code *-rwxr-xr-x*. It's best understood with reference to the following table:

	permissions		
type	**user**	**group**	**others**
-	rwx	r-x	r-x

The first character of the code (-) denotes the type of the file. The dash means that /bin/cp is a regular file. In Linux, executables such as →*cp* (a program used to copy files) are treated as regular files. Other file types are mentioned in the section on →ls later in this book.

The following group of characters (*rwx*) denotes the access permissions given to the file's owner (= "user"). The letter *r* means that the owner is allowed to read the file, *w* means that she is allowed to write to it (possibly deleting it), and *x* means that she has the right to execute it. In the context of a directory, *x* means the right to enter the directory using *cd* and view its contents using *ls*.

The next group of characters (*r-x*) tells us what users belonging to the file's group can do with the file: they may read (*r*) and execute it (*x*), but they're not given write access because the letter *w* is missing. The last group of characters (*r-x*) specifies what all other users may do with the file, that is, users not owning the file and not belonging to its group.

I have included this lengthy discussion of file ownership and permissions because inexperienced Linux users frequently see the following error message:

> *Permission denied*

This error occurs whenever you lack the permission to do something, e.g. modify a file. The solution is to look at the file's permissions and make the necessary adjustments using →*chmod*. Sometimes it makes sense to change the file's owner and group using →*chown* and →*chgrp*, but note that most system files are root-owned and should stay that way.