NEW TERM A *boot loader* is a program that steps in just after the BIOS loads and offers you the opportunity to start one of a number of operating systems that might be installed on the computer. The boot loader that installs with Linux is LILO.

Working with the X Window System

The end of the installation process is the most difficult, and the most important. It's time to set up your video card and monitor so that you can use the X Window System. You'll need it to use any of the graphical desktop environments, such as GNOME. If the video card and monitor are listed as compatible hardware for the GNU/Linux operating system, this process will be much easier.

You'll first be presented with a list of monitors. Search through the list to see whether your monitor is listed. If it isn't, you'll need to select one of the generic monitors. Pick the monitor that does what yours does. As in

```
A monitor that can do 1024 X 1280 at 60hz
```

Next, you'll select a resolution and color depth for the display. It is very important that you do not exceed the capabilities for your monitor and video card.

After you enter the correct information, the installation process will ask whether you want to test the X configuration. Answer yes. You should see a message that says, `Can you see this?` If you can, select Yes. Don't worry: If you can't see the message, a few seconds will pass and it will be assumed that the test failed and you'll be sent back to the beginning to try again.

The last step in the X configuration process is to tell the system whether you want the system to start the X Window System automatically when the system boots. Click the No button. Should you have a problem, you would rather be at the shell prompt when the system starts. From the shell prompt, you need only type **`startx`** to display the graphical user interface.

Congratulations! You've made it to the end of the long installation process. Remove the floppy disk and the CD-ROM from the drives and let the system restart itself.

Logging In and Logging Out

You can quit holding your breath. Take a look at your monitor; you should see a login prompt. What's a login prompt? To put it simply, the login prompt is where you type in a unique name that identifies you as an authorized user on the system. Depending on the distribution you installed and how you set up the X Window System, the login prompt might take various shapes and forms. If you installed Linux-Mandrake and elected not to start the X Window System when the system boots, the login prompt and associated bits of information might look as shown in Listing 2.3.

LISTING 2.3 A Sample of the Linux-Mandrake Login Screen

```
Linux Version 2.2.15-0.25mdk
Compiled #1 Mon Apr 24 03:00:34 PDT 2000
One 133MHz Intel Pentium Processor, 80M RAM
264.60 Bogomips Total
localhost.localdomain

Linux Mandrake release 7.1b (hydrogen)
Kernel 2.2.15-0.25mdk on an i586 / tty1
localhost login:
```

That's a lot of information. And, yes folks, this is not the computer you saw in Listings 2.1 and 2.2. It is one of the other computers in our computer lab and the one you'll see most often as you flip through these pages. Listing 2.3 shows an old, beat-up computer with a 133MHz Pentium processor and 80MB of RAM. But, back to the important part, the login prompt, which is followed by the cursor. What does it mean to log in? Logging in involves supplying a username and a password, which Linux verifies to determine whether the user has the authority to access the computer system.

NEW TERM When you installed and configured the Linux system, you set up at least one user account—the superuser account whose username is root. The *superuser* has unrestricted access to the entire operating system—every file and every command. Only the superuser can change the system setup and configuration. You might have also set up an *unprivileged user* account. This account is assigned to a regular user. Such a user can access files in his home directory and files used by general users (such as help files and program execution files).

At this point, you need to give the system your username before you can do anything with the system. The username that you supply is up to you. Every distribution requires that an account be created for a user named `root`. This account is assigned to the superuser (or the person in charge of maintaining and updating the computer system), and should be used only when performing system administration tasks.

NEW TERM The *system administrator* is responsible for the care, maintenance, upgrade, repair, configuration, and security of a computer, the computer operating system, attached peripherals, installed applications, and data files. The system administrator grants access to the computer, peripherals, and files to the various users on the system. The system administrator also makes sure that the users do not have problems operating the system and that the system is configured to efficiently help them perform their jobs.

Your other option is to log in as an unprivileged user. You might or might not have been given the opportunity to set up this type of user account during the installation. If you did not set one up, turn to Hour 12, "Sharing Files with Other Users on the System," to

learn how to create an unprivileged user account. This is the account in which you will be working the most. The unprivileged user account is where you will perform your day-to-day work, such as composing correspondence, searching the Internet, creating graphics for your Web pages, and keeping your checkbook balanced.

Listing 2.4 shows an example of a login session for a regular user account. It's quite easy to get into the system. First, type the username assigned to you and then press Enter. You will be asked for a password. Notice that you do not see the password as you type. After you have typed the password, press Enter. If the login was successful (that is, the system found your username and password in the database), you're ready to start exploring.

LISTING 2.4 The Login Session for a User Named `rusty`

```
localhost login: rusty
Password:
[rusty@localhost rusty]$
```

Logging In as the Superuser

There are two ways to work as the superuser. The first is to log in to the root account directly. This means that you type the username of **root** and then give the password for the root account.

> Because the superuser has access to every file, command, and program on the system, it is possible for the superuser to crash the system, delete files, or make other destructive changes. Be careful to give the superuser password to only those people with a need to know, and use the superuser account only for system administration. Use an unprivileged user account to perform other work.

The other way to get to be the superuser is to temporarily switch to the superuser account while working in your user account. This involves executing the su command.

Switching Between Consoles

You can run six different user sessions at one time. Use the function keys along the top of the keyboard. F1 through F6 are the available consoles. Press Alt plus the appropriate function key to move from console to console.

> Be security conscious when working with multiple consoles. It is easy to forget that multiple consoles and user accounts are open. Before you leave the computer, check each console to make sure that no users are logged in. This helps to prevent unauthorized entry into the system.

Task: Exploring the Login Process

Before you can drive your new operating system around the block, you'll need to turn the key to start the engine. Your key is your username and password. Follow these steps to learn how to log in to the superuser account and your unprivileged user account:

1. Access the superuser account. At the login prompt, type **root** and press Enter. The password prompt will appear.
2. Type the root password and press Enter. The shell prompt for the root user account displays and might look something like the following:

   ```
   [root@localhost /root]#
   ```

3. Change to a different console. Press Alt+F2.
4. Access your user account. At the login prompt, type the username for the unprivileged user account that you set up during the installation and press Enter. You will be asked for the account's password. Type the password and press Enter.
5. Temporarily change to the superuser account from the unprivileged user account. Type **su** and press Enter. You'll be asked for the password for the root user account. Type the password and press Enter. The shell prompt will look something like the following:

   ```
   [root@localhost rusty]#
   ```

6. Exit the superuser account. Type **exit** and press Enter. You'll be returned to your user account and once again the shell prompt might look like the following:

   ```
   [rusty@localhost rusty]$
   ```

You'll want to do your daily work from this user account. To get back to the console at which the root user is logged in, press Alt+F1.

Troubleshooting the Installation

You might need some extra help during the installation. There are a number of places where you can find some good information. One good source is the Linux HOWTOs. You might find these in the filesystem, but if not there, they are on the Web at `www.linuxdocs.org`. Here's a list of Linux HOWTOs that you might find useful:

- Installation HOWTO
- Config HOWTO
- Online Troubleshooting HOWTO
- Tips HOWTO
- Partition mini-HOWTO
- Multi-Disk HOWTO
- LILO mini-HOWTO
- Multiboot with LILO mini-HOWTO
- Bootdisk HOWTO
- From Power Up to Bash Prompt HOWTO

Task: Shutting Down the System

When it's time to shut down the computer for the evening, the designated system administrator must be called in. Shutting down the system is a job for the system administrator because the shutdown command can be executed only by the superuser account. Here's your first task as the system administrator of your system. Follow these steps to log in to the superuser account and turn off the computer:

1. Log out of your user account. Type **logout** and press Enter.
2. Return to the console where the root user is logged in. Press Alt+F1.
3. Terminate the session. Type **shutdown -h now** and press Enter. The system will go through the shutdown process and will turn off services and unmount devices. You'll see a long list of items scroll by on the screen. When the operating system has shut itself down, a message appears at the bottom of the screen telling you to power down.

> If you just want to restart the computer, type **shutdown -r now** and press Enter.

Summary

It's been a long hour but, hopefully, you are up and running with your brand-new GNU/Linux operating system. During this hour, you installed the operating system, selected the types of applications with which you'd like to experiment, and configured the X Window System so that you could display cool graphics on your screen. After the system was set up, you learned how to log in to the superuser and unprivileged user accounts and how to shut down the operating system and turn off the computer. Now you're ready to start exploring a completely GNU world of free software. Have a good time!

Q&A

Q I really do want to use the GNU/Linux operating system, but I don't want to install it. Do I have any other options?

A You could invite a GNU/Linux guru to dinner and then ask him to install the operating system and applications for you. You might need to bribe him with some homemade raisin oatmeal cookies for dessert. You could also check out a GNU/Linux user group near your home. Many GNU/Linux user groups hold installation meetings where you can take your computer and have someone help you with the installation. You'll find a list of user groups at `www.linux.org/users`. You can also purchase a computer with Linux pre-installed. To find a list of vendors, cruise over to `www.linux.org/hardware`.

Q Is there another program that I can use instead of LILO to boot between Microsoft Windows and GNU/Linux?

A There are two commercial programs that we know of. If you want to boot between multiple operating system, try System Commander (`www.systemcommander.com`) or BootMagic. BootMagic is included with the PowerQuest PartitionMagic program (`www.powerquest.com`). Some distributions use grub to select which operating system to load when the system starts.

Q When I set up the X Window System, I told the installation process that I wanted to start the X Window System automatically when the system boots. How can I get past the graphical login display and display the login prompt on a text-only display?

A You will need to change to a different virtual terminal. Press Ctrl+Alt+F1 to see a text-only login prompt. If you want to get back to the graphical login, press Alt+F7.

Workshop

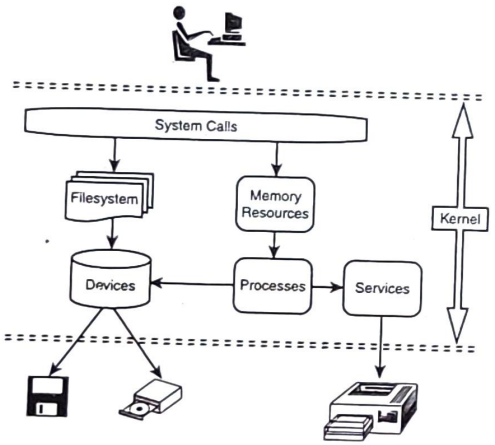
Test your knowledge of the material covered in this hour by answering the following questions and working through the exercises.

- Discover a few good programs and get help along the way
- Learn how processes fit into the operating system picture
- Get acquainted with the system log files

The Anatomy of the Linux Operating System

The center of the Linux operating system is the kernel. The kernel is a piece of software that provides an interface between you, the computer hardware, and attached peripherals. The kernel is responsible for maintaining the filesystem, executing commands, starting programs, timing system activities, and managing system memory and other resources. Figure 4.1 shows how the different parts of the kernel work together.

FIGURE 4.1
The kernel uses several tools to help it run the operating system.



The kernel gives instructions to the system using system calls. These system calls coordinate the activities of the kernel to produce the output for a command that has been executed by the system user.

NEW TERM *System calls* are communications between the kernel and the devices it manages, which direct the performance of actions requested by commands it receives from internal system processes or running applications, or are passed from the shell. These are the vehicles used by the kernel to execute shell commands. You'll learn how the shell acts as the interface between the user and the operating system in Hour 5, "Getting Acquainted with the Shell."

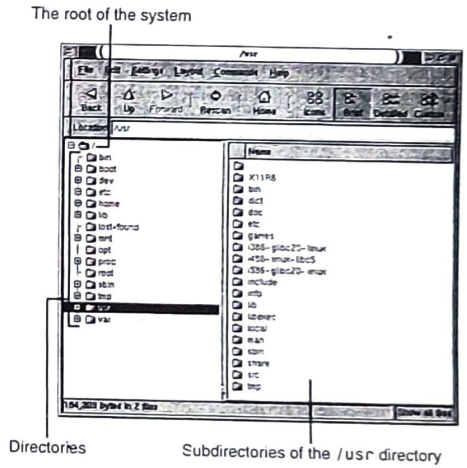
The most important job of the kernel is to manage the computer's memory resources. The memory resources are required to start processes that activate devices and services and cause the computer system to perform its job. All these goings on are managed through the filesystem.

Exploring the Directory Structure

Because earlier we told you that the Linux operating system is in reality a filing system, let's take a quick look at how the filing system is organized. Linux sees everything as files; therefore, it's important to understand exactly what a Linux file is and how Linux organizes files within the system. Every kind of file you might need is stored in the Linux filesystem; system files, data files, application files, utility files, driver files, configuration files, and more. Each of these files has a name and a directory in which it is stored in the filesystem.

The Linux filesystem is based on a tree structure, as are the DOS, Microsoft Windows, and UNIX filesystems. There is a root directory (identified by the / character) out of which all other directories and subdirectories grow. The files are contained within these directories and subdirectories. Figure 4.2 shows the basic structure of the Linux filesystem, as shown from the GNU Midnight Commander file manager.

FIGURE 4.2
The basic structure of the Linux filesystem.



Directory names can be made up of a combination of numbers, symbols, and characters. The name of the directory includes the location of the directory in the filesystem relative to the root (/) directory. In other words, the directory name includes the path to follow through the filesystem to get to the directory.

> The forward slash character (/) is also used to separate directories, subdirectories, and files. The slash character (/) by itself denotes the root directory.

Directories that are nested (contained) within other directories are called *subdirectories*. A parent/child relationship exists between the two directories. The home directory for your user account is one of these subdirectory branches. It's name begins with the / character, which signifies the root directory. Off the root directory is a subdirectory that contains all the home directories for the system users—the `/home` directory. Each user has a subdirectory of his or her own in the `/home` subdirectory.

For example, a user named "rusty" would have a home directory located in the `/home/rusty` directory path. Notice that there's nothing in front of the `r` in front of home. This signifies that `/home` is a subdirectory of the root directory. Because there's no / after `/rusty`, it could be confusing whether `/rusty` is a subdirectory of `/home` or a file stored there. Of course, to end the confusion, you can go ahead and place a / after `/rusty` to indicate that it is a directory.

The `/home` directory is not the only directory in the Linux filesystem. The following is a list of directories you'll find when looking around the Linux filesystem:

- The base of the operating system is the root (/) directory. It contains all the subdirectories and files in the Linux system. One important item that you'll usually find in the `/boot` directory is the Linux kernel. If you don't find it here, you'll find it in the / directory.

> Do not store any files in the / directory. Only system-specific configuration files (such as boot information files) are to be located at the root of the system.

- Executable files are stored in the `/bin` (for binary) directory. This is where Linux keeps its basic commands and programs. You'll learn about many of these commands as you progress through the book. There are commands that start the various shells, commands for working with files (such as copying and moving), commands to change file permissions, and configuration utilities such as `linuxconf`.
- The `/boot` directory is where the boot configuration files and commands are stored. This directory contains everything needed to boot (or *start*) the system.
- The `/dev` directory is where all the device files (or *drivers*) are kept for all your computer's hardware components. When you configure Linux to use a particular device, you edit one of the configuration files in this directory. One device that is attached to most computers is a floppy drive. The floppy drive is controlled by the `/dev/fd0` device file. The IDE hard drive device file is `/dev/hda`, or `/dev/hdb` for a second hard drive. For those who like SCSI these would be sda, sdb, sdc, etc.
- The `/etc` directory is where Linux keeps the system configuration files and initialization scripts. In later hours, you'll learn how to edit the files in this directory with a text editor to make configuration changes. You can edit these files to add users to the system and change their passwords (the `/etc/passwd` file), create group accounts for file sharing (the `/etc/group` file), and set up a dial-up Internet connection (the `/etc/resolv.conf` file).
- Each user on the system has a personal directory in which to store the files he or she creates. This directory is contained in the `/home` directory. Within the `/home` directory, each user has a separate subdirectory (except for the superuser). User home directories are for storing personal files and are not accessible by other users on the system.
- The `/lib` directory contains the libraries for C and other programming languages. It also contains the shared library images that are needed to boot the system and to run commands.
- The `/lost+found` directory is where to look for a file if you think that Linux has lost it. If there are several partitions on the system, there's a `/lost+found` directory on each partition.
- The `/mnt` directory is where other filesystems can be attached or mounted to the Linux filesystem. To view the contents of a CD-ROM, you would usually look in the `/mnt/cdrom` directory. To see the contents of a floppy disk, look in `/mnt/floppy`.

- The `/proc` directory contains virtual files that Linux uses to keep track of ongoing processes. You might think that some of these files are huge, but they are *virtual* files and are not actually there. They have other funny attributes, too. Although they take up no space at all on your disk, you can't delete them. You'll learn how to use the `/proc` files to collect information about the computer in Hour 9, "Hooking Up Hardware Devices."
- The `/root` directory is the home directory for the superuser or system administrator.
- The `/sbin` directory is where files (or tools) for use by the superuser (system administrator) are kept. The `/sbin` directory contains commands that shut down the system, set the system clock, check the filesystem for errors, and set up networking.
- Temporary files are located in the `/tmp` directory. This is a place where all users can temporarily store files for access by other users or for temporary storage. All the data stored in the `/tmp` directory will be lost when the system is rebooted.
- The `/usr` directory contains files that are not a part of the Linux filesystem. Applications for the X Window System and the Linux game collections are examples of what you'll find in this directory. There's also quite a long list of help files in the `/usr` directory.
- The log files are found in the `/var` directory. In here, you'll find the boot logs, the `dmesg` log, and others. Information in these logs indicates what software was loaded at boot and how the system hardware is configured.

Task: Exploring the `/usr` Directory

You have two different ways to explore the contents of the filesystem. One is to use some basic Linux commands from the command line. The other is to use one of the graphical file managers included with the desktop environment (whether its KDE, GNOME, or FVWM) that you are using.

This task takes you on an adventure through the `/usr` directory. This is probably the safest directory for you to explore because the files in this directory are available to all users, and they're read-only files. Follow these steps for an example of the useful information and programs you can find in the `/usr` directory:

1. Start with some fun and find the `/usr/games` directory. From the command line, use the `ls` command to list the directory contents. Type the following:

```
ls /usr/games
```


Now press Enter. You should see a list of games on your screen.

```
[rusty@localhost rusty]$ ls /usr/games
banner  bomber  civclient  civserver  fortune  matcher  xsolider
[rusty@localhost rusty]$
```

2. Try the `fortune` game. At the command line, type **`fortune`** and press Enter. If you're working from a terminal window, your display will look like the one shown in Figure 4.3.

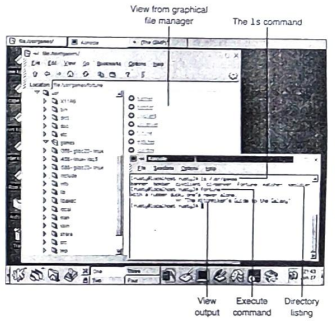
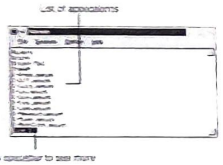


FIGURE 4.3
The KDE Desktop showing a file manager and a terminal emulator.

3. View the list of applications that are installed on the system and are available for use by any user. Type **`ls /usr/bin`** and press Enter. This is quite a long list, and it may scroll by quickly on the screen.
4. Read the contents of the `/usr/bin` directory in a pager (a special file viewer). Type **`ls /usr/bin | less`** and press Enter. The display will look similar to Figure 4.4. To scroll down a page, press the spacebar. When you're finished looking at the file list, press the Q key to quit the pager.



5. Find info manuals for the applications you just found. Type `ls /usr/doc | less` and press Enter. The files in these directories are a combination of text files and HTML files.

> If you're viewing the `/usr/doc` directory from a graphical file manager, you can open a help file by double-clicking the file. For example, you may want to read the Linux FAQ found in the `/usr/doc/FAQ` Linux directory. Double-click the file named `index.html`. The file will open in the file manager or in a Web browser (such as Netscape Navigator).

6. Look for artwork hidden away in the filing system. You'll find desktop backgrounds in the `/usr/share/wallpapers` directory and in the `/usr/share/pixmaps/backgrounds` directory. There are a variety of icons in the X Pixmap (`.xpm`) file format, which is a bitmapped file format, in the `/usr/share/icons` directory.

This task was a short introduction to some of the fun and useful information you'll find in the `/usr` directory. You'll learn more about using the `ls` command to list directory contents in Hour 6, "Looking at the Command Line." You'll also learn more about using pagers and how to use the `man` command to find helpful information about commands installed on the system in Hour 6.

Looking at File Types

Within each directory, you'll find a collection of files. A Linux file is a unique collection of data with a unique filename. The filename of a Linux file is a combination of its location in the filesystem relative to the root (base) directory, followed by the name assigned to it.

> It is possible to have more than one file in the filesystem with the same assigned name, but they cannot be stored in the same directory.

Linux contains many files, each filled with different information. They serve different purposes, and as you learn more about Linux, you'll learn more about the types of files and their use. To begin, we're going to concentrate on three types of files with which you need to become familiar:

- User data files are created by you and usually contain simple data made up of text and numbers. Files such as `.txt` files can be read with a text editor, but the more complex ones, such as those you create with graphics or spreadsheet programs, will require a viewer or interpreter program to display them.
- System data files are used by the operating system to keep track of user passwords, logins, file permissions, and other things pertaining to keeping the system running smoothly. You will not be involved in these as a regular user, but if you are the system administrator charged with the responsibility of maintaining the system, you may need to edit these files from time to time.
- Executable files contain the instructions that tell the computer what to do. These files are usually called *programs*. Whenever you give the computer a command to do something, you are telling it to follow the instructions in an executable file.

Task: Naming Files and Directories

Any system configuration files that you may edit and any application programs that you may install are assigned names and a place where they live in the system. As you work with these applications, you'll add new files to the filesystem for the letters,

reports, spreadsheets, Web pages, and pictures that you create. The first time you save a document, you need to select a directory in the filesystem in which to store the file and give the file a name. Here are a few steps to follow to help you store documents in the right place and name the files you create:

> The following rules apply to the naming of directories, because a directory is just seen as a file by the Linux filesystem.

1. Determine the length of the filename. A filename can contain as many as 256 characters. The filename is not just the name of the individual file; it includes the directory path that leads to the file. For example, the individual file named `picture.gif` that is located in the `/home/rusty/images` directory is 30 characters long. The actual filename is `/home/rusty/images/picture.gif`.
2. Use appropriate characters when creating a filename. You can use any character in the alphabet (from A to Z, in both lowercase and uppercase letters) or any number from 0 to 9. You can also use dashes (-), underscores (_), and periods (.).

> Linux filenames are case sensitive. An uppercase A is not the same character as a lowercase a. A file named `report99.txt` is seen as a completely different file from a file named `REPORT99.txt`.

3. Avoid keyboard characters that can cause problems in filenames. Do not put a space in the filename and do not use the following characters:
   `< > ' * ( ) [ ] { } ^ ! \ | & $ ? -`
4. To create hidden files, begin the filename with a period (.). If you want to keep your diary file hidden, give the file a name similar to `.diary.txt`.
5. Find the command line and change to your home directory. Type `cd` and press Enter to make sure you're in your home directory.
6. Create a file named `practice`. Type `cat > practice` and press Enter. The cursor will move to the next line on the screen. Type a few words and press Enter to move to the next line. Type a few words and press Ctrl+D to save and close the file. See Figure 4.5 for an example.
7. Read the file you just created. Type `cat practice` and press Enter.

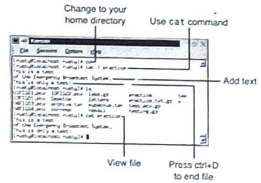


FIGURE 4.5 *Use the `cat` command to create a simple text file.*

Spawning Processes

Linux defines running programs, applications, utilities, and daemons (servers) as *processes*. As mentioned earlier, Linux maintains a virtual filesystem in the `/proc` directory, where it maintains pointers to all the ongoing processes in the system. Linux is a true multitasking operating system, meaning it can do more than one thing at a time. To accomplish all this, the Linux kernel just calls up a few daemons, and the daemons take over and perform the job, doing whatever is necessary.

It isn't quite that simple, but here is a real-world example of the kernel calling up the oldest daemon to get a job done. Suppose you have a document that you want to print, so you send a print command to the kernel. The kernel doesn't know enough about printing to do the printing itself, so it calls `lpd`, the line printer daemon.

> The line printer daemon (`lpd`) is the oldest daemon. In the very beginning of the computer age, the only output from computers was to a loud, obnoxious device called a *line printer*. The modern line printer daemon is much more sophisticated, but it still has the same name.

The `lpd` takes the print job from the kernel and spawns (forks is another term often used). This just means that it makes a copy of itself. Then the `lpd` gives the job to it's clone, along with all the details about which printer and where and when, and sends it

off to do the printing. The original `lp` daemon goes back to hanging out close to the kernel in case it gets called again. This allows the kernel to use the `lp` daemon to run many print jobs at once on many different kinds of printers—a real example of true multitasking.

Task: Reading the Kernel Boot Messages File

Several system log files will help you find out what the operating system is doing. These files are usually found in the `/var/log` directory. One file that you should look at is the `dmesg` file. This file contains all the messages from the kernel during the boot process. Here's how to view system log files:

1. Log in as the superuser. Type `su` and press Enter. You will be prompted for the password for the root account.
2. Read the messages that scroll by on the screen when you first boot the Linux operating system. Find the command line and type `less /var/log/dmesg` and press Enter. Your screen will look similar to the one shown in Figure 4.6.

FIGURE 4.6
Various messages appear on the screen while the operating system performs the boot process.

3. Scroll through the kernel messages. Press the spacebar to move down one screen at a time. Press the J key to move down one line at a time. Press Ctrl+B to move up one screen at a time.

4. Look for hardware detected when the system started. You may see lines that look like the following:

```
hda: WDC AC31600H, ATA DISK drive
hdb: COMPAQ CRD-256P, ATAPI CDROM drive
ide0 at 0x1f0-0x3f6 on irq 14
```

5. Press the Q key when you are finished to return to the shell prompt.

Summary

In this hour, you were introduced to the Linux operating system. You saw how Linux uses the filesystem to control how your computer operates. You learned what is contained in each of the directories and how the files in each directory work.

Q&A

Q I'd like to learn more about the Linux filesystem. Do you have any good references?

A The standard for the Linux filesystem structure can be found at `www.pathname.com/fhs/1.2/fsstnd-toc.html`. This standard details the types of files you'll find within each directory.

Q What is the easiest way to read the help files in the `/usr/doc` directory?

A Because much of the help information is written as HTML files, you may want to start the X Window System and your favorite desktop environment. Select a graphical file manager and navigate to the `/usr/doc` directory. If you double-click any of the HTML files, these files will most likely open in Netscape Navigator. If the file manager does not know which application in which to open the file, you'll be asked to choose one. You can try one of the many text editors installed on the system.

Q How can I display the processes that are running on my system?

A Try the `top` command. From the shell prompt, type **`top`** and press Enter. The `top` command shows you which users are logged in and which commands, applications, and services they are using. You'll learn more about processes in Hour 13, "Working with Processes."

Task: Creating a Temporary User Account

When you installed Linux, you created a superuser (or root user) account and possibly one unprivileged user account. For each account, you might have selected a shell program or you might have used the default settings. Whichever shell you chose, the shell loads automatically when you log in to your account. The shell found in most Linux distributions and installed by default is the `bash` shell.

If you didn't create an account for an unprivileged user during the installation of the Linux operating system, you will want to do so now. You will remember from previous chapters that the superuser account should be used only for system administration and configuration jobs. Before you begin working with the shell, experimenting with commands and applications, and exploring the Linux operating system, you need to create a login for a user account. Here's how to use the `adduser` and `passwd` commands to create a login:

1. Select a username. Each user on the system needs a unique login name to access the system and his or her user account. Usernames should be no more than eight characters. To make the job of the system administrator simpler, use lowercase letters for usernames.

   In this task, we'll create a user account for Rusty Spoon. The username that we have selected is *rusty*. You can substitute your own username, or you use this username to play with the Linux commands and explore the system.

2. Choose a password. A good password should be a combination of uppercase letters, lowercase letters, and numbers. A password should be at least eight characters. Do not use any word that can be found in the dictionary and use a character only once in a password.
3. Log in as superuser. At the login prompt, type **`root`** and press Enter. You are then prompted for a password. Type the password and press Enter. The shell prompt for the superuser account on a Linux-Mandrake system might look something like:

```
[root@localhost /root]#
```

4. Create the user account. At the shell prompt, type **`adduser rusty`** and press Enter (you may substitute your own username). Your display should look like the following:

```
[root@localhost /root]# adduser rusty
[root@localhost /root]#
```

5. Assign the password to the account. Type **`passwd rusty`** and press Enter. You'll see the following:

```
[root@localhost /root]# passwd rusty
Changing password for user rusty
New UNIX password:
```


Type the password and press Enter. If the system uses shadow passwords, you'll notice that the password does not appear while you are typing. This ensures that your system stays secure. You will need to verify the password by entering it a second time and pressing Enter. The system shows the following:

```
Retype new UNIX password:
passwd: all authentication tokens updated successfully
[root@localhost /root]#
```

6. Switch to a new terminal. Instead of logging out of the superuser account to get to your user account, press Alt+F2 to display a new terminal window and a login prompt as follows:

```
Linux Mandrake release 7.1b (hydrogen)
Kernel 2.2.15 0.25mdk on an i586 / tty2
localhost login:
```

   In the second line, tty2 is the virtual terminal that you are currently using. If you need to return to the superuser account, press Alt+F1.

7. Log in to your user account. At the login prompt, type the username for the account and press Enter. You are then prompted for a password. Type the password and press Enter. You're now working in the shell and the shell prompt will look something like the following:

```
[rusty@localhost rusty]$
```

The `adduser` and `passwd` commands are a fast and easy way to add new users to the system, assign a shell, and create a home directory. Our imaginary user *rusty* can now use the system. When *rusty* logs in, the `bash` shell will start because it is the default shell used by the `passwd` command. In addition, the home directory for the account is located at `/home/rusty`.

Looking at the Shell Programs

One of the installation suggestions made in Hour 2, "Installing Linux-Mandrake," was that you not have the X Window System start automatically when you boot the Linux operating system. If you followed this suggestion, when the system boots, you are presented with a text screen that asks for a login name and a password. After you enter this information, the system checks to see whether you are a properly authorized user and, if you are, it presents you with the shell prompt. An example of a shell prompt for the superuser on a Linux-Mandrake system is

```
[root@localhost /root]#
```

which it does (in step 4). After the application is running, the user communicates directly (step 5) with the application while the shell (in step 6) maintains the necessary data flow between the application and the kernel.

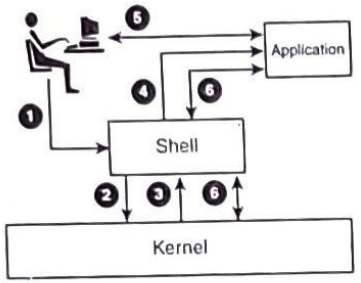


FIGURE 5.1
The process that the shell uses to maintain communication between a user, an application, and the Linux kernel.

There are a number of shells available for Linux and most Linux distributions provide a collection of the most commonly used ones. Some of these shells evolved out of the UNIX operating system, and have both a commercial and Open Source version. The Bourne shell (`sh`) is the oldest UNIX shell program. The default shell on all Linux distributions is the GNU Bourne again shell (`bash`). Shells available include the open source versions of the C shell (`tcsh`), the K shell (`pdksh`), and a number of others.

> The shells from which you can choose during the installation depend on which Linux distribution you chose and how it was installed. In addition to the standard shells, you'll find the public domain version of Dave Korn's Korn shell (`pdksh`) in the Red Hat distribution. The Z shell (`zsh`), the A shell (`ash`), and some others that are specific to particular Linux distributions are also available.

The most widely distributed UNIX shell is the Bourne shell (`sh`). It is probably installed on every existing UNIX system. Many of the newer shells are backward compatible with the Bourne shell. The GNU Bourne again shell (`bash`), written by Brian Fox and Chet Ramey, is the GNU/Linux version of the `sh` shell. It is a feature-enriched version with many of the features found in the C shell (`csh`) and the Korn shell (`ksh`). It is completely backward compatible with `sh`.

Another of the UNIX shells, the Korn shell (`ksh`) originally created by Dave Korn, is also available in an open source form. The public domain free version of the shell, created by Eric Gisin, is called the public domain Korn shell (`pdksh`). However, in March of 2000, AT&T released a freely downloadable version of the original Korn shell, updated by Dave Korn in 1993, called `ksh93`. It is not exactly "open source," but it is available for download for private and noncommercial use.

The C shell (`csh`) was developed at University of California at Berkeley to be more compatible with the C programming language. The open source version of the C shell is `tcsh`. There is a pointer (symbolic link) in the `/bin` directory for the `csh` shell that points to the `tcsh` shell.

The `bash` shell works like the Bourne shell (`sh`). There is a symbolic link, in the `/bin` directory of `sh`, that points to the `bash` shell. The `bash` shell is perhaps the most widely used shell in Linux, and it is used as the example in this hour.

Task: Running the bash Shell

You're probably anxiously waiting at the shell prompt and wondering what to do next. The first thing you need to do is check to see which shell you are using. During the rest of the hour, you will be working with the `bash` shell. So, if another shell is running on your system, you need to start `bash` to follow along. Follow these steps to start `bash` and learn about the shell:

1. Display the shell you are using. This information is found in the `SHELL` environment variable. Type **`echo $SHELL`** and press Enter. If you are using `bash`, the system will display the following:

```
[rusty@localhost /rusty]$ echo $SHELL
/bin/bash
[rusty@localhost /rusty]$
```

2. If you are not using `bash`, change to the `bash` shell. Type **`bash`** and press Enter. You will see the same shell prompt on the screen.
3. If you want `bash` to be the default shell, **`type chsh -s /bin/bash`** and press Enter. Double-check that you are using `bash`. Type **`echo $SHELL`** and press Enter (just as you did in step 1).
4. Display the `bash` help function. Type **`help`** and press Enter. The `help` command lists all the commands that are built into the shell.

When you log in to your Linux console, you will see the shell prompt. It will look something like this (if you are using the Linux-Mandrake distribution found on the CD in the back of this book and have created an unprivileged user account):

```
[rusty@localhost rusty]$
```

The first *rusty* is the username and it is followed by the hostname of the computer, and the final *rusty* is the name of the directory that *rusty* is in presently. You are automatically placed in your own user account when you first log in to the system.

The default shell prompt is often different for different distributions. For instance, for a user account, the prompts for the Slackware and Debian distributions look like this

```
localhost:~$
```

The default shell prompt for TurboLinux looks the same as the shell prompt for Linux-Mandrake.

When you are logged in as root, the shell prompt displayed earlier changes to use a pound sign (`#`) instead of the dollar sign (`$`) of the user accounts, and it looks like this

```
[root@localhost root]#
```

Task: Changing the Shell Prompt

The format of the `bash` prompt (as well as the prompt for the other shells) can be customized to better suit your needs. The `bash` shell has more than a dozen choices for prompts and some of the other shells have at least three times that many. The shell prompt can be changed to display information such as the username for a user, the directory in which the user is working, the current date and time, or a special message. To change the shell prompt for your current login session, follow these steps:

1. Display the shell variables. At the shell prompt, type **`printenv`** and press Enter. If you cannot read the entire screen, send the `printenv` command to the `less` pager (type **`printenv | less`**).
2. Look for the `PS1` variable. The `PS1` variable on a Linux-Mandrake system might look as follows:

   ```
   PS1=[\u@\h \W]\$
   ```

   This variable tells the system to display the username for whoever is using the host computer and that user's current working directory inside brackets followed by the `$` prompt (or `#` prompt for the root account).

3. Write down the information from the `PS1` variable.
4. Decide how you want the prompt to look. Table 5.1 lists the character codes that you can use to customize the shell prompt.

TABLE 5.1 Character Codes Used to Change the Shell Prompt

Character Code	*Description*
`\!`	Displays the number used in the history list to access the command that will be executed at the shell prompt
`\$`	Displays a `$` (normally found at the end of the shell prompt) in the prompt for regular users and a `#` for the superuser (or root user)
`\\`	Displays the backslash character
`\d`	Displays the current date
`\h`	Displays the hostname of the computer at which you are working
`\s`	Displays the name of the shell in which you are working
`\t`	Displays the current time
`\u`	Displays the username of the user who is logged in to the system
`\W`	Displays the current working directory
`\xxx`	Displays any special comments (replace *xxx* with your text)

5. Change the prompt so that it displays the time, the date, the current working directory, and the `$` prompt. Type **`PS1='[\t \d \W]\$ '`** and press Enter. The shell prompt will change to display the following:

   ```
   [14:15:33 Fri Jun 16 rusty]$
   ```

Once you log out of your account, the shell prompt will return to the default shell prompt.

In this task, you were introduced to environment variables and how they can be used to change how the shell looks and functions. The `printenv` command listed quite a number of environment variables. Besides the `PS1` variable, you might also be interested in the `$PATH` environment variable. The `$PATH` variable tells `bash` where to look for executable programs. If you do not have a path defined in this environment variable containing the directory where your program is stored, you will have to type in the absolute filename for the program to tell `bash` where to find it.

NEW TERM The shell program runs in an environment that depends on having certain preset values or pathnames in various resource text files found in each user's home directory and in the `/etc` directory. These are called *environment variables*. The default systemwide variables for the `bash` shell are found in the `/etc/profile` file. The personal variable settings are in the `.bashrc` file in your home directory.

Bashing Ahead

Everyone has his or her own way of working and a desire to make things work better in the way that a job is accomplished. The shell programs offer a way to customize the way your computer works to suit your preferences. You can make use of the built-in shell commands to accomplish tasks, and create shell scripts to schedule jobs to run at a specific time and day. The shell can also set up your Linux work area and load the software (on startup) that you need to work most efficiently. Now it's time to find out what the shell does for you to make it easier to communicate with the operating system.

Talking to the Shell

Linux uses the GNU `readline` command line editor to read and write to the command line. The `readline` editor is what you see at work when you type text after the shell prompt or read responses there. The editor can also edit whatever is on the command line. To use the command line, just begin typing after the shell prompt. You'll see the cursor is just one character ahead of you. If you make a mistake, just press the Backspace key to erase the text. Use the left and right arrow keys to move back and forth across the command. There are also several commands for moving around the command line and editing text, as shown in Table 5.1.

TABLE 5.1 Text Editing Commands for the Command Line

To Do This	*Press These Keys*
Move one character to the left	Ctrl+B
Move one character to the right	Ctrl+F
Move left by one word	Esc+b
Move right by one word	Esc+f
Move to beginning of the line	Ctrl+A
Move to the end of the line	Ctrl+E
Delete character to the left	Del
Delete character to the right	Ctrl+D

TABLE 5.1 continued

To Do This	*Press These Keys*
Delete word to the left	Esc+Del
Delete word to the right	Esc+d
Delete from cursor to end of line	Ctrl+K
Delete the entire line	Ctrl+U

One way to reduce the number of keystrokes that you type at the shell prompt is by using command line completion. Command line completion means that you type part of the command or program name and then you ask `bash` to finish your typing. If you type part of a command or application name on the command line and press Tab, `bash` attempts to finish the command for you. The `bash` shell searches through the system for all the commands and application names that match what you have typed in so far and completes the command as far as it is able. If the shell received insufficient data to make a choice, you will be presented with a list of possible completions. This makes it a simple matter to finish the command entry. To see this feature in action on the command line, type **`bas`** and press Tab.

After you press Tab to complete the command, command line completion will also attempt to finish filenames for you. If you begin typing the filename and then press Tab again the shell looks for possible matches in the files listed for the current working directory.

Another way to reduce keystrokes is by using wildcards. Wildcards take the place of other characters in the command line. The `bash` shell uses three different wildcards:

- The asterisk (`*`) character matches any number of characters.
- The question mark (`?`) matches a single character.
- Brackets (`[` and `]`) are used to match specific characters contained inside the brackets.

The `bash` shell and some of the other shells have a command history option that enables you to resubmit commands without having to type them in again. The way to access the command history is to use the up and down arrow keys to move up and down the list. Each of the previously typed-in commands will appear on the command line. You can press the Enter key to use any of the commands again. This feature can be a lifesaver if you are doing something that requires tiring repetitious typing of the same complex commands. The default number for the number of commands to be kept in the history is user configurable.

In this hour, you will:

- Identify the basic commands needed to properly operate as a user on the Linux operating system
- Use proper command-line grammar when typing at the shell prompt
- Become an efficient command-line typist by using wildcards and command-line completion
- Read manual pages to find out what task a command performs and how to use the command
- Search the filesystem for files and look up file information

Learning About Commands

There are many good reasons to become familiar with commands and their usage. If the X Window System did not install correctly on your system, you will need to work with the command line to properly configure your video display. After you work with Linux for a while, you'll find that it isn't possible to perform every task from a graphical desktop environment; you need to type a command or create a script. Those with repetitive strain injuries will find the command line a more comfortable place to work. Files can be moved and copied without the strain that dragging and dropping places on the mouse finger. These are just a few of the reasons it is important to learn about Linux commands and how to operate from the shell prompt.

You can use commands to manage your Linux system and these commands can be modified to behave the way you want. There's quite a bit of flexibility in how a command processes a given request. Commands can be modified to perform tasks in addition to the default actions of the command. These modifications can take the form of a command option, a parameter, an input/output redirection, or a pipe. After you discover how all these elements work together on the command line, you can use any Linux command.

All commands follow the same syntax, or command-line grammar. After you understand this structure, the job of typing commands will become a simple task. Two examples of the command syntax in its basic form can be seen in Figure 6.1.

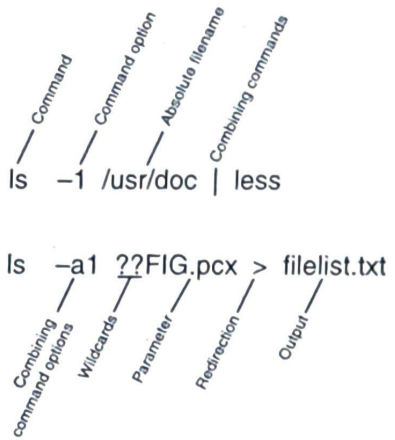


FIGURE 6.1
Two examples of the ls command; each uses a slightly different syntax.

The Basic Syntax for a Command

In the hours that follow, you'll see many places where the syntax for a particular command is given. For each command, there will be an example of the command usage, similar to the following example:

```
commandname -options [parameter]
```

This single-line command is what is referred to as the command line, and it is typed after the shell prompt. To execute the command, press Enter. The minimum usage for most commands is the command and one or more command options.

NEW TERM *Command options* are typed after the command and are used to modify the results of the command. Before typing an option into the command line, there must first be a space between the command and the command option. Then, a dash (-) character must precede the command option. The dash tells Linux to treat each letter that follows the dash as a command option. There is no space between the dash and the command option. There can be more than one command option, but do not put any spaces between the dash and the options.

In Chapter 7, "Moving Around the Filesystem," you'll learn how to create a simple command to view the files contained in a directory. Here's an example of this type of simple command:

```
ls -l /usr/doc
```

The command used in this example is `ls`, which displays a file list for the contents of the current working directory. The command option is `-l`, which modifies the information displayed about each file in the listing. The parameter tells the command to list the contents of the `/usr/doc` directory instead of the current working directory.

New Term A *command parameter* is some other type of information that the command can use to modify its output. A parameter can be a file, a directory, or a period of time. Parameters are not preceded by a dash. Parameters specify which file or directory a command is to be acted upon, or they can set a time limit in which the command is to be executed.

When reading the command syntax, keep the following in mind:

- Any text that appears in **bold** type must be typed exactly as it appears.
- Text that appears in *italics* is a placeholder for a name or value. The italicized text needs to be replaced with an actual filename or directory, or some other user-supplied data.
- Text that appears inside the square brackets ([]) is an optional part of the command. This part of the command syntax is not necessary to execute the command. Do not type the square brackets.

Typing on the Command Line

There are two different ways you can approach the command line. If you followed the directions in Hour 2, you will be in a text screen after you log in to your user account. If you feel brave, you can work from this text screen. If you log in to your user account and are taken directly to a desktop environment (such as KDE or GNOME), you can still work on the command line from a terminal emulator program. You might find a variety of terminal emulators on your system. The most popular is `xterm`. Search around your system, you might find a terminal emulator that is fun to use. One such terminal emulator is `eterm` (shown in Figure 6.2). Whether you type from a text screen or from a terminal emulator, the commands will perform the same actions.

Shell prompt

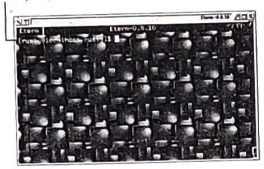


Figure 6.2 *The eterm terminal emulator can be configured to display a background and large text.*

Correcting Command-Line Mistakes

As you're typing along on the command line, you're bound to press a wrong key. What's a typist to do? You have several choices:

- For serious typing mistakes, you might want to consider clearing everything that appears on the command line. Just press Ctrl+U. After you have an empty command line, you can try typing the command again.
- If you caught the error quickly, press Backspace to clear the character to the left of the cursor. Backspace deletes only one character at a time. Press Backspace until the error is deleted, and then continue typing the command correctly.
- When the typo is in the middle of the command line, press the left-arrow key until the cursor is to the right of the error. Press the Backspace key to delete the error. You can then either insert new text (it will be inserted to the left of the cursor) or use the arrow key to move to the end of the command.

Using Command Shortcuts

There are a few typing shortcuts that will help make your job at the command line a little easier. Here's how you can cut down on your number of keystrokes per hour:

- Maybe your memory is fading and you can't remember a command you used in recent history. Press the up-arrow key and watch time travel backward as the previous command that you typed appears after the shell prompt. Press the up-arrow key to continue looking at each previously typed command. When the command you need appears after the shell prompt, press Enter to execute the command.

- The bash shell includes a trait that allows you to type a portion of a command or filename and the shell will guess at the complete command or filename. You need to type enough of the command or filename so that it can be uniquely identified. After you type a few characters, press Tab and the shell completes the rest of the command. If the shell cannot make a guess, you will hear a beep and a list of potential commands appears. Enter more characters and try to complete the command again.
- One way to get more mileage out of a command is with the use of wildcard characters. Wildcards are used to select files or directories that meet certain requirements. The `*` wildcard replaces one or more characters in the file or directory name. The `?` wildcard replaces a single character in the file or directory name. You'll work with wildcards in Hour 7, when you learn how to view the list of files in a directory.

Wildcards are used only for filenames and directory names. Do not use wildcards for command names or command options; Linux will not recognize the command.

Changing the Standard Output

When you execute a command, the command is read from the keyboard (the standard input) and the results (or the output) of the command appear on the screen (the standard output). If you want the command output to appear in a different place, or if you want to send the output to another command for processing, you will need to change the output with a process called *input/output redirection*. Command redirection is done by separating two or more commands with the `>` or `|` character.

One way to redirect the output is to save the results of the command to a file instead of displaying it on the screen. This enables you to save a picture of the output for later use. The `>` character is used when you want to send the output of a command to a file or to a printer.

Most Linux commands are simple tools that perform a single task. There might be times when you need to combine two or more commands to do a job. For example, if the list of files that displays as the result of the `ls` command is too long to read, combine the `ls` command with the `less` pager. This gives you the ability to scroll through the file list.

Combining commands is accomplished with a *pipe* (specified with the `|` character). The pipe acts as a go-between for two commands in which the output of the first command is 'piped' over to become the input of the second command.

After you become a command-line pro, you'll start combining commands and pretty soon your commands will be longer than the screen. When a command is too long to fit on one line, type the Backslash (\) character and press Enter to continue the command on the next line.

Task: Viewing the Command History List

The shell keeps track of the last 500 commands that you've typed. If you've used the command line frequently, using the arrow keys to scroll through the command history list might take some time. An easier way to see the commands you've used is with the `history` command. Here's how to easily list those commands you've used and select a command from the list to execute.

1. To see the command history list, type **`history`** and press Enter. The previously executed commands appear in a list. You'll see a number to the left of the command. An example of a history list is shown in Listing 6.1. This list contains two columns. The first column indicates the order in which the commands were typed. This is also the number you use to execute a command from the list. The second column is the command as it was executed.

Listing 6.1 Sample Output of the `history` Command

```
1 startx
2 man adduser
3 man passwd
4 shutdown
5 vi practice.txt
```

2. Search through the list until you find the command that you want to execute. To execute the command, type **`!`*`number`*** and press Enter, where *`number`* is the number in the first column of the history list that corresponds to the command you want to use. Using Listing 6.1, to start the X Window System (shown as `1 startx` in the example), type **`!1`**.

In this hour, you will

- Work with files by changing between directories and displaying the list of files contained in a directory
- Add and delete directories and files in your user account
- Create files and view the file properties
- Copy and rename files

Exploring the Home Directory

In Hour 4, "Understanding the Linux Operating System," you were introduced to the basic structure of the filesystem. You saw how files fit into directories, and how directories fit within other directories. When users are assigned accounts on the system, they are also allocated space within the `/home` directory in which to store their personal files. If a user has the login name of `rusty`, that user's personal files are stored in `/home/rusty`. This is known as the *home directory* and is the directory in which the user is working after logging in to his account. From this directory, the user can create subdirectories in which to organize files, create new files, modify these files, and delete unneeded files. Only the user, those authorized by the user, and the root account, may access files in the user's home directory.

When you first log in to your user account, the shell prompt looks something like this (that is, if your username is `rusty`):

```
[rusty@localhost rusty]$
```

Cool, but where does that leave you in the filesystem? Type **`pwd`** and press Enter. The response appears on the following line on the screen and looks like this for the user named `rusty`:

```
/home/rusty
```

Notice that the shell prompt appears on the next line. It's waiting for you to give it another command. That's good, because this is a good time to explore your home directory.

NEW TERM When working at a text-only screen, it might seem like it would be easy to get lost trying to find your way around the filesystem. There's an easy way to find out where you are. The `pwd` command prints the name of the working directory on the screen. The *working directory* is also known as the current directory. It is the directory

where any files you create are saved, unless you specify a different directory. If you request a file, the system will look in your working directory for the file. If the file is in a different directory, you will need to change to that directory, or include the directory path as a command parameter.

To see the contents of your home directory (which, at this point, should also be your working directory), use the `ls` command. Type **`ls`** and press Enter. Our trusty assistant `rusty` has been doing a little work and has already created a few files and played with a few programs. When `rusty` typed the `ls` command, the output in Listing 7.1 displayed.

LISTING 7.1 The Output of the `ls` Command

```
Desktop/     XBFIG03.pcx  XBFIG05.pcx  XBFIG07.pcx  tmp/
XBFIG01.pcx  XBFIG04.pcx  XBFIG06.pcx  nsmail/
```

The file list in Listing 7.1 doesn't supply much information. It's hard to tell which item in the list is a file and which is a directory. This is where command options come into play. One way to list more information about the files and directories is to add the long list format command option to the `ls` command. So, by modifying the `ls` command with the `-l` option (type **`ls -l`** and press Enter), the file list contains quite a bit of file information as shown in Listing 7.2.

LISTING 7.2 The Output of the `ls -l` Command

```
total 382
drwxr-xr-x   4 rusty   rusty     1024 Apr 25 19:37 Desktop/
-rw-r--r--   1 rusty   rusty    52899 Apr 25 20:50 XBFIG01.pcx
-rw-r--r--   1 rusty   rusty    51966 May  1 07:20 XBFIG03.pcx
-rw-r--r--   1 rusty   rusty    93230 May  1 08:12 XBFIG04.pcx
-rw-r--r--   1 rusty   rusty    76645 May  1 08:56 XBFIG05.pcx
-rw-r--r--   1 rusty   rusty    64473 May  1 09:11 XBFIG06.pcx
-rw-r--r--   1 rusty   rusty    40241 May  1 09:36 XBFIG07.pcx
drwx------   2 rusty   rusty        0 May 10 20:05 list2.txt
drwx------   2 rusty   rusty     1024 May  9 08:01 nsmail/
drwx------   2 rusty   rusty     1024 Apr 25 19:33 tmp/
```

At first glance, all the information in Listing 7.2 might look confusing. Nevertheless, take a moment to look at each piece of file information. Figure 7.1 will help give you a better picture.

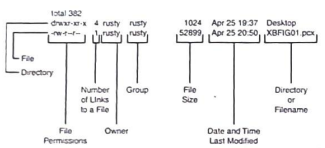


FIGURE 7.1
The file list displayed by the `ls -l` command.

Is it a file or is it a directory? The first column in the file listing tells you two things. The first character in the listing indicates whether the item is a directory (the `d` character) or a file (the `-` character).

The second item answers an important question: Can I use it? The next nine characters (the collection of `r`, `w`, and `x` characters) reflect the access permissions for the file. The first group of three characters is the file owner's permissions, the second group of three is the group permissions, and the last group is the permissions for all other users. The permissions are read (`r`), write (`w`), and execute (`x`). The dash means that access is denied. The third and fourth columns display the group and user that own the file.

> You'll learn more about setting file permissions, changing file ownership, and assigning a file to a group in Hour 12, "Sharing Files with Other Users on the System."

The last three columns of the file listing display the size of the file (in bytes), when it was last updated, and the filename.

The Syntax for the `ls` Command

SYNTAX

The `ls` command lists the contents of a directory in a variety of formats and the files are listed vertically. If no options are used, the names of files and directories (but not hidden files) appear in columns across the screen. The following code shows the syntax for the `ls` command:

```
ls [options] [filenames]
```

The `ls` command can be used with the *`filenames`* parameter to specify a directory path, a specific file, or a group of files. If you do not specify a directory path, the `ls` command will list those files in your working directory. If a directory path is used, all the files in

that directory will be listed. Wildcards can be used to display certain files, such as using `*.txt` to display only the text files in a directory.

Try these useful command options:

- To list the files horizontally instead of vertically, type `ls -x` and press Enter.
- File information such as permissions and owners, file size, and modification date for each file can be displayed in the long listing. Type `ls -l` and press Enter.
- To display all files including any hidden files, type `ls -a` and press Enter. If you want to include the hidden files in the long listing, type `ls -la` and press Enter.
- Different characters can be added to the end of a filename to make it easier to recognize different file types. Type `ls -F` and press Enter. A slash (`/`) appears after directories, an asterisk (`*`) after executable files, and an at (`@`) character after symbolic links.

> You can use multiple command options after the dash (`-`) character (for example, the `ls -la` command). It does not matter in which order the command options appear, so `ls -la` and `ls -al` provide the same output. You can only use multiple command options with commands that use single letters to specify command options.

Redirecting the Command Output

Displaying a file list in the long list fashion is fine if the file list isn't too long and fits on the screen. If you have a long list of files, the list will just scroll on past and you'll miss most of the information. You have choices—you can display the file list in a pager (such as `less`) or you can send the output to a file or to the printer. This looks like a good time to practice those pipe and redirection skills.

If you send the output to the `less` pager, you can scroll through the list at your leisure. To send the file list to the pager, type `ls -l | less` and press Enter. By sending the output to a file, you'll need to display the file in a text editor. To create the text file, type `ls -l > filename.txt` and press Enter.

> You'll learn about the `vi` text editor in Hour 8, "Text Editing with `vi`." You might also find other text editors installed on your system, such as `emacs`, `gxedit` (for GNOME), or `kedit` (for the K Desktop Environment).

Task: Displaying the `ls` Command Options

It's hard to remember all the command options associated with every command. Here's a quick way to list the command options for the `ls` command:

1. Type `ls --help | less` and press Enter. The list of options that can be used with the `ls` command will display in the `less` pager.
2. Use the arrow keys to read through the options section of the `ls` manual page.
3. Make a note of any file list command options that might be helpful to you.
4. Press Q to exit the `less` pager and return to the shell prompt.

Expanding the Directory Tree

As you work with applications, you'll need a place to store your work. The best place for you to store these files is in your user account. Now, you could just save all your files in your home directory, but things would become confused after a while. To help organize your filing system, create directories in which to store related information. To keep track of correspondence, spreadsheets, and scanned photographs, create directories for each type of file. If you run a small business, you might want to create directories for each client or project.

Creating Directories

When you want to create a subdirectory within the working directory, use the `mkdir` command. The easiest way to create a new directory is to make the directory in which you want the new directory to appear the working directory.

For example, user `rusty` wants to create a subdirectory named `letters` in which to store correspondence with family and friends. If the working directory is `/home/rusty`, a subdirectory named `/home/rusty/letters` is created by using the following command:

```
mkdir letters
```

To make sure that the subdirectory exists, use the `ls` command. Then, to change from the home directory to the new directory, use the `cd` command. The `cd` command makes the specified directory the new working directory. In the previous example, user `rusty` would change to the `letters` directory by typing `cd letters`. The shell prompt changes to show the new working directory, as shown here

```
[rusty@localhost letters]$
```

Then, to confirm that you are indeed in the directory in which you intended, use the `pwd` command to verify the working directory.

> To get back to your user home directory, type `cd` by itself on the command line and press Enter.

The Syntax for the `mkdir` Command

SYNTAX

If you remember DOS commands, you'll find a number of similarities when it comes to creating, deleting, and copying directories (and files, too). It is easiest to create a new directory with the `mkdir` command if the working directory is the directory that will contain the new directory. The following shows the syntax for the `mkdir` command:

```
mkdir [options] directoryname
```

The *`directoryname`* parameter is the name of the new directory to be created. This directory will be the child directory of the working directory.

There are two command options that you can use with the `mkdir` command that you'll find helpful:

- To create a parent directory with a series of subdirectories, use the *parents* (`-p`) command option. For example, you might want to store your correspondence files in a separate directory within your home directory. Within this directory, you plan to have a directory for each client, and within each client directory, there will be a separate directory for each year's correspondence. Using this example, type `mkdir -p corresp/client/2000` and press Enter to create all three directories. This is much easier than creating the `corresp` directory, changing to the `corresp` directory (with the `cd` command), creating the `client` directory, and....
- You can also assign access permissions to the newly created directory with the *mode* (`-m`) command option. You'll learn how to assign file permissions to directories in Hour 12.

Task: Creating a Directory Structure for a User Account

It's time to use some of your command-line knowledge and create a few directories in your user account. Storing your data files in directories makes it easier to locate files and keep them organized. Here are some ideas to help make creating directories from the command line a little easier:

1. Make sure that you are working in your home directory. Type **pwd** at the shell prompt and press Enter. If your username is *rusty*, the **pwd** command should display your home directory as follows:

   ```
   /home/rusty
   ```

   If you are not in your home directory, type **cd** and press Enter. Now try the **pwd** command to make sure of your location.
2. Create a directory in which to store correspondence files. Type **mkdir corresp** and press Enter.
3. Make sure that the **corresp** directory is located in your home directory. Type **ls** and press Enter.
4. Change to the **corresp** directory. Type **cd corresp** and press Enter. The shell prompt will change and will look something like the following:

   ```
   [rusty@localhost corresp]$
   ```
5. Create a correspondence directory for a client named BigBiz for the month of January. Type **mkdir -p BigBiz/Jan** and press Enter.
6. Create a correspondence directory for BigBiz, but for the month of February. Use the command history to make this job easier. Press the up-arrow key to display the previous command that you typed, press Backspace to erase the **Jan** portion of the command line, and then type **Feb** and press Enter.
7. Make sure that BigBiz and child directories were created. Change to the **BigBiz** directory (type **cd BigBiz** and press Enter), and then list the contents (with the **ls** command).
8. Go back to your home directory. Type **cd** and press Enter.
9. Review all the files and directories contained in your user account with the **find** command. Type **find $HOME -print** and press Enter. Every file and directory (including the full directory path) prints on the screen. If the files fly past on the screen, redirect the output to the **less** pager (type **find $HOME -print | less**) or to a file (type **find $HOME -print > *filename***) to make it easier to look at the list.

Understanding Absolute and Relative Filenames

When you type **ls** to list the contents of a directory, only the contents of the working directory display. If you search for a file, and you only specify a filename, Linux will only look in the current directory for the file. If you want to work with a file that is not located in the working directory, you must specify the directory where the file is located and the name of the file.

The method used to move to a different directory in the filesystem is the **cd** command. To change to a different directory, type **cd** followed by the directory path. If the home directory is the working directory (for example, **/home/rusty**) and you want to go to **/home/rusty/letters/chainmail/Feb**, type **cd letters/chainmail/Feb** and press Enter. There is no need to type the entire directory path because the directory you want is relative to the working directory. However, if you want to go to the **/usr/doc** directory to see the help files for programs installed on the system, you would need to type **cd /usr/doc** and press Enter. You use the absolute filename because the **/usr/doc** directory is not in the path of the working directory.

New Term There are two types of Linux filenames—absolute and relative filenames. An *absolute filename* begins with the slash (**/**) character indicating the root of the filesystem, followed by the complete directory path to a file. Every file on the system has an absolute filename and the absolute filename can be used from any location in the filesystem. The *relative filename* refers to a file's location beginning from the working directory. The relative filename does not begin with a slash character.

Try these shortcuts when using commands in relation to your working directory: One dot (**.**) indicates your current directory, and two dots (**..**) specifies the parent directory of your current directory. If you want to change to the parent directory of the current working directory, type **cd ..** and press Enter.

Deleting Directories

The **rmdir** command is used to remove directories. Before you can delete a directory, you must first delete the contents of the directory. This means that any files contained in the directory need to be removed with the **rm** command (discussed later in this hour), and subdirectories need to be emptied and then deleted with the **rmdir** command.

While you are learning to drive your new operating system, you might want to limit the directories you delete to those you have created within your user account.

The Syntax for the rmdir Command

Syntax

When you want to delete a directory, the working directory must be the parent directory of the directory to be deleted. Also, if there are any files in the directory to be deleted, the files must be removed first. The following shows the syntax for the **rmdir** command:

```
rmdir directory
```

The *directory* parameter is the name of the directory to be deleted. If the directory is not empty, a message displays:

```
Rmdir: <directory>: Directory not empty
```

This warning message lets you know that you need to manually delete the contents of the directory. The command does not remove the directory or any of the files contained in the directory if the directory to be deleted is not empty.

Working with Files

A file is probably the most basic piece of electronic information with which you'll work on the system. Files hold data that you create when you use an application, such as a graphic image, a word processing document, a Web page, or a database. Each file that you create has a distinct name and can be stored in one of many directories on the system. The most likely place that you'll store files will be in your home directory.

After you have saved a file on the system, you can perform a variety of tasks with the file. You can update the file by opening it in an application and making changes to the information. Files can be moved from directory to directory to better organize your information. You can also keep copies of a file in several directories if you need a file for different purposes.

Creating Files

If you want to read the contents of a file quickly, use the **cat** command. The **cat** command can read one or more files at a time, or it can be used to combine the contents of two files into one file. It will even create a new file. And...all this comes to you right from the command line. Pretty slick, huh?

Before you ask **cat** to display a file, you'll need to change to the directory that contains the file. Remember, the **cd** command moves you from directory to directory. When you execute the **cat** command (type **cat *filename*** and press Enter), the file quickly rolls along on the screen.

Here are some other cool tricks that you can teach the **cat**:

- When you need to combine the contents of several files into one new file, tell **cat** which files to combine and in what order and the name of the new file. The rest can be left to **cat**. For example, user **rusty** collects jokes. Each joke is placed in a separate file and occasionally **rusty** picks a few good jokes to share with friends. This week, **rusty** wants to send the joke files named **joke27** and **joke53** to some pals. To combine these two files into a third file named **grins1**, type **cat joke27 joke53 > grins1** and press Enter.
- To add the contents of one file to the end of another file, tell **cat** to append the file. If **rusty** has a file named **guffaws1** that some friends would enjoy, but also wants to also share **joke27**, **rusty** would use **cat** to copy the contents of **joke27** and then paste the contents to the end of the file named **guffaws1**. To append the file, type **cat joke27 >> guffaws1** and press Enter.
- The **cat** command can also be used to create a blank file. User **rusty** wants to put together a collection of the best jokes of the year. To create a new file named **best2000**, type **cat > best2000** and press Ctrl+D. Now, with a new, blank file, **rusty** can search the joke collection and use **cat** to append the best of the best to the end of the **best2000** file.

Task: Displaying File Information with the stat Command

Earlier in the hour, you saw how the **ls -l** command displayed quite a bit of information about a file. Another way to display file information, but this time for an individual file, is with the **stat** command. The **stat** command can also display this same information for a directory. Here's how to use the **stat** command:

1. Change to the directory that contains the file about which you want more information.
2. Display the file list for the directory (use the **ls** command) and locate the file.
3. Type **stat *filename*** and press Enter. A sample output of the **stat** command is found in Listing 7.3. If you receive the **Can't stat file** error message, it means that the file does not exist.

Renaming Files

There might be many reasons why you want to rename a file. Maybe you don't like the original filename; it might not be descriptive enough. The `mv` command is used for renaming files.

The Syntax for the `mv` Command

SYNTAX

When you rename a file, the original filename is erased and then replaced with the new filename. All the information stays the same, only the name is changed. The following shows the syntax for the `mv` command:

```
mv [options] sourcefile targetfile
```

The *`sourcefile`* parameter is the name of the file to be copied and the *`targetfile`* parameter is the new name of the file. For example, if a file is named `sample1` and you want to change the filename to `sample2`, type **`mv sample1 sample2`** and press Enter. You can also rename directories in this same way. When used without any command options, the `mv` command deletes the original file when the renamed file is created. In the example, the file named `sample1` is renamed to `sample2`, and the file `sample1` is deleted from the system.

Two command options are used frequently:

- To ensure that you don't overwrite an existing file (by renaming a file using the same filename as another file in the same directory), use the *backup* (`-b`) command option. To use this option, type **`mv -b sample1 sample2`** and press Enter. This creates a new file named `sample2`, deletes the file named `sample1`, and if there were an existing file named `sample2`, it would now be called `sample2~`.
- Another way to ensure that you don't copy over an existing file is to use the *interactive* (`-i`) command option. The interactive option lets you know if you are about to overwrite an existing file. To use the interactive command option, type **`mv -i sample1 sample2`** and press Enter.

▲

Duplicating Files

When you need to make a copy of a file, use the `cp` command. The `cp` command works much the same way as the `mv` command, except that the `cp` command does not delete the original file. To copy the file named `copy1` to create a new file named `copy2`, type **`cp copy1 copy2`** and press Enter.

Use the interactive command option (`-i`) with the `cp` command so that you don't overwrite any existing files.

Task: Sharing Files with Microsoft Windows

If you create files on a computer using the GNU/Linux operating system, you might find that you want to use these files on another computer that is running Microsoft Windows. Maybe you scanned a few photographs of your pet and enhanced it in The Gimp, and now you want to share it with a friend. What do you do when your friend has an old Windows machine, and hasn't heard about the Internet? Maybe you should introduce him to the 21st century and GNU. Until then, format a floppy that will work on your friend's computer and copy that fabulous photo. Here's how to format an MS-DOS floppy disk:

1. Place the floppy disk in the floppy disk drive.
2. Log in as the superuser from your user account. Type **`su`** and press Enter. You will be asked for the root password. Type the root password and press Enter.
3. Perform a low-level format on the floppy disk. Type **`fdformat /dev/fd0H1440`** and press Enter. (`/dev/fd0` is the name of the device file that controls the floppy disk drive, and `H1440` is the size of the disk.) You'll see a message such as the one in Listing 7.4.

LISTING 7.4 The Output of the `fdformat /dev/fd0H1440` Command

```
double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB.
Formatting … done
Verifying … done
```

If there are any bad blocks on the disk, a message similar to Listing 7.5 displays.

LISTING 7.5 Sample Error Message from the `fdformat` Command

```
double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB.
Formatting … done
Verifying … floppy0: data CRC error: track 54, head 1, sector 5, size 2
Floppy0: data CRC error: track 54, head 1, secotor 5, size 2
Problem reading cylinder 54, expected 18432, read 11264
```

4. Create a DOS-FAT filesystem on the floppy so that you can share the floppy between a GNU/Linux system and a Microsoft Windows system. Type **`mformat a:`** and press Enter.

5. To copy a file from your user account to the floppy disk, change to the directory that contains the file, and type **`cp filename /mnt/floppy`** and press Enter.
6. To get back to your user account, type **`su username`** and press Enter.

If you are using the Debian GNU/Linux distribution, use the `superformat` command instead of the `fdformat` command.

Summary

After a little practice on the command line, you should begin to feel more comfortable working on your GNU/Linux system. During the past hour, you saw how easy it is to navigate the filesystem and create a directory structure in which to store your files. And you accomplished all this straight off the command line. If you still feel a little unsteady when working with files and directories, read the manual pages. You might also want to experiment in your home directory. Remember, you can always delete your experiments later.

Q&A

Q Are there some good Web sites where I can learn more about file management and maintaining my GNU/Linux system?

A Here's a couple of sites that we've come across that might be of interest to you:

The Kansas City Area Linux Special Interest Group put together a guide especially for new GNU/Linux users that explains the concepts of user and file management. You'll find it at `www.sabakisoftware.com/linux/topics/admin`.

The Linux Newbie Administrator Guide at `sunsite.auc.dk/linux-newbie/` contains information useful to the new user and system administrator. You'll find information on installing the operating system, configuring the system startup, upgrading the operating system, working with files and directories, available applications, and the X Window System.

You can also search through the stacks at the Linux Documentation Project (`www.linuxdoc.org`). You'll find a number of guides, the Linux HOWTOs and FAQs, and you can search the manual pages.

LISTING 7.3 The Output of the `stat` Command

```
  File: `XBFIG06.pcx'
  Size: 64473           Allocated Blocks: 128          Filetype: Regular File
  Mode: (0644/-rw-r--r--)         Uid: (  503/   rusty)  Gid: (  505/   rusty)
Device:  3,7   Inode: 75819     Links: 1
Access: Mon May  1 09:36:39 2000
Modify: Mon May  1 09:11:38 2000
Change: Mon May  1 09:11:38 2000
```

Compare the output of the `stat` command with the `ls -l` command. Type **`ls -l *filename*`** and press Enter.

You'll notice the same file permissions and user information. The `stat` command displays more information about when the file was last accessed and last changed.

Deleting Files

Sometimes you just don't need a file anymore. To get rid of unneeded files, delete them with the `rm` command. When you use the `rm` command, you need to specify the file (or files) that you want to delete. You must also have write access to the file before you can remove the file from the system.

You might want to play it safe when you delete files. Linux can prompt you each time that you try to delete a file to make sure that you really want to delete the file permanently. The `rm` command has an interactive option so that each time you delete a file, you are asked whether you really want to remove the file. If you want to be prompted each time you delete a file, type **`rm -i *filename*`** and press Enter. You'll see the following message:

```
rm: remove `filename'?
```

If you want to delete this file, type **`Y`** and press Enter. This permanently deletes the file. If you've executed the `rm` command in error, type **`N`** and press Enter. The file will be left in place.

After you delete a file, there is no way to get it back. To avoid an unnecessary loss, always use the `rm` command with the interactive option.

Renaming Files

There might be many reasons why you want to rename a file. Maybe you don't like the original filename; it might not be descriptive enough. The `mv` command is used for renaming files.

The Syntax for the `mv` Command

SYNTAX

When you rename a file, the original filename is erased and then replaced with the new filename. All the information stays the same, only the name is changed. The following shows the syntax for the `mv` command:

```
mv [options] sourcefile targetfile
```

The *`sourcefile`* parameter is the name of the file to be copied and the *`targetfile`* parameter is the new name of the file. For example, if a file is named `sample1` and you want to change the filename to `sample2`, type **`mv sample1 sample2`** and press Enter. You can also rename directories in this same way. When used without any command options, the `mv` command deletes the original file when the renamed file is created. In the example, the file named `sample1` is renamed to `sample2`, and the file `sample1` is deleted from the system.

Two command options are used frequently:

- To ensure that you don't overwrite an existing file (by renaming a file using the same filename as another file in the same directory), use the *backup* (`-b`) command option. To use this option, type **`mv -b sample1 sample2`** and press Enter. This creates a new file named `sample2`, deletes the file named `sample1`, and if there were an existing file named `sample2`, it would now be called `sample2~`.
- Another way to ensure that you don't copy over an existing file is to use the *interactive* (`-i`) command option. The interactive option lets you know if you are about to overwrite an existing file. To use the interactive command option, type **`mv -i sample1 sample2`** and press Enter.

▲

Duplicating Files

When you need to make a copy of a file, use the `cp` command. The `cp` command works much the same way as the `mv` command, except that the `cp` command does not delete the original file. To copy the file named `copy1` to create a new file named `copy2`, type **`cp copy1 copy2`** and press Enter.

Use the interactive command option (`-i`) with the `cp` command so that you don't overwrite any existing files.

Task: Sharing Files with Microsoft Windows

If you create files on a computer using the GNU/Linux operating system, you might find that you want to use these files on another computer that is running Microsoft Windows. Maybe you scanned a few photographs of your pet and enhanced it in The Gimp, and now you want to share it with a friend. What do you do when your friend has an old Windows machine, and hasn't heard about the Internet? Maybe you should introduce him to the 21st century and GNU. Until then, format a floppy that will work on your friend's computer and copy that fabulous photo. Here's how to format an MS-DOS floppy disk:

1. Place the floppy disk in the floppy disk drive.
2. Log in as the superuser from your user account. Type **`su`** and press Enter. You will be asked for the root password. Type the root password and press Enter.
3. Perform a low-level format on the floppy disk. Type **`fdformat /dev/fd0H1440`** and press Enter. (`/dev/fd0` is the name of the device file that controls the floppy disk drive, and `H1440` is the size of the disk.) You'll see a message such as the one in Listing 7.4.

LISTING 7.4 The Output of the `fdformat /dev/fd0H1440` Command

```
double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB.
Formatting … done
Verifying … done
```

If there are any bad blocks on the disk, a message similar to Listing 7.5 displays.

LISTING 7.5 Sample Error Message from the `fdformat` Command

```
double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB.
Formatting … done
Verifying … floppy0: data CRC error: track 54, head 1, sector 5, size 2
Floppy0: data CRC error: track 54, head 1, secotor 5, size 2
Problem reading cylinder 54, expected 18432, read 11264
```

4. Create a DOS-FAT filesystem on the floppy so that you can share the floppy between a GNU/Linux system and a Microsoft Windows system. Type **`mformat a:`** and press Enter.

5. To copy a file from your user account to the floppy disk, change to the directory that contains the file, and type **`cp *filename* /mnt/floppy`** and press Enter.
6. To get back to your user account, type **`su *username*`** and press Enter.

If you are using the Debian GNU/Linux distribution, use the `superformat` command instead of the `fdformat` command.

Summary

After a little practice on the command line, you should begin to feel more comfortable working on your GNU/Linux system. During the past hour, you saw how easy it is to navigate the filesystem and create a directory structure in which to store your files. And you accomplished all this straight off the command line. If you still feel a little unsteady when working with files and directories, read the manual pages. You might also want to experiment in your home directory. Remember, you can always delete your experiments later.

Q&A

Q Are there some good Web sites where I can learn more about file management and maintaining my GNU/Linux system?

A Here's a couple of sites that we've come across that might be of interest to you:

The Kansas City Area Linux Special Interest Group put together a guide especially for new GNU/Linux users that explains the concepts of user and file management. You'll find it at `www.sabakisoftware.com/linux/topics/admin`.

The Linux Newbie Administrator Guide at `sunsite.auc.dk/linux-newbie/` contains information useful to the new user and system administrator. You'll find information on installing the operating system, configuring the system startup, upgrading the operating system, working with files and directories, available applications, and the X Window System.

You can also search through the stacks at the Linux Documentation Project (`www.linuxdoc.org`). You'll find a number of guides, the Linux HOWTOs and FAQs, and you can search the manual pages.

It is important that you learn to work a text editor from the command line. If you ever have system problems, cannot get to a GUI, and need to make configuration changes to get the system up and running, you'll be glad you spent an hour with `vi`.

In this hour, you will:

- Learn about the different types of configuration files that can be updated in a text editor
- Open and read files in the `vi` text editor
- Edit files and save the changes

Working with Text Editors

All text editors perform the same basic, common tasks—opening files, editing text, and saving the changes. Because all text editors read and write ASCII text files, a file created in one text editor can be edited in another text editor. One way to make changes to a file is to insert and delete text. All text editors provide a way to move around in the text file, use the Backspace key to erase mistakes, and insert text in any location. Another way to change the file is to move information around a file, or to copy a piece of information and use it in another location. To help make the editing job a little easier, text editors can search a file for a single word or a string of text. You might want to search a file to find a particular word in the file, or to replace the text with some other text. After you've made your changes, remember to save the file.

Editing Configuration Files

The main job of a text editor is to edit system configuration files. Yes, we know these configuration tasks can be performed using a graphical front end to change screen resolution, create Internet connections, and change passwords. And there are some excellent front ends. One well-known configuration utility is linuxconf (found in Appendix C, "Introducing GNOME and Linuxconf"). linuxconf does the job of editing configuration files, creating user and group accounts, setting up file sharing, establishing Internet connections, displaying system logs, and changing the date and time.

Even though you can use a cool tool to edit configuration files, you might find that you have more control over how your system operates if you edit these configuration files yourself. After you are comfortable editing configuration files, you can quickly back up a file, type a few changes, test those changes, fix any problems, and implement your new configuration. There are two types of files that you might find a need to edit. These files control the configuration of your user account and the configuration of the entire system.

New Term *User startup files* on the Linux system enable you to configure the behavior of the shell and other programs when you access your user account. Some programs do not need a startup file to save configuration changes. These programs use their built-in default configuration.

Startup files are located in your home directory as hidden files (indicated by the period at the beginning of the filename). Here are some of the startup files that you might want to explore:

- The `.bashrc` shell script is a startup file that controls a number of occurrences when you log in to your user account. You'll need to edit this file if you want to change the format of the shell prompt, the number of commands saved in the command history list, the directories in which the shell looks for commands, and special keyboard shortcuts, just to name a few. If you are using the C shell or `tcsh`, edit the `.cshrc` file.
- When working in the X Window System, you can start programs automatically when you start the graphical desktop environment. The commands to start the programs are added to the `.xinitrc` file. If you want to change the default window size, background color, and font type and size, make those changes in the `.Xdefaults` file.
- Desktop environments, such as KDE, place icons on the desktop that are shortcuts to applications and files. KDE uses the `.kdelink` file to create links between the execution file for the program and the desktop.
- If you're a regular newsgroup lurker, keep track of your newsgroup list in the `.newsrc` file.

New Term Linux contains a number of files, called *system configuration files*, that control how the system operates. These files specify how the system boots, what filesystems and peripherals are available for use, the screen resolution and color depth, and connections to the Internet or a network.

If you want to attempt system configuration changes by hand-editing files, here are a few of the files with which you'll need to become familiar:

- Systems that run both Linux and another operating system need a way to switch between operating systems. This can be done with a tool called `lilo`. If you need to set up `lilo` or if there are problems booting operating systems, edit the LILO configuration file. This file is usually found in the `/etc/lilo.conf` file. If you can't find this file, look for `/boot/lilo.conf` or `/etc/lilo/config`.

- Before you can access any filesystem on the computer, the filesystem must be mounted; that is, the filesystem must be associated with a particular directory. The file that controls the mounting of filesystems at boot time is the `/etc/fstab` file. You might need to edit this file if you add a new hard drive, Zip drive, or CD-ROM drive to the computer.
- To start a process when the operating system boots, edit the `/etc/inittab` file. A process can be a program, a command, or a scheduled job.
- Printers are added to the system with the `/etc/printcap` file. This file contains the make and model of the printer, the port to which it is connected, and where the print spool is located.
- To work with the X Window System, you will need to configure the monitor. The settings for the monitor model and type, refresh rates, screen resolution, and color depth are configured in the `/etc/X11/XF86Config` file.
- As the system administrator, you are responsible for adding and deleting accounts for each user and group that will be working on the system. One way to add new users is to edit the `/etc/passwd` file. To add group accounts to the system, edit the `/etc/group` file.

It is very dangerous to tinker with the password and group files and you must be careful to make backups of all configuration files before you edit anything in them. The methods for doing this can be found in Chapter 12, "Sharing Files with Other Users on the System."

- Creating an Internet connection involves creating two scripts and editing one file. You will need a script that uses the modem to make the dial-up connection, and another script to exchange data such as your username and password. Then, so that the system can find the domain name server, edit the `/etc/resolv.conf` file.
- You might want to set up a TCP/IP network to run a network for your home or small business. To set up a network, information such as IP addresses, gateway addresses, and hostnames for each computer attached to the network needs to be added to (depending on your particular distribution) the `rc.inet1`, `rc.inet2`, `/etc/hosts`, `/etc/networks`, `/etc/hosts.conf`, and `/etc/resolv.conf` files.

Choosing a Text Editor

There might be a dozen text editors on the CD-ROM for your favorite Linux distribution, and there are many more available for download off the Internet. At some time during your Linux career, you will require the services of a text editor. Knowing how to work a text editor from the command line might seem like a waste of time, but you might have an emergency (such as a system crash!), and you'll need the aid of a text editor to recover. Table 8.1 lists several text editors that you'll find bundled with most Linux distributions.

TABLE 8.1 Linux Text Editors

Editor	*Description*
Code Commander	Used by programmers to edit source code
Cooledit	Another text editor (with a graphical interface) for programmers that supports the Python programming language
emacs	The most powerful of the GNU text editors which supports text editing, programming, time management, and games
xemacs	emacs with a graphical user interface
gedit	A text editor for the GNOME desktop environment
GXedit	A text editor with a graphical interface that contains a spell checking utility and can send text as an email message
jed	A small text editor that can emulate emacs and WordStar
xjed	A version of jed that works with the X Window System
joe	An easy to use text editor that uses WordStar keybindings
nedit	Uses a graphical interface and supports different languages and text processors
vim	An improved version of vi that allows several windows to be displayed at one time and multiple undo levels
yudit	Another text editor for the X Window System that supports text editing in different languages

The most popular text editor is `vi` because it is one of the oldest text editors and because it migrated to Linux from the UNIX world. There are many flavors of `vi` and you will most likely find one of the `vi` clones installed on your system. The list of clones includes `vim`, `elvis`, and `nvi`. If you use the `vi` command to start `vi`, one of these clones will appear if `vi` is not installed on your system. Depending on the Linux distribution, one of the `vi` clones is installed and a symbolic link is created from the `vi` command to the command that starts the default text editor. Linux-Mandrake uses `vim`. Slackware defaults to `elvis`. The `vim` text editor is used as the example during this hour.

The quest for a command-line–driven text editor does not end with `vi`. Several other text editors are worth mentioning. A portable and lightweight text editor that is good to have around for emergencies is the `ed` text editor. The executable file for `ed` can be installed on a floppy disk, giving you access to a text editor in the case of a system crash. The `ed` text editor can be used if the X Window System is not working, or if you cannot get terminal emulation. A text editor that works similar to `ed` is `sed`, a stream editor.

Another text editor that is installed with most distributions is `emacs`. The `emacs` editor is probably the most powerful application installed with any Linux distribution. Not only is `emacs` available for Linux, but it is available for most any other operating system imaginable. There is even a graphical version of `emacs` called `xemacs` that runs under the X Window System. In addition to being a text editor, `emacs` is also a programming tool, an appointment book, and a collection of Internet utilities.

Task: Reading the `vi` Manual Page

Before you try out any command or program, it is a good idea to learn a little about the program before you embark. One place to look for information is the manual pages (which were discussed in Hour 6, "Looking at the Command Line"). In addition to covering all the commands you'll use on your Linux system, the manual pages also include information on many programs. Follow these steps to learn how to search the manual pages for information on text editors:

1. Display the `vi` manual page. Type **`man vi`** and press Enter. The manual page for the `vi`-compatible text editor installed on your system displays in a pager.

2. Read the manual page. Pay particular attention to how to use the online help system. If you need assistance while working in the `vi` text editor, use the `:help` command.
3. Read the manual pages for any of the editors discussed earlier in this hour. For example, if you are interested in `emacs`, type **`man emacs`** and press Enter.
4. Search for other text editors that might be installed on the system. Type **`man -k text editor`** and press Enter. Browse the list and look for entries that are similar to the following:

```
gEdit (1)     - GTK+ based text editor
mcedit (1)    - Full featured terminal text editor for Unix like systems.
vim (1)       - Vi IMproved, a programmers text editor
xedit (1x)    - simple text editor for X
```

5. Read the manual pages for any other text editors that might be installed on the system.

Getting Started with the `vim` Text Editor

The `vi` text editor (and its clones `vim`, `elvis`, and `nvi`) will edit any file in ASCII text format. You'll find `vi` most helpful when editing system configuration files and editing programs. While working in `vi`, you can always call on the online help system to keep you moving forward (press Escape, type **`:help`**, and press Enter). The `vim` text editor contains a few enhancements over `vi`, such as the ability to undo edits, view multiple files, and search for text.

If you're not sure about testing out `vi` from the command line, there are a couple of `vi` alternatives. If you are working in a GUI (such as KDE or GNOME), try out the graphical version of `vim`. Open a terminal emulator and type **`gvim`**. The `vim` text editor opens in a window, lists all available commands in the menus, and uses a toolbar for frequently used commands.

When you first start `vi`, it might seem complex and unmanageable. With a little patience, and some memorization practice, you'll be moving around a file and executing keyboard commands in no time. If you started a new file by typing **`vi practice.txt`**, your screen would be similar to the one shown in Figure 8.1.

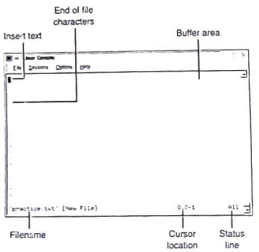


FIGURE 8.1
The vim text editor as it should appear on your screen.

Notice the cursor at the upper-left corner. Press the i key and you can begin typing. The text will appear before the cursor as you type. Like most word processing programs, vi has text-wrapping capabilities. You don't need to press the Enter key at the end of each line. The default line width is 80 characters and vi automatically moves the cursor to the next line when this width is reached. While you are typing, you can use the Backspace key to correct any mistakes that you might make. When you reach the end of a paragraph, press the Enter key to start on a new line.

NEW TERM The vi text editor works under two different modes: the *Command mode* and the *Insert mode*. To begin working in a file by adding text, you'll need to change to Insert mode. To move the cursor around the file or to delete text, you'll work in Command mode. You also need to be in Command mode to access Insert mode. Press the Escape key when you want to change to Command mode. To access a mode, you type a single letter command. After you are in a mode, several mode commands make it easy to edit text.

When you first open a new file with vi, you are in Command mode. Command mode enables you to type commands to perform tasks. The first command you'll encounter is i, which switches to Insert mode. To execute this command, just press i on the keyboard (do not use the uppercase letter I). Notice that the "—— INSERT ——" message appears at the bottom-left corner of the screen. After you are in Insert mode, you can begin adding text to a file. Correct your mistakes with the Backspace key. When you want to switch back to Command mode, press the Escape key.

Command mode also enables you to move the cursor around in a file. You can use the arrow keys to move the cursor up, down, to the right, and to the left. The k, j, l, and h keys perform the same movement.

> While you are practicing in the insert mode in vi, do not use the arrow keys, the Page Up and Page Down keys, or the Home and End keys to move the cursor around in a file. If you use these keys and find strange characters in your file, press the Escape key (at least once to ensure that you get to Command mode), and type `:q!`. This closes the file without saving the changes. By doing this while you are practicing you can ensure that you don't inadvertently save bad data to the file.

You should also save the file on a regular basis. You never know when there might be a sudden power outage. Commands that open and save files can only be used in Command mode. To access these commands, first type the colon (:) character. While you are working on a file, you can quickly save your changes (also known as writing a file) by typing `:w` and pressing Enter. At the bottom of the screen, you'll see a message similar to the following:

```
"practice.txt" 778L, 25518C written
```

If you decide you want to work with another file, type `:e filename` and press Enter. You must save changes to the file in the buffer area before you can open a new file. If you do not save your changes first, you will see an error message.

When you are finished working in vi, type `:q` and press Enter. The text editor will close and you will be returned to the shell prompt. To make things a little simpler, you can save the changes and quit vi at the same time; type `:wq` and press Enter. You will get an error if you try to quit without saving the file unless you use the `:q!` command discussed in the caution above.

The Syntax for vi

When you open a file in vi, the cursor is located at the beginning of the file. If you want to start working at some other location, you can specify that spot and vi will open to it. The following code shows the syntax for vi:

```
vi [+linenumber] [+/pattern] filename
```

The *filename* parameter is the name of the file you want vi to open when the program starts. For example, if you want to work with a file named *practice.txt*, type the command `vi practice.txt`.

The *linenumber* parameter (see Table 8.5) is the line where the cursor should be placed when the file opens in the buffer. In vi, you can display the line numbers along the left margin. Line numbers are useful if you are writing programming code or a legal document. For example, to start working at line 31 in the file named *practice.txt*, use the command `vi +31 practice.txt`.

The *pattern* parameter is used when you want vi to search for a string of text. When vi finds this text string, the cursor is placed at the first instance where the text string is found. For example, if a section of the text file where you want to start working contains the text string "introduction," use the command `vi +/introduction practice.txt`.

Task: Creating a Text File

Before you can practice with the vi text editor, you'll need a file in which you can practice cursor movement and text editing commands. One way to follow along during the rest of the hour, and to have a second source of help, is to copy the tutorial that was installed with vim. The following steps guide you through the process of creating a new file and adding some text to the file:

1. Look for the vim tutorial file. Look in the `/usr/doc/` directory for a `vim-common-??/` subdirectory. On a Linux-Mandrake system, this file is located at `/usr/doc/vim-common-5.6/tutor/tutor`.
2. Change to the directory that you want to use as your working directory (use the `cd` command). You might want to use your home directory or a subdirectory within your home directory.
3. Open vi and create a practice file at the same time. Type `vi practice.txt` and press Enter. This file will be stored in the working directory.
4. Add the contents of the tutorial file (which you located in step 1) to the practice file. Type `:r /usr/doc/vim-common-5.6/tutor/tutor` and press Enter. If the tutorial is located in another directory, make sure to use the correct directory path. Your file should look like the one shown in Figure 8.2.

> In Hour 7, "Moving Around the Filesystem," you learned how to use the `cat` command to append (or insert) the contents of one file to the end of another file. In vi, the `:r filename` command performs the same job.

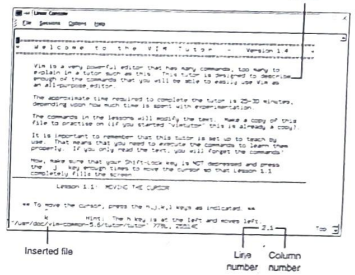


FIGURE 8.2
The VIM Tutor provides a quick start to learning your way around the text editor.

5. Save the file. Type `:w` and press Enter.
6. Make a copy of the file. Type `:w practice2.txt` and press Enter. This is similar to the `cp` command discussed in Hour 7.

You now have two copies of the tutorial file stored in your working directory. You can use these files to practice the text editing commands covered during the rest of this hour. We also recommend that you read the tutorial and follow the lessons. The extra practice with vim will help you memorize keyboard commands and cursor movements.

Text Editing Basics

The first time you use vi, you might feel overwhelmed by the vast assortment of keyboard commands needed to operate the text editor. Just remember, even though vi is a complex text editor that requires you to memorize a bunch of commands, it becomes easier to execute commands and edit text the more often you work with the editor.

Positioning the Cursor in a File

Memorizing the cursor movement commands shouldn't take much time; there aren't many. The simplest way to move the cursor around the screen is to place your fingers on the home row of the keyboard. Come on, you remember typing class. The home row is the middle row of letters that starts with the A key on the left. You'll need your right hand to move the cursor. Try these cursor movement commands (but remember to press Escape first):

- Press h to move the cursor one character to the left
- Press l to move the cursor one character to the right
- Press j to move the cursor down one line
- Press k to move the cursor up one line

Sometimes speed is a necessity and you might need to move through a file a little quicker than a single line at a time. To scroll down the file one screen at a time, press Ctrl+F. Then scroll back up with Ctrl+B. There are a few more cursor movement commands. Look at Table 8.2 for a description of the common commands used to move the cursor around in a buffer window.

TABLE 8.2 Using Cursor Commands to Move Around the Buffer

Cursor Movement	*Command*
One character to the left	h
One character to the right	l
Down one line	j
Up one line	k
Move to the next word	w
Move to the previous word	b
To the end of line	$
To the beginning of the line	0
To the next sentence	)
To the previous sentence	(
To the next paragraph	}
To the previous paragraph	{

TABLE 8.2 continued

Cursor Movement	*Command*
Scroll forward one screen	Ctrl+F
Scroll backward one screen	Ctrl+B
Scroll forward one-half screen	Ctrl+D
Scroll backward one-half screen	Ctrl+U
Go to end of the file	G
Go to line 10	10G

Inserting and Deleting Text

Press the Escape key. Did the system beep? If it did, you're in Command mode and you're ready to give the command to insert some text. Well, that is, after you move the cursor to the place where you want the text inserted. After the cursor is in position, press the i key (use lowercase) and begin typing the text. To move the cursor to a new location, press Escape and use the cursor commands to move around. You'll find more ways to insert text in Table 8.3.

TABLE 8.3 Inserting Text into a File

To Insert Text	*Use This Command*
Before the character cursor	i
At the beginning of the line	I
After the cursor	a
To the end of the line	A
On a new line below the cursor	o
On a new line above the cursor	O
Overwrite existing text	R

There are several commands that will delete text, whether you need to delete a single character or a block of text. The basic command for deleting text is d followed by one of the cursor movement symbols to show what should be deleted. A few commands used to delete text are shown in Table 8.4.

TABLE 8.4 Deleting Text from the Buffer Area

To Delete	*Use This Command*
From the cursor to the end of a word	dw
The three words following the cursor	3dw
From the cursor to the end of a line	d$
From the cursor to the beginning of a line	d0
From the cursor to the end of a paragraph	d}
The line on which the cursor rests	dd
Current line and 3 following lines	4dd
The character under the cursor	x
The character to the left of the cursor	X
Undo unwanted changes	u

Copying Text

Organizing a document involves quite a bit of moving text around. To move text around in vi, you must first yank the text, move the cursor to the new location for the text, and then paste the text in the new location.

NEW TERM When you *yank* text, you are selecting it (although you might not see the text highlighted on the screen) and making a copy of the selected text. The yanked text is stored in an unnamed buffer. This unnamed buffer is a temporary holding pen where only the last piece of yanked text is kept.

To yank text, move the cursor to the beginning of the text that you want to copy, press the y key followed by the cursor movement command (found in Table 8.1) that will move the cursor to the end of the text. For example, to copy the text from the cursor position to the end of the line, press **y$**. To position the cursor in the place where you want to copy the text, use the cursor movement commands. After the cursor is in place, paste the text by pressing p. The text will be copied on the line below the cursor.

If you want to move text, first delete the text to be moved (use one of the commands found in Table 8.3). Then move the cursor to the place where you want to paste the text and press the p key.

Searching and Replacing

One final way to edit your text file is to find a word or a group of words quickly and to replace those words with other words. You might need to replace text to give more meaning to your document. Or, information might need to be updated and you have to search for an old version number of a software application and replace it with the new version number. You might also want to search the file to locate places in the document that need to be double-checked.

When you want to search for a specific word, press Escape to enter Command mode, type a slash (/) character, type the word you for which you are searching, and then press Enter. For example, to find all instances of the word "fox" in the file, type `/fox` and press Enter. The vi text editor starts the search from the cursor location and looks for the first instance of the word "fox." When a match is found, the cursor is placed on the first letter of the word. To find the next occurrence of "fox," type / and press Enter.

If you need to look for a word and then replace it with another word, the command is quite different. If you want to find all instances of the word "fox" and replace them with the word "dog," press the Escape key to enter Command mode, type `:%s/fox/dog/g`, and press Enter. This command makes the search-and-replace procedure automatic. The colon (:) character tells vi that you are about to type a command, the `%s` parameter tells vi to search the entire file, and the `g` parameter tells vi to replace all instances of the word "fox" with the word "dog." When vi has finished replacing the text, a message similar to the following appears at the bottom of the screen:

```
3 substitutions on 3 lines
```

If you want to replace only the first match found by vi, do not use the `g` parameter. So, to replace only the first occurrence of the word "fox," type `:%s/fox/dog` and press Enter.

You'll find the same type of command line history list as you saw in Hour 6. After you type the colon (:) character, press the up-arrow key. The last command you typed in vi will appear on the command line. You can use the Backspace key to make any changes.

Task: Configuring vi

There are a number of options within vi that you can set to control the editing environment. These options are enabled and disabled with the `:set` command, and range from listing line numbers along the left side of the screen to displaying the cursor position. Table 8.4 shows some of the configuration changes you can make. To enable any of these options, press Escape to enter Command mode, type the option (as shown in Table 8.5), and press Enter. For example, to display line numbers along the left side of the screen, use the `:set number` option. Then, to disable an option, type `no` before the option. So, when you want to turn off line numbering, use the `:set nonumber` command.

TABLE 8.5 Control the Editing Environment with Configuration Settings

Option	*Description*
`:set backup`	Keep a copy of a file after overwriting a file
`:set incsearch`	Show matches for partially typed search commands
`:set lines=X`	Change the number of lines displayed on the screen at one time (*X* is the number of lines)
`:set number`	Show line numbers along the left side of the screen
`:set ruler`	Show the cursor position in the status line
`:set scroll=X`	Number of lines to scroll the window at a time (*X* is the number of lines)
`:set visualbell`	Use a visual signal (displays as `^[` on the status line) instead of a beep

When you change options with the `:set` command, the configuration change is only temporary. When you close the program, the configuration changes are not stored. The next time you use vi, the default configuration will be used. To change the default configuration, you will need to edit the `.vimrc` file. Here's how to use vi to change the default configuration of the program:

1. Find the configuration file in your home directory. At the shell prompt, type `ls -a` and press Enter. You should see a file named `.vimrc` or `.exrc` in your home directory.
2. Make a backup copy of the `.vimrc` file. Type `cp .vimrc .vimrc.original` and press Enter.
3. Open the configuration file in vi. Type `vi .vimrc` and press Enter. A sample `.vimrc` file is shown in Figure 8.3.

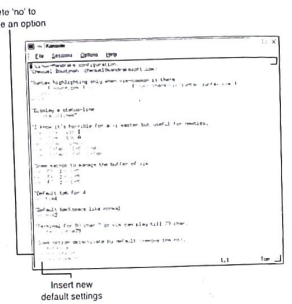


FIGURE 8.3 *Edit and add options to change the default settings for vi.*

4. Use the j key to read the configuration file line by line. You should read this file so that you understand the current vi configuration.
5. Change vi so that it automatically makes a backup of any file you accidentally overwrite. Scroll down to the section that reads:

```
"Some option desactivate by default (remove the no).
set nobackup
set nohlsearch
set noincsearch
```

   Change the `set nobackup` option so that it reads:

```
set backup
```

6. Change the default configuration so that line numbers display in each open file. Type the following (starting on a blank line):

```
"Display line numbers to the left of the text
set number
```

7. Save the file and exit vi. Type `:wq` and press Enter.
8. Open a file in vi to see the configuration changes. Figure 8.4 shows an example of a file that uses line numbers.

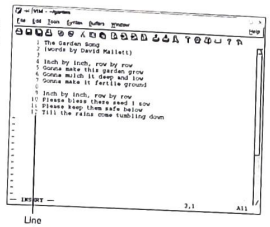


FIGURE 8.4 *Line numbers displayed in the graphical version of the vim text editor.*

Summary

It's been a long hour spent behind a text editor, and you've just barely scratched the surface. You might want to go back and commit to memory some of the editing commands you learned about. These commands make it easier to create files, move around within a file, and edit text. Then, you might want to explore some of the reasons for using a text editor. Go back and look at the list of configuration files that can be easily changed with just a few keystrokes. You might want to learn a little more about these files, and then try a few configuration changes. And, always remember to make a backup copy of a file first.

Q&A

Q Where can I find information about Linux text editors on the Internet?

A The most complete source of text editors is available from the Metalab Linux Archive at `metalab.unc.edu/pub/Linux/apps/editors`. Another good source is the Dave Central Linux Software Archive at `linux.davecentral.com/officeed.html`. Dave offers downloads of many Linux text editors in several formats (RPM, source code, and binary files) and provides a link to the home page for the editor.

You might also want to try the home pages for the individual text editors. You'll find vim at `www.vim.org` and GNU Emacs at `www.gnu.org/software/emacs`.

Q Where can I find more information about the programs that are installed on my Linux system?

A Many applications store help files in the `/usr/doc/` directory. You can read these files with the `less` and `more` pagers. You might also find a manual page for installed programs.

Workshop

Test your knowledge of the material covered in this hour by answering the following questions and working through the exercises.

Quiz

1. The basic job of the text editor is to edit ASCII plain-text files. Plain-text files are used for a variety of purposes. A text editor can be used to compose email correspondence, keep a diary file, and edit configuration files.

   The files that will probably be edited most often are configuration files. There are two types of configuration files—those that control the configuration of an individual user account (also known as startup files), and those that control how the system operates (called system configuration files). Startup files configure how programs behave when a user accesses his account. System configuration files control such items as the boot process, access to filesystems and peripherals, and screen resolution and color depth.
2. You should always try to learn a little something about a program or command before you use it. A quick and easy way to do this is to read the manual page for the program or command.

   It does not matter which vi clone is installed on a system, it will be executed by typing `vi` on the command line. Different versions may have some special features available if you care to dig deeper. Depending on the Linux distribution, there is a symbolic link between the vi command and the command that starts the vi clone. For example, Linux-Mandrake uses the `vim` text editor; therefore, there is a symbolic link between the `vi` and `vim` commands.
3. The vi text editor uses two modes—Command mode and Insert mode. When a file is first opened, vi is in Command mode. Command mode enables you to move the cursor around a file and delete text. To add text to the file, shift to Insert mode (just press the lowercase I character on the keyboard). To get back to Command mode, press the Escape key.

This means that the floppy drive is located at `/dev/fd0`. Also, look for the lines that describe your CD-ROM drive. Here's an example:

```
hdb: ATAPI 24X CD_ROM drive, 128kB Cache
Uniform CDROM driver Revision 2.56
```

2. Place the floppy disk in the floppy disk drive.
3. Mount the drive. At the command line, type

   ```
   mount /dev/fd0 /mnt/floppy
   ```

   (where `/dev/fd0` is the device listed in the `dmesg` command from step 1) and press Enter. You'll see the drive light appear for a few seconds.
4. Copy a file to the floppy disk. Type **`cp filename /mnt/floppy`** and press Enter. You'll see the drive light go on for a few seconds while the file is copied to the floppy disk.
5. Unmount the floppy drive. Type **`umount /dev/fd0`** and press Enter.
6. Take the floppy out of the drive. It is important that you unmount the drive before you remove the floppy disk.

If unprivileged users cannot access the floppy drive, you'll need to change the permissions. As superuser, type **`chmod o+rw /dev/fd0`** at the command line and press Enter.

Mounting and unmounting a CD-ROM works the same way. To mount the CD-ROM, type **`mount /dev/hdb /mnt/cdrom`** and press Enter. You should see the following message:

```
Mount: block device /dev/hdb is write-protected, mounting in read-only
```

After the CD-ROM is mounted, you can read directory lists and view files using the commands you learned about in Hour 7, "Moving Around the Filesystem."

Working with DOS Floppies

If all this mounting and unmounting stuff has you confused, there's another way you can work with floppy disks. In the distribution, you'll find a package called `mtools`. The `mtools` package contains a collection of commands that enable you to work with directories and files on a floppy disk in the same way that you work with floppies on a DOS (or Microsoft Windows) system. Also, you don't have to use the Linux filesystem (at `/mnt/floppy` or `/dev/fd0`) to access the floppy disk. You can use the old, familiar A:

designation for the floppy disk drive. When using `mtools`, you don't need to mount a floppy before you can use it. The following list contains some of the `mtools` commands that you'll find useful.

- To list the contents of the floppy disk, use the `mdir` command.
- To copy a file from a floppy to the GNU/Linux system or from the GNU/Linux system to the floppy, use the `mcopy` command.
- To delete files on the floppy disk, use the `mdel` command.
- To rename a file located on the floppy disk, use the `mren` command.
- To change the file attributes for a file, use the `mattrib` command. Using this command, you can make a file read-only.
- To format a floppy disk, use the `mformat` command.
- To check for bad blocks (errors) on a floppy disk, use the `mbadblocks` command.
- To create a label on a floppy disk that describes the contents of the floppy, use the `mlabel` command.

Formatting a Floppy Disk

When you buy floppy disks, they usually come preformatted. These disks can be used right out of the box on your GNU/Linux system. When you want to completely erase a floppy, you might want to reformat the floppy so that all the information stored on the floppy is destroyed and the floppy appears to be brand new. Here's how to format and work with a floppy by using the `mtools` commands:

1. Place the floppy disk in the floppy disk drive.
2. Format the floppy. Type **`mformat a:`** and press Enter. If you see the following error message

   ```
   mformat: Non-removable media is not supported
   (You must tell the complete geometry of the disk,
   either in /etc/mtools or on the command line)
   ```

   you will need to type

   ```
   mformat -s 18 -h 2 -t 80 a:
   ```

   and press Enter. This command (used to format a 1.44MB floppy disk) tells the `mformat` command that there are 18 sectors, 2 heads, and 80 tracks on the disk. You can find this information in the `/etc/fdprm` file. You can read this file using the `cat` command or in the `less` pager.

3. Label the floppy. Type **mlabel a: '*description*'** and press Enter. Where *description* is the text you want to appear as the label. For example, to call a floppy Testing, type **mlabel a: 'Testing'** and press Enter.
4. View the contents of the floppy. Type **mdir a:** and press Enter. You'll see the following:

```
[rusty@localhost rusty]$ mdir a:
 Volume in drive A is Testing (abbr=TESTING          )
 Volume Serial Number is 1EC7-50FF
Directory for A:/

No files
                        1 457 664 bytes free
```

In this example, the label set in step 3 is TESTING (shown on the line that reads, "Volume in drive A is Testing"). There are no files on the disk, and all 1.44MB are free.

5. Copy a file to the floppy. Change to the directory where the file you want to copy is located. To copy a file, type **mcopy *filename* a:** and press Enter. To see whether the file has been copied to the floppy, type **mdir a:** and press Enter.
6. Copy a file on the floppy to the GNU/Linux system. Change to the directory in which you want to store the file. Type **mcopy a:*filename*** and press Enter.

Locating Hardware Information

A big job for the person acting as the system administrator for your GNU/Linux system (probably you) is to maintain configuration of the system to take full advantage of the flexibility and power of the operating system. By being aware of what things are being configured for your system by the various configuration files, you can track system performance and reconfigure your system to operate at its optimum. You will accomplish this performance fine-tuning by editing the configuration files that control the way that the system acts. Before you can really get a handle on this, however, you'll need to get together all the information about the system that you can.

Task: Gathering Important System Information

Do you remember those messages that scrolled by on the screen before the login prompt appeared? They are the kernel boot messages and they are stored in the `/var/log/messages` file. The messages that display depend on the drivers that are

configured on the system. Other files, located in the `/proc` filesystem, contain information about the system architecture; that is, all the busses, boards, controllers, and peripherals attached to the system. This is important information to have before you make any configuration changes to the system. Here's how to view these files as read-only files in the `vi` text editor:

1. Log in as superuser. Type **su** and press Enter. You will be prompted for the root password. Type the password and press Enter.
2. Open the kernel boot message file as read-only. Type **vi -R /var/log/messages** and press Enter. The kernel boot messages are different, from system to system, depending on the drivers that are compiled as modules or loaded into the kernel. As you browse through the file, you'll notice messages for each device as it is initialized during the boot process.
3. Look for system configuration information. You'll need to know the screen font and color when the screen is used as a text terminal, PCI bus and PCI cards, CPU type and processor speed, system memory usage, type of mouse, serial and USB ports, mounted filesystems, and the version number of the Linux kernel.
4. Move around the file. Press Ctrl+F to move forward in the file one screen. Press Ctrl+B to move backward one screen.
5. When you are finished reading the file, exit (or quit) the text editor. Type **:q** and press Enter. The `vi` text editor will close and you will be returned to the shell prompt.
6. Return to your user account. Type **su *username*** and press Enter.
7. View information about the CPU type and speed. Type **vi -R /proc/cpuinfo** and press Enter. The **-R** command option tells `vi` to open the file as read-only. An example of the `/proc/cpuinfo` file is shown in Listing 9.3. When you are finished reading the file, type **:q** and press Enter to close the file and exit `vi`.

LISTING 9.3 The Contents of the `/proc/cpuinfo` File

```
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 5
model           : 2
model name      : Pentium 75 - 200
stepping        : 12
cpu MHz         : 132.770406
fdiv_bug        : no
hlt_bug         : no
sep_bug         : no
f00f_bug        : yes
```


LISTING 9.3 continued

```
coma_bug        : no
fpu             : yes
fpu_exception   : yes
cpuid level     : 1
wp              : yes
flags           : fpu vme de pse tsc msr mce cx8
bogomips        : 264.60
```

8. Display total system RAM, used memory, free memory, and swap space. Type **vi -R /proc/meminfo** and press Enter. Type **:q** and press Enter when you are done.
9. List the filesystems that are mounted at startup. Type **vi -R /proc/mounts** and press Enter. Do not close `vi`.
10. View the information about the PCI devices installed on the system. Type **:e /proc/pci** and press Enter. The **:e** command tells `vi` that you want to work with another file. Type **:q** and press Enter when you are done.
11. List occupied I/O addresses, IRQs, and DMA channels. Type **vi -R /proc/ioports /proc/interrupts /proc/dma** and press Enter. You can open several files in `vi` at one time by listing the filename, separated by a space. Only one file can be displayed at a time.
12. Read the `/proc/ioports` file and when you are finished, display the `/proc/interrupts` file. Type **:next** and press Enter. Use the **:next** command to move forward through the list of files; use the **:prev** command to move backward through the list of files.
13. Read the last file and exit `vi` when you are finished.

Adding New Hardware

During the installation, you created different filesystems and corresponding mount points for the partitions on the hard drive. If you think you're running out of storage room, you might want to consider adding a second hard drive to the system. Here are the steps required to add a disk to the system:

- Make sure that the piece of hardware is compatible with the operating system. You'll find this information in the Linux Hardware Compatibility HOWTO. There are some other HOWTOs that you might want to read: The Hard Disk Upgrade mini-HOWTO, the Filesystems HOWTO, the CD-ROM HOWTO, the Zip Drive mini-HOWTO, the Jaz Drive HOWTO, the Plug-and-Play HOWTO, and the CD Writing HOWTO.

- Turn off the system and install the hard drive according to the manufacturer's directions. The manufacturer usually does a low-level format, but if not, you will need to take care of this. Remember to keep notes in your Linux diary so that you can refer to them on that slight chance that something goes wrong.
- When you turn the computer on and the system reboots, watch the boot message. You'll see the new device file for the hard drive listed.
- Your next big job is to create one or more partitions on the hard drive. Use the Linux `fdisk` utility to make the Linux partitions. After the partitions are created, use the `fsck` command to check the filesystem for errors.
- Configure the device in the `/etc/fstab` file. You can change parameters in the file so that filesystems are mounted automatically at boot time and to tell the kernel where in the filesystem the device stores files.
- When the `/etc/fstab` configuration file is set up, filesystems can be mounted automatically at boot time or mounted with the `mount` command.

Throughout this hour, you've listed filesystem information by viewing the `/proc/mounts` file, executing the `mount` command, and reading the kernel boot messages. All this information is stored in the `/etc/fstab` file. A sample `/etc/fstab` is shown in Listing 9.4. You can easily view this file with the `cat` command (type **cat /etc/fstab** and press Enter). The `/etc/fstab` configuration file contains the information required to mount filesystems when the system is booted.

LISTING 9.4 The Contents of the `/etc/fstab` File

```
/dev/hda5       /               ext2    defaults                        1 1
/dev/hda1       /boot           ext2    defaults                        1 2
none            /dev/pts        devpts  mode=0620                       0 0
/dev/hda7       /home           ext2    defaults,usrquota               1 2
/mnt/cdrom      /mnt/cdrom              supermount fs=iso9660,dev=/dev/cdrom 0 0
/mnt/floppy     /mnt/floppy             supermount fs=vfat,dev=/dev/fd0 0 0
none            /proc           proc    defaults                        0 0
/dev/hda6       swap            swap    defaults                        0 0
```

Your listing might not show the contents of the file in neat and orderly rows. Listing 9.4 has been modified to illustrate the six fields required for each device that is attached to the filesystem and listed in the `/etc/fstab` file. The entries in the file use the following format:

- The first column is the device file. This file is the go-between for the user and the kernel. Various applications and commands use this file to access the piece of hardware, such as a hard drive or Zip drive.

Developing a Backup Plan

The importance of backing up the system can never be stressed enough. You never know when the power may go out or a hard drive may crash. Even though you can restore the operating system from the distribution CD-ROM, there are other files that you need to consider. What about the configuration changes that you worked on for hours and hours before your system was set up just perfectly?

Always make a copy of configuration files before you make any configuration changes. Then, once you've made the changes, you will also want to keep a copy of the new configuration file.

Then there are the files created by users. These files change on a daily basis (okay, some of us change files by the minute). Save yourself the pain, agony, and frustration by creating a sound plan for backing important files and restoring the system after a disaster.

NEW TERM *Backing up* files is almost the same as copying files, except that backed up files can be easily restored to the system in case files are lost. Backing up makes a special set of files that, when restored, places the files in their original position in the directory structure.

Follow these steps to create a backup plan:

1. **Make a list of the files and directories that need to be backed up.** You'll always want to back up system configuration files in the `/etc` directory, other configuration files that may be found in `/usr/lib`, X Window System configuration files usually found in `/usr/X11R6/lib/X11`, and kernel sources in `/usr/src/linux`. In addition, you'll want to back up user files in the `/home` directory as well as files for the superuser in the `/root` directory.

Read the "Upgrading Your Linux Distribution" mini-HOWTO for more ideas on how to design a solid plan for backing up your Linux system.

2. **Find a few tools to use when backing up and archiving files and directories.** Several Linux tools are available that will archive a group of files, and there are tools that will compress files and archives. When looking at the Linux archive and compression utilities, you'll need to decide whether you want to use the archive on just a Linux machine or if a compressed file will be used on a computer running the Microsoft Windows or Apple Macintosh operating system.

3. **Decide how often the system and individual files need to be backed up.** You'll need to look at how often files change. If files change frequently, you may need to perform a backup every day. If you only make one or two configuration changes on occasion, you can easily backup the configuration files only when the change is made.
4. **Select a storage media that will be used to store the file.** If you have a few files to back up, you could just store them on a floppy disk. If you have more files, consider using a Zip drive or a recording CD-R/W drive. Linux has always worked well with tape drives. Even though tape backups might be on the way out, there is still a lot of use for older equipment.
5. **Find a safe place to store the files.** The safest place to store the backup media is at a location different from where the computer is located. This location also needs to be protected from fire and other hazards. A favorite storage place for many people is the safe deposit vault at their bank. Others may use a fireproof safe that is located at a secured storage facility. You may also want to keep a copy of the backups close by so that you can quickly restore lost files.

Task: Performing a Simple Backup

The easiest way to back up a small number of files is to copy them onto floppy disks. The simplest method is to use the `cp` command. If the file is too large for a floppy, it can be compressed with the `gzip` command. When several files need to be backed up, create an archive with the `tar` command and compress the archive with the `gzip` command. Try this simple backup routine to copy a single file onto a floppy disk:

1. Place the floppy disk in the floppy disk drive. If you need help formatting the disk, go to Hour 7, "Moving Around the Filesystem." If you buy formatted disks in bulk from your favorite warehouse club, there's no need to format these disks to use them on your Linux system. Just put them in the drive and you're ready to go.

   Before you can copy the file to the floppy, you need to mount the drive. To do so, use the `mount` command in the following format:

   ```
   mount /dev/fd0 /mnt/floppy
   ```

2. Change to the directory that contains the files you want to back up. Use the `pwd` command to see where you are, the `cd` command to move from directory to directory, and the `ls` command to view the list of files in a directory.
3. Copy a file to the floppy disk. Use the `cp` command in the following format:

   ```
   cp filename /mnt/floppy
   ```

   In Figure 10.1, our imaginary user, rusty, has copied a file to a floppy and then verified the contents of the floppy disk.

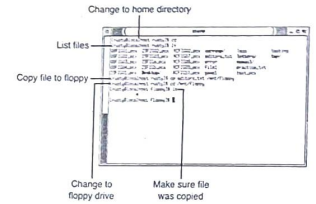


FIGURE 10.1
Floppy disks are an inexpensive backup media and easy to maintain for a small number of files.

Another compression program that you may want to try is Bzip2. You can learn more about Bzip2 by reading the Bzip2 mini-HOWTO.

Working with Storage Devices

You have several ways to store your backups. The choice you make depends on how much information needs to be stored, your computer budget, and how much time you want to take to configure devices on the Linux system.

Floppy disks have been used as a backup medium since the early days of computing. Floppy disks are inexpensive (and they come already formatted), easy to store, and every computer comes with a floppy drive, with the exception of some Macintosh setups that only have zip drives. Floppy disks are reasonably reliable, and you can lengthen the life of a floppy by storing it correctly. Floppy disks work well if you only have a few configuration files to back up, or if you only want to backup certain files in your home directory.

When storage space is a consideration, you may want to consider using removable disks on a Zip or Jaz drive. These devices allow you to store over 100MB of files and are more reliable than floppy disks. There is also a utility called `GtkZip`, shown in Figure 10.2, that can make managing Zip disks easier. `GtkZip` can also set a security level on Zip disks.

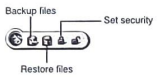


FIGURE 10.2
Use your Zip drive to store backups with the `GtkZip` tool.

Another backup alternative is a second hard drive. You can easily copy files (or the entire system) from the Linux drive to the spare hard drive. Because hard drives are not prone to failure until after considerable use, a hard drive that is only used to store backups may be a very reliable storage medium. Then when it's time to restore the system, the hard drive can do it much faster than other media.

At about the same price as a hard drive is a read/write CD-R drive. This device allows you to make a one-time backup that cannot be overwritten. You should use a CD-R drive if you need a permanent record of your backup files. You can also archive as much as 650MB of files on a single CD.

Your choice of storage devices for your important backup files depends on how much room you'll need to store files, what storage devices are already attached to the computer system, and how much your budget can afford a new storage device. Before you consider a storage device, you may want to read the following HOWTOs:

- If you need help setting up and using a Zip drive, read the "Zip Drive" mini-HOWTO. Most Linux distributions will automatically detect any parallel port versions of the Zip100 and will install the necessary drivers and create a mount point. The SCSI and IDE internal drives are common now and they are treated like hard drives. If you have a Jaz drive, read the "Jaz Drive" HOWTO.
- CD writers have recently become affordable to the average home computer user. The "CD-Writing" HOWTO explains the CD writers that are compatible with Linux, how to set them up, and what software you can use to write data files to a CD as well as how to record music onto a CD.
- If you want to back up your entire hard disk onto a second hard drive, read the "Hard Disk Upgrade" mini-HOWTO. This procedure allows you to just swap hard disks if the primary hard disk fails.

Task: Saving Disk Storage Space

One way to reduce the size of a file is to use a compression utility. You may have a file that is too large to fit on a floppy disk. If so, use the `gzip` command to compress the file enough so that it fits. You may have files in your home directory that you use infrequently. If so, save some space and compress the files. Try these compression tricks:

1. Compress the file by using the `gzip` command as follows:

   `gzip -v filename`

   This command tells `gzip` to compress the file and to display the compression percentage for the file. The original file is replaced with the compressed file, and the extension `.gz` is added to the end of the filename. For example, user rusty has three files that are not used often and that take up space (the file listing appears in Figure 10.1). When rusty executes the command

   `gzip -v 23FIG??.pcx`

   the following appears on the screen:

   ```
   23FIG01.pcx:        76.1% -- replaced with 23FIG01.pcx.gz
   23FIG02.pcx:        76.1% -- replaced with 23FIG02.pcx.gz
   23FIG03.pcx:        76.1% -- replaced with 23FIG03.pcx.gz
   ```

   This listing shows the original filename, the percentage of compression, and the new filename of the compressed file.

2. Uncompress the file by using the `gunzip` command as follows:

   `gunzip filename`

   For example, if rusty wanted to decompress the file named `23FIG01.pcx.gz`, he would type this:

   `gunzip 23FIG01.pcx.gz`

   The file is restored to its original, uncompressed state and can be used by an application that can read files of that type.

3. Compress every file in a directory by using the `gzip` command as follows:

   `gzip -rv directorypath`

   If rusty wants to compress the contents of the `corresp` subdirectory located in the user home directory, the following command would be used:

   `gzip -rv /home/rusty/corresp`

4. Uncompress every file in a directory by using the `gunzip` command as follows:

   `gunzip -r directorypath`

   When rusty wants to use the files in the `corresp` subdirectory, the following command needs to be executed:

   `gunzip -r /home/rusty/corresp`

   All the files in the subdirectory are now in their original state.

If you email files, you can use `gzip` to create a compressed file. You can then attach this file to an email message. Your recipient should be able to uncompress the file no matter which operating system he or she is using.

Taking Extra Precautions

One of the most important accessories you can buy for your computer is an uninterruptible power supply (UPS). A power failure or brownout can occur at any time, and when these occur, it is very possible that you could lose important data. Many of the available UPS systems provide surge protection and power buffering as well. Another key consideration for attaching a UPS to the computer is that the Linux operating system can be severely damaged if it is not shut down properly. For this reason, it is very important that you always use the `shutdown` command to turn off the Linux system.

A UPS gives you anywhere from 5 to 15 minutes to shut down the system when the power goes out. This is great if you're sitting behind the computer screen when the lights go out. In these instances, when you hear the UPS start beeping, log out of any user accounts and turn off the computer.

What do you do when you're away from the computer and you've left the computer running when the power goes down? Most UPSs are packaged with a serial cable and software that can be used to shut down the system when the UPS detects a reduction in power. If you use a UPS with the Windows operating system, you may already be using this setup. You can do the same thing with the Linux system. If the UPS did not come with software for Linux, read the "UPS" HOWTO for more information and then head over to `metalab.unc.edu/pub/Linux/system/ups` to find software that will run your UPS.

Task: Archiving Your Home Directory

One directory that should be backed up frequently is your home directory. The files in your home directory are used most often and changed frequently. The easiest way to back up your home directory is with the `tar` command. The `tar` command combines several files into one file and saves the original directory structure. Here's how to create an archive of your entire home directory:

1. Change to the directory that you want to archive. Use the `cd` command to move between directories.

> You may want to compress all the files in the directory before you create the archive. Use the `gzip` command to compress the files.

2. Place a floppy disk in the floppy disk drive.
3. Use the `tar` command to create an archive on a set of floppies. Use the following command syntax:

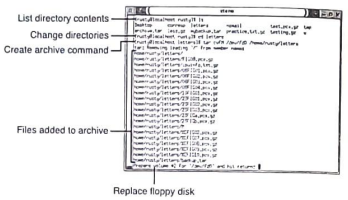
   ```
   tar cvfM /dev/fd0 /home/username
   ```

   This command tells `tar` to create a new archive, list the files on the screen as they are placed in the archive, and create the archive on the floppy disk drive. To make this example shorter, rusty is only going to archive a subdirectory within the home directory by typing this:

   ```
   tar cvfM /dev/fd0 /home/rusty/letters
   ```

   As the files are copied to the floppy disk, each file is listed on the screen, as shown in Figure 10.3. When the first disk is full, you'll be prompted to remove the floppy disk and place a second floppy disk in the drive.
4. Remove the first disk, place the second floppy disk in the drive, and press Enter. The rest of the files in your home directory will be copied to the floppy disk. Depending on the number of files you have to back up, you may be prompted to replace the disk in the drive. When the archive is finished, you'll be returned to the shell prompt.
5. Label the backup disks with the disk number and the date.

FIGURE 10.3
Back up the contents of your home directory into a single archive to make restoring your files a snap.



> Use the write-protect tab on the floppy disk so that the contents cannot be accidentally deleted or changed.

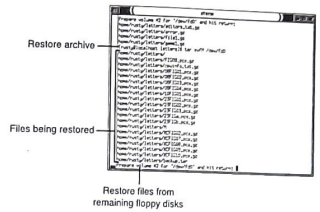
6. Restore the archive to your home directory. Place the first disk in the floppy drive and type this:

   ```
   tar xvfM /dev/fd0
   ```

   The `tar` command will begin copying files from the floppy drive and writing them to the appropriate directories (that is, the directories in which they were originally stored on the hard drive when you created the archive). See Figure 10.4 for an example. When the files from the first floppy disk are copied, you'll be asked for the second, third, and fourth disks. When the restoration of the archive is complete, you'll be returned to the shell prompt.
7. After the archive has been restored, place the floppy disks in a safe place in case you need them again.

> When you restore a `tar` archive, any files contained in the directory that use the same name as the files in the `tar` file will be overwritten without warning.

FIGURE 10.4
When you need to restore the contents of your home directory, use your floppy archive.



Summary

In this hour, you learned how important it is to plan a backup schedule and to maintain backups of important files. You learned how you could simply use the `cp` command to make a copy of a file on a floppy disk. You also learned how to compress files with the `gzip` command and how to create file archives with the `tar` command. Now that you have a few tools, it's time to create your own backup plan that will help you keep your business running even when your computer stops.

Q&A

Q Are there any other backup commands or utilities I can use to back up my files?

A The `cpio` command works much the same way as the `tar` command, but with different command options. There's also the `zip` command for compressing files. A popular utility is the BRU2000 backup system (available at `www.estinc.com`), which will work with a Zip drive.

Q How long should I keep copies of backed up files?

A That's a hard question to answer. You need to consider how important the data is and how frequently you access the data. This will give you some idea how long you need to keep the backed up files on the storage media. If there are certain files you only use every month, you may want to keep backups of these files for eight weeks. After the eight weeks are over, you can recycle the backup media by erasing the data on the media and using it for your next backup. You may have files you created for a client or a project that you need to archive. You may want to store these files for several years on a long-lasting storage medium (such as a CD-R/W).

Q What if I don't know what is contained in a compressed file? Is there any way I can see what is in the file?

A You can read the contents of a compressed file (that is, any file with the `.gz` or `.Z` extension) with the `zcat` command. To use the `zcat` command, type **`zcat filename`** and press Enter. The text will appear on the screen. The `zcat` command does *not* uncompress the file; it's only a viewer.

Workshop

Test your knowledge of the material covered in this hour by answering the following questions and working through the exercises.

Quiz

1. What are the most important files on the Linux system that need to be backed up on a regular basis?

   Configuration files should be backed up before a configuration change is made and after a configuration is made. Most configuration files are found in the `/etc` directory; other configuration files are found in the `/usr/lib` directory. X Window System configuration files are usually found in `/usr/X11R6/lib/X11`, and kernel sources are located in `/usr/src/linux`. In addition, user files in the `/home` directory as well as files for the superuser in the `/root` directory need to be backed up.
2. What is the easiest way to back up a small number of files?

   The easiest way to back up a small number of files is to copy them onto floppy disks by using the `cp` command.
3. What types of storage media can you use to back up important data?

   You can use several types of media, depending on your needs and budget. The least expensive media is the floppy disk, which works great for storing just a few files or for archiving small directories. Zip and Jaz drives are probably the most useful storage media for most users. Most people could back up their home directories and important configuration files on one or two Zip or Jaz disks. When you want to back up the entire system, you might want to consider backing up to a second hard drive. This way, if a hard drive crashes, you can easily replace the crashed drive with the backup drive. Of course, this can be done with tape drives as well. Another storage medium that is gaining popularity is the recordable CD-ROM drive. Recordable CDs can contain as much as 650MB of data.

Understanding the Linux Print Services

Before you can print any documents from the cool GNU applications that you've been learning about, you need to configure a printer. Linux doesn't provide any comfortable plug-and-play, auto-detecting hardware and easy installation methods for printers. In fact, Linux has a complex printing system that allows many users to print to several computers at one time. Linux doesn't care if all the printers are busy; it can patiently hold onto a print job until the printer is ready.

When you press the print button in an application or execute the `lpr` command, as shown in Figure 11.1, the command takes the file and temporarily stores it in a holding area called a *spool directory*. The file waits in the spool directory until the line printer daemon (the `lpd`) realizes that there is a file waiting to be printed. At this point, the `lpd` makes a copy of itself, grabs the file in the spool, and waits with the file until the printer is available. Then, when the printer is free, `lpd` and the file head off to the printer where the file is printed.

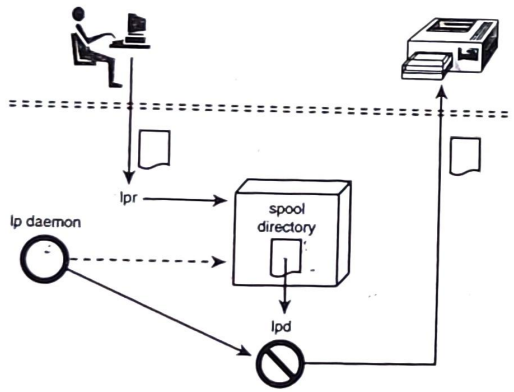


FIGURE 11.1
The line printer daemon watches the spool directory for any files that may come its way.

NEW TERM The *print spool* is a directory in the filesystem in which files are temporarily stored until they can be sent to the printer.

If there are several printers attached to the system, each printer has it's own spool directory. You would tell the `lpr` command which printer you want the file to print to by specifying the spool directory associated with that printer. The line printer daemon spawns an `lpd` for each spool directory on the system.

Task: Attaching the Printer to the Computer

Before you can configure printing services in Linux, you must attach the printer to the computer and make sure that the operating system can connect to the printer port and communicate with the printer. Before you set up printing services, follow these steps to make sure the printer is operating correctly:

1. Check the printer compatibility with the Linux operating system. One source of information is the Linux Hardware Compatibility HOWTO. You may also want to read the Linux Printing HOWTO.
2. Attach the printer to the computer. Shut down the computer and turn off the power. Attach the printer's serial or parallel port cable, plug in the printer's power supply, and turn on the printer. Turn on the computer and start the operating system.
3. Determine the port to which the printer is attached and the printer device being used. Type **`dmesg | less`** and press Enter. You may need to scroll to the bottom of the file (use the spacebar to scroll down one screen at a time) to find the printer information. Here's an example of printer information found by the kernel during the system boot:

   ```
   parport0: PC-style at 0x378 [SPP,PS2]
   parport0: Printer, Hewlett-Packard HP LaserJet 1100
   lp0: using parport0 (polling).
   ```

   Printers usually use the lp0 or lp1 printer device. The last line in the preceding listing shows that the lp0 printer device is being used by the printer. This means that the device file for the printer is named `/dev/lp0`. If your listing shows lp1, the printer device file is named `/dev/lp1`.
4. Make sure that the connection between the device file and the printer works. Type **`ls > /dev/lp0`** and press Enter. Alternatively, if the printer is connected to lp1, type **`ls > /dev/lp1`** and press Enter. The printer should print the current directory listing.

Configuring the Print System

Once the printer is attached and can communicate with the operating system, you'll need to set up the print system. This involves giving the printer a name, assigning a directory to the print spool, specifying the printer device file, and telling the line printer daemon a few specifics about the printer hardware.

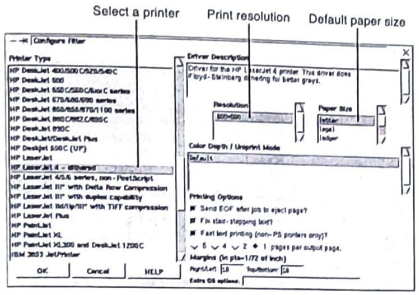


FIGURE 11.4
Choose a printer and set print options.

9. Test the printer. Select the printer from the list and click Tests, Print ASCII Test Page. The test page is sent to the print spool directory and should print on the printer. If the page did not print correctly, click Edit and change the printer configuration.
10. After you have the text test page set up they way you like, try to print a PostScript test page.
11. When you are finished, click PrintTool, Quit.

Troubleshooting the Print System

As you set up the printer and the print services, you may run into a few problems. Here are a few ideas to help:

- If the text is stair-stepped on the printed page, select the Fix Stair-Stepping Text option when you select the printer type in `printtool`.
- If your printer is not on the printer type list, select a printer that it will emulate or a printer that closely matches your printer. You may need to find the printer user manual to dig up this information.
- When printing several pages, the last page may not eject from the printer. Use the Send EOF After Job to Eject Page option.
- Other tools are available that you can use to manage your printer. Try the Linux MetaLab Archives at `metalab.unc.edu/pub/Linux/system/printing/!INDEX.html`. For those with HP LaserJet printers, there's a control panel that allows you to change the printer properties (see Figure 11.5).

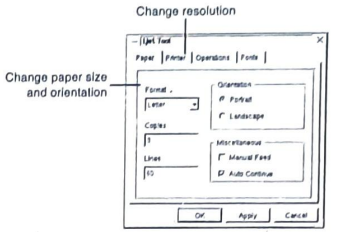


FIGURE 11.5
The LJet Tool makes it easy to print with any HP LaserJet printer.

Task: Sending Files to the Printer

Now that your printer is set up and ready to go, you'll want to start sending files to the printer. It's most likely that you'll be working in a graphical environment and you'll just press a print button in the application in which you are working. If this is the case, the file is just sent to the printer and you don't need to worry about a thing. You can also send a file to the printer by using the `lpr` command. Here's how to send files to the print spool and how to manage the print spool:

1. Send the file to the printer. To print a file, type **`lpr filename`** and press Enter. In the example shown in Figure 11.6, the user (the superuser, in this instance) listed the directory contents and then selected two files to print. This command sends the file to the print spool (for the default printer, named `lp` by default in `printtool`) where it waits until the printer is available.

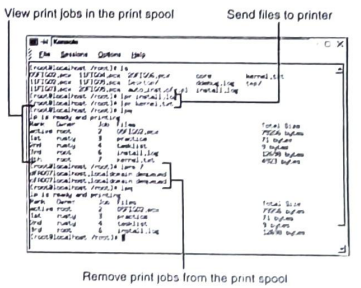


FIGURE 11.6
View the list of jobs sent to the printer and remove print jobs if necessary.

Be efficient when issuing commands. Instead of issuing two separate `lpr` commands to send two files to the printer, use one command and specify all the files. To send multiples files to the printer, type **`lpr filename filename`** and press Enter.

2. If a second printer is set up, send a file to that printer. Type **`lpr -Pprintername filename`** and press Enter. For example, to send the file named `install.log` to the printer named lp0, type **`lpr -Plp0 install.log`**.
3. Check on the status of the files in the print spool. Type **`lpq`** and press Enter. This command lists all the jobs that are either being processed by the printer or are waiting to be processed, and in the order that the jobs will be processed.
4. Delete a job from the print spool. Type **`lprm jobnumber`** and press Enter. The *jobnumber* is the number listed in the Job column of the output of the `lpq` command. For example, in Figure 11.6, to remove the file named `kernel.txt` from the print spool, type **`lprm 7`** and press Enter.
5. Clear all the print jobs that belong to your user account. Type **`lprm -`** and press Enter. If you execute this command as the superuser, all the print jobs will be removed from the print spool.

Summary

In this hour, you learned how to set up print services in Linux so that you can print the files you've created. You saw how Linux doesn't actually send a file to the printer right away. It first puts the file in a spool directory until the line printer daemon knows that the printer is available for use. You learned that each distribution uses the print utility in different ways to configure a printer and print services and that this configuration information is stored in the `/etc/printcap` file. Also, because mistakes do happen, you learned how to delete a print job from the spool directory.

Q&A

Q What are some of the applications (in addition to Ljet) that I can use to make it easier to work with my printer?

A The Ljet tool makes it easy to work with HP LaserJet printers. Also included with KDE is a print spool monitor called Printer Queue, or `klpq`. If you use KDE, you may have noticed the printer icon on the desktop. When you drag a file from the file manager and drop it onto the printer icon, the `lpr` command is executed. There's also a printer applet you can place on the GNOME panel.

Q Where can I learn more about the Linux print system?

A Start with the HOWTOs. The Linux Printing HOWTO for Linux shows how to set up a printer, configure printing software, find fax software, and preview files before they are sent to the printer. The Linux Printing Usage HOWTO tells how to print different file formats. If you have a Hewlett Packard printer, you should read the HP HOWTO

Workshop

Test your knowledge of the material covered in this hour by answering the following questions and working through the exercises.

Quiz

1. What happens when you tell Linux that you want to print a document?

   Pressing the print button in an application executes the `lpr` command. You can also execute the `lpr` command from the shell prompt. The `lpr` command takes the specified file and places it in the print spool directory for the printer that will print the file. The file waits in the spool for the line printer daemon, which spawns itself so that it can process the file and send it to the printer as soon as the printer is available.

2. Which command do you use to find out the device name of the port to which the printer is attached?

   Use the `dmesg` command. You'll probably want to view the output of the command in a pager, such as `less`, so that you can read the entire contents. The printer device information lists the manufacturer and model of the printer attached to the port and the port name (either lp0, lp1, or lp2).

3. What can you do to make sure there's a connection between the operating system, the device file, and the printer?

   Send a small text file to the printer or direct the output of a command (such as the `ls` command) to the printer device with a command like

   ```
   Ls > /dev/lp0
   ```

4. Where is the printer configuration information stored?

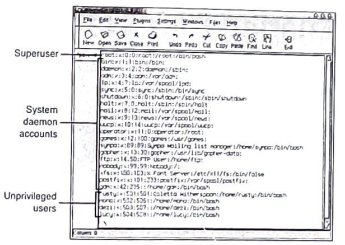
   All the printer configuration utilities store printer information in the `/etc/printcap` file. It is not recommended that you manually edit the `/etc/printcap` file because the structure is complex and you don't want to make an incorrect change.

Maintaining User Accounts

In Hour 5, "Getting Acquainted with the Shell," you learned how to use the `adduser` command to create an unprivileged user account that you could use to experiment with your Linux system. You can use this command to create as many user accounts as you need. Ideally, you should create an account for each person who uses the computer and the Linux operating system. To see the list of system users, open the `/etc/passwd` file (shown in Figure 12.1).

Please be careful when viewing and editing the files mentioned during this hour. One typographical error could cause you many headaches. One way to avoid problems is to make a backup copy of the file before you attempt to make changes. Then, if you make a mistake, you can revert back to the original file. If you feel uncomfortable editing these files, you might want to explore LinuxConf, which is covered in Appendix B, "Introducing GNOME and LinuxConf."

FIGURE 12.1
The `/etc/password` file lists all the users who have been set up in the system.



You'll notice that there are three types of users set up on the system. These are:

- Superuser—The root account is for the superuser. This account should only be used to perform system maintenance and configuration tasks.
- Individual—Unprivileged user accounts are assigned to the individuals who use the system. Each person has a username and password that are different from those of other users on the system.

- Fictitious User—System daemon accounts are accounts that are set up by the operating system and are not used by individuals. These accounts perform special tasks (such as retrieving the mail and sending files to the printer) and need access to files on the system. You can think of these system daemon accounts as fictitious users.

Each user has a separate entry in the `/etc/passwd` file that contains seven fields. Each field is separated by the colon (`:`) character. In order of appearance, these fields are

- The *username* is the name that the user types to log in to his account.
- The *password* is indicated by the letter `x`. This means that the passwords are encrypted so that someone can't look in the `/etc/passwd` file and learn everyone's passwords.

If you want to prevent a user from being able to access his account, place an asterisk (`*`) at the front of the password field.

- The *user identification number* (uid) is a unique number that is assigned to each account. The system uses this number to identify the user when dealing with file permissions.
- The *group identification number* (gid) identifies the primary group to which the user belongs. A user can belong to more than one group, but only the primary group is listed in the `/etc/passwd` file. You'll learn about group accounts later in the hour.
- *Miscellaneous information* (gecos) about the user can also be placed in the file. This could contain the user's real name and location information (such as an office number or phone number).
- The *user's home directory* (homedir) is where the user stores personal files. This is also the default working directory when the user logs in to his account.
- The *shell* that runs when the user logs in to his account is the last field in the entry. The default for most Linux distributions is `/bin/bash`.

When a user will no longer be using the system, you will want to delete his account. You have two options: You can delete the account and the user's home directory (along with its contents), or you can delete the account and leave the home directory in place. To delete a user's account and home directory, type

```
userdel -r username
```

To delete the user's account and leave his home directory in place, type

```
userdel username
```

You must be logged in to the root account to do this, and you cannot delete a user account while the user is logged in.

Task: Changing Your Password

It is very important that each user (and the superuser) account has a password and that the password is changed on a regular basis. Here's how to change the password for your user account:

1. Log in to your user account.
2. Type **passwd** and press Enter. You will be prompted for the current account password.
3. Type the password and press Enter. You will be prompted for the new password.
4. Type the new password and press Enter. You will be asked to verify the password.
5. Type the new password (again) and press Enter.

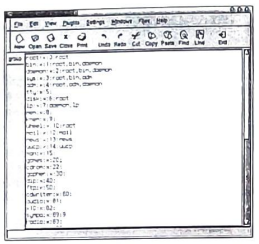
Here are a few rules about changing passwords:

- As a general rule of thumb, change your password (and the root account password) on a regular cycle, such as once a month or every three months.
- If the password becomes known by someone who does not need access to the account, change the password.
- When someone who was responsible for maintaining the system is no longer in charge of the system, change the superuser password. Some reasons for the change might be that the former system administrator has been transferred to another department, or has left the company.

Setting Up Group Accounts

In addition to setting up user accounts on your Linux system, you might want to consider setting up group accounts. Group accounts are useful when several users need to share the same files. You can create as many group accounts as needed and assign any combination of users to the group. Users can even belong to more than one group. To see the list of groups that are already set up on your system, look at the `/etc/group` file (shown in Figure 12.2).

FIGURE 12.2
The `/etc/group` file lists all the groups that have been created.



Like the `/etc/passwd` file, each group has a separate entry in the `/etc/group` file. Each entry contains four fields, with each field separated by the colon (`:`) character. In order of appearance, these fields are

- The name given to the group.
- The password field, which is optional. This field usually contains the `x` character. You can give a group a password so that users who are not part of that group can access the group's files. The outsider would need to use the `newgrp` command to access the group.
- The group identification number.
- The list of additional users who have been assigned to the group. These are users who do not have this particular group listed in the group identification field of the `/etc/passwd` file.

Task: Creating a Group Account

When you create a user account in Linux, each user is assigned to a primary group that has the same name as her username. The easiest way to add a new group is to edit the `/etc/group` file. Before you begin, you must be logged in to the superuser account to create group accounts. Here's how to create a group account:

1. Open the `/etc/group` file in your favorite text editor.

2. Scroll to the bottom of the file and create a new line. This is where you will type the information for the new group. Remember, the group information uses the following format:

   ```
   group_name:x:groupID:group_members
   ```

3. Type the name that you want to assign to the group, followed by a colon. For example, if you want to create a group in which the sales staff can store promotional literature and order forms, you might want to call the group `sales`.
4. Leave the password field blank. Type `x:`.
5. Assign a group identification number. Look at the group identification number of the last entry and use the next number. For example, if the last group in the list has a group identification number of `508`, assign the new group a group identification number of `509`. Follow this number with a colon.
6. Assign members to the group. If users named `rusty` and `mona` will be assigned to the group, separate each name with a comma (but no space between the names). Using this example, the entry for the `sales` group would look like this:

   ```
   sales:x:509:rusty,mona
   ```

7. Save the file and close it.
8. Create a directory in which the group can store files. A good place to create directories for groups is in `/usr/local`. For example, create a `/usr/local/sales` directory in which the `sales` group can store files and create subdirectories.

When an individual wants to know the groups to which he belongs, all that's necessary is to use the `groups` command. For example, user `rusty` would see the following:

```
[rusty@localhost rusty]$ groups
rusty sales
[rusty@localhost rusty$
```

This shows that `rusty` has a group all to himself and that he belongs to the `sales` group. A user can also display the identification number for his user account and assigned groups with the `id` command, as shown here:

```
[rusty@localhost rusty]$ id
uid=501(rusty) gid=501(rusty) groups=501(rusty),509(sales)
[rusty@localhost rusty]$
```

If the user is working in his account, the new group account for the user might not be listed. The user will need to log out of his account and then log back in.

If you want to delete a group account, delete the entry for the group in the `/etc/group` file.

Granting Access to Files

After you have created accounts for each user and group that will be using the system, users and groups can then start storing files on the system. When a file (or directory, for that matter) is created by a group member, that member is the owner of the file. It is this user who can grant ownership and permissions to other group members and to other system users. This system of file ownership and permissions forms the basis for restricting and allowing access to files. There are three types of file ownership:

- The owner is the user who created the file.
- The group is the default group to which the owner belongs. By default, this is the primary group specified in the `/etc/passwd` file for the file owner.
- Other ownership can be assigned to system users that are not members of the group.

NEW TERM *File permissions* determine which users on the system have access to a file and the type of access they are allowed. File permissions help maintain security for files by restricting access to files.

The file owner can also determine the type of access (or permissions) that other users may have to the file. There are three different levels of file permissions that can be assigned to a file or directory, and these permissions can be set differently for the different types of owners:

- Read access on a file allows the user, and groups that have permission, to view the contents of a file. Read access on a directory means that the user and groups can list the contents of the directory.
- Write access on a file gives selected users and groups the ability to make changes to the file or to delete the file. Write access for a directory allows the user and groups to create files in the directory and to delete those files.
- Execute access for a file means that the file can be run as a command or program. Execute access for a directory allows the user and groups to access the directory and list information about the files contained in the directory.

As you learned in Chapter 7, "Moving Around the Filesystem," the `ls -l` command lists the user and group ownership of a file and the permissions assigned to each. Figure 12.3

shows an example. In this example, the owner has full access to the file (read, write, and execute). The group, owner, and other users have read and execute permissions on the file.

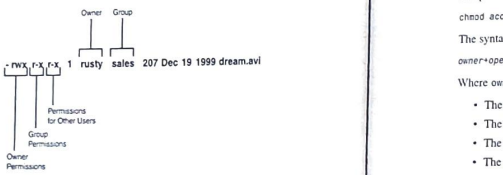


FIGURE 12.3
The owner has read, write, and execute permissions on the file.

Changing File Ownership

To change the ownership of a file (that is, the user owner and the group owner), use the `chown` and `chgrp` commands. Let's look at two files that user `rusty` has created and placed in the `sales` group directory:

```
-rw-r--r--  1 rusty   rusty   747293  Jul 4 04:04  packages
-rw-r--r--  1 rusty   rusty      155  Jul 7 01:01  tealist
```

For both files, the group ownership needs to be changed to the `sales` group. To do this, `rusty` would `cd` to the `sales` directory and use the `chgrp` command as follows:

```
[rusty@localhost sales]$ chgrp sales packages
[rusty@localhost sales]$ chgrp sales tealist
```

The `chgrp` command is followed by the name of the group to which group ownership is to be assigned and the filename. By using the `ls -l` command, you can see the changes:

```
-rw-r--r--  1 rusty   sales   747293  Jul 4 04:04  packages
-rw-r--r--  1 rusty   sales      155  Jul 7 01:01  tealist
```

If `rusty` wants to transfer the `tealist` file into the hands of user `mona`, `rusty` would change the owner with the following command:

```
[rusty@localhost sales]$ chown mona tealist
```

The listing would then appear as follows:

```
-rw-r--r--  1 rusty   sales   747293  Jul 4 04:04  packages
-rw-r--r--  1 mona    sales      155  Jul 7 01:01  tealist
```


Protecting Files

Now that the owners of the file have been set, you can change the type of access that each type of owner is permitted. The process for changing file permissions is a bit more complex. The `chmod` command changes the file permissions. The syntax of `chmod` is

```
chmod access_string filename
```

The syntax of the *access_string* is

```
owner+operator+file permission
```

Where `owner` is one or more of the following file owners:

- The `u` character sets permissions for the user owner
- The `g` character sets permissions for the group owner
- The `o` character sets permissions for all others
- The `a` character sets the permissions for all three types of owners

Where *operator* is one of the following:

- To add a file permission for the specified owner, use the `+` character
- To remove a file permission, use the `-` character
- To use the exact file permission, use the `=` character

Where *file permission* is one or more of the following:

- Read access is applied with the `r` character
- Write access is applied with the `w` character
- Execute access is applied with the `x` character

Use the following file as an example:

```
-rw-r--r--  1 rusty   sales   747293  Jul 4 04:04  packages
```

If user `rusty` wants to allow write access to the `sales` group, `rusty` would use the `chmod` command as follows:

```
[rusty@localhost sales]$ chmod g+w packages
```

To see the changes, use the `ls -l` command to display the new file permissions:

```
-rw-rw-r--  1 rusty   sales   747293  Jul 4 04:04  packages
```

You'll notice the `w` (for write access) in the group permissions.

Task: Making a File Read-Only

You might want to make a file read-only so that the contents cannot be changed, only viewed. Here's how:

1. You must be the owner of the file.
2. Change to the directory in which the file is contained.
3. Display the file for which you want to change permissions. For example:
   ```
   -rwxrw-rw-  1 mona    sales      155  Jul 7 01:01  tasklist
   ```
4. Use the `chmod` command to change the file permission to read-only for all users. Type **`chmod a=r filename`** and press Enter. For this example, type **`chmod a=r tasklist`** and press Enter. The file permissions will change to show the following:
   ```
   -r--r--r--  1 mona    sales      155  Jul 7 01:01  tasklist
   ```
5. Prevent the other users from reading the file. Type **`chmod o-r filename`** and press Enter. The file permissions for the example would by changed by typing **`chmod o-r tasklist`**. The listing would then show as
   ```
   -r--r-----  1 mona    sales      155  Jul 7 01:01  tasklist
   ```
   Notice that the other users have no permissions set for the file and, therefore, are prevented from having any access to the file.

Summary

The ability to limit access to files is important if several people use the computer and want to share files. Before users can share files, they must be assigned to groups and directories must be set up in which to store the group's files. After groups are established, group members can begin storing files in these directories and specifying the type of access they want to give to other system users who want to use the files.

Q&A

Q I don't like the username I assigned myself when I first set up the system, but I have files stored in my home directory that I don't want to lose. Is there any way I can just change the username?

A Log in to the superuser account and use the `usermod` command. You can change not only your username, but you can also change the name of your home directory to match your new username without messing up any of the files you have stored there. To use this command, type

```
usermod -l new_username -d /home/new_username -m username
```


The `-l` command option identifies the new username, the `-d` command option specifies the new directory, and the `-m` command option moves all the user's files to the new directory. For example, if a user named `marilyn` (you met her earlier) wants to change her username to `jane`, type

```
usermod -l jane -d /home/jane -m marilyn
```

Q I read the manual page for the `chmod` command and it talked about using numbers to change file permissions. Can you tell me more?

A The `chmod` command gives you two different options for dealing with file permissions. This hour showed you how to use letters to change the read, write, and execute permissions for user owners, group owners, and all other users. You might want to use a numerical approach in which `4` stands for read access, `2` for write access, and `1` for execute access. So, if you want read and write access to a file, add `4` and `2` to get `6`. Then, this access needs to be determined for each type of owner. So, if you want read and write access for the user and group owners and read access for all other users, the command would be `chmod 664 filename`.

Workshop

Test your knowledge of the material covered in this hour by answering the following questions and working through the exercises.

Quiz

1. When you use the `adduser` command to add a new user to the system, where is this information stored?

   The `adduser` command places user information in the `/etc/passwd` file. The `adduser` command uses default information to assign a user identification number, a home directory, and a shell program. The password field in the `/etc/passwd` file is encrypted so that anyone who happens to view this file will not learn the passwords of the other users.

2. What are the three types of users who can have access to a file?

   The person who created the file is known as the *owner* and is in charge of the file. This person can change the file permissions to transfer ownership of the file to other users. The *group owner* is the group to which the user owner belongs that has access to the file. You can also assign ownership to other users who are not the owner or a member of the group.

What Is a Process?

As you've wandered through the hours, testing out different commands, maybe writing a few shell scripts, you've been spawning processes and not even knowing it. In Hour 5, "Getting Acquainted with the Shell," you learned how the bash shell acts as the interface between the user and the Linux kernel and interprets the instructions from the user. For each command that the user issues, the shell spawns a clone of itself to execute the command. The clone is called a *child process*. The relationship between the shell and its clone (or subshell) is described as a *parent/child relationship*. After the command has been handed off to the child process (the subshell) for execution, the parent process (the shell) goes back and waits for another job. When it receives another command, it spawns another child process.

NEW TERM A *process* is an activity running in memory to perform the actions called for by the execution of a command. Commands can be input on the command line by the user, contained in a shell script, or input by another process running in memory like the desktop GUIs. When the process has finished its job, it is removed from memory, thereby making CPU time available for other processes. A typical system runs about 50 processes at one time and each process is assigned a unique identification number.

There are some commands for which the shell doesn't spawn subshells. These are the shell commands that are called *built-ins*. The shell itself carries out the actions called for by the built-in commands.

Types of Processes

Processes can be divided into three basic categories as far as the user and interaction are concerned:

- Interactive processes require input from the user. An interactive process may be a program that you run, such as a text editor or a graphics program.
- Batch processes execute automatically requiring no user interaction. A batch process may be a script that you write to back up your home directory at a specific time.
- Daemon processes provide services to the user and applications while running in background. You start some daemon processes at startup, such as the one that takes over when a file is sent to the printer (lpd the line printer daemon) or the PPP daemon that jumps in when you ask the system to dial your ISP and start a PPP connection to the Internet

SYNTAX

The Syntax for the ps Command

The ps command can be used along with several options to provide tailored information about the processes that are running on your system. The syntax of the ps command is

```
ps [options]
```

When the ps command is used by itself, it displays only those processes that are currently active, the process identification number, and the command as it was typed at the shell prompt. The ps command can display more information about each process by adding command options. Here are a few command options you might find useful:

- When you want to display running and stalled processes, type **ps ax** and press Enter. This might be a long list. If so, you can pipe the output to the less viewer (with the `ps ax | less` command).
- By default, the ps command shows only the command as it was typed on the command line. To see the actual command that was executed (including the full directory path to the command), type **ps e** and press Enter.
- To view just background processes, type **ps x** and press Enter.
- When you want to know which users have executed the various running processes, type **ps u** and press Enter. This listing also shows the amount of CPU and memory usage used by the process. (This command may require root access.)

Task: Creating a Process

It's time to see what this techno-speak about processes is all about and watch a few processes in action. Because you'll probably spend most of your computer time behind a graphical user environment (such as KDE or GNOME), open a terminal emulator program and learn the simple ins and outs of executing and killing a process.

1. Log in to your user account and start the X Window System to display your favorite desktop environment.
2. Open a terminal emulator program. You'll find icons on the GNOME and KDE panels or open the menus and begin looking. Figure 13.1 shows the KDE desktop using the Konsole terminal emulator.
3. At the command line, type **ps** and press Enter. The processes that are currently running display in the terminal emulator. Your output might be similar to the following:

```
[dezi@localhost dezi]$ ps
  PID TTY          TIME CMD
2666 pts/0    00:00:00 bash
2679 pts/0    00:00:00 ps
[dezi@localhost dezi]$
```

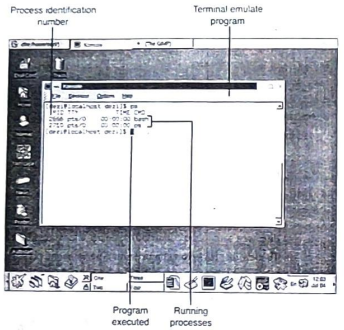


FIGURE 13.1
Processes help you manage commands that are running on the system.

- The first field (PID) is the process identification number. Each process is assigned a unique number. You'll need this number to manage an individual process.
- The TTY field is the terminal from which the process was executed. If several users are working at separate terminals, each terminal is assigned a unique TTY number.
- The TIME field shows the amount of CPU time the process has used.
- The CMD field is the name of the individual command or program that was executed by the user. The example in Figure 13.1 shows that the bash shell is running and that the ps command was used to display the output.

4. Run a program in the background. By running a program in the background, the command is executed and stays running in memory while control of the shell prompt is returned to the user. We will use a fun X program called xeyes as an example. Type **xeyes &** and press Enter. The & character tells the shell that you want the program run in the background. Your screen should look like the one in Figure 13.2.

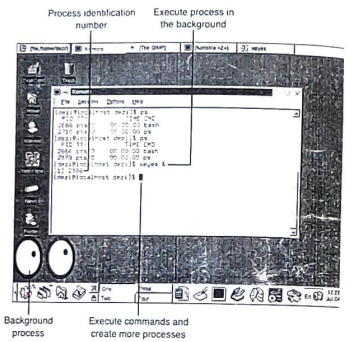


FIGURE 13.2
Run commands in the background while executing new commands from the shell prompt.

5. View the running processes. Type **ps** and press Enter. Notice that the xeyes program is listed in the output, similar to the following:

```
[dezi@localhost dezi]$ ps
  PID TTY          TIME CMD
 2666 pts/0    00:00:00 bash
 2886 pts/0    00:00:01 xeyes
 2897 pts/0    00:00:00 ps
 [dezi@localhost dezi$
```

6. Close the xeyes program. One way to close a program is to tell the `kill` command which process you want to quit. To do this, you need to know the process identification number (pid) for the command and be aware that these vary from run to run. In this example, if you want to kill the xeyes applet whose pid in the example is 2886, type **kill 2886** and press Enter. You'll see the xeyes program window close on the screen and you'll see the following in the terminal window:

```
dezi@localhost dezi]$ kill 2886
[1]+  Terminated              xeyes
[dezi@localhost dezi]$
```


7. Look at the running processes one more time. Type **ps** and press Enter. You'll notice that your original process (bash) and the most recent execution of the ps command are the only commands running.

Viewing System Memory Usage

One of the great things about Linux is that it can use computer resources to the fullest advantage. Linux uses a reserved portion of your hard drive, called *swap space*, to increase the amount of RAM memory resources available to run processes. Linux can make use of the swap space just as if it were RAM.

Because Linux runs processes in system memory (or RAM), it is important to keep track of system memory usage. Now, if you're running a powerhouse machine with 256MB of RAM and the same amount of swap space, this might never be a concern. You might have heard that Linux and the X Window System will run adequately with 16MB of RAM and 16MB of swap space and if you've chosen this route for your bare-bones machine, memory usage will certainly be one of your system management priorities.

The `free` command displays the amount of memory present on the system. This command enables you to quickly examine how physical (or RAM) memory and virtual (or swap) memory are being used. To use the `free` command, type **free** and press Enter at the shell prompt. An example of the `free` command is used in Listing 13.1.

LISTING 13.1 The Output of the *free* Command

```
[root@localhost /root]# free
             total       used       free     shared    buffers     cached
Mem:         79252      77720       1532      49972       6292      28520
-/+ buffers/cache:      42908      36344
Swap:       145112         16     145096
```

The `free` command lists memory in kilobytes (one megabyte equals 1024 kilobytes). If you would rather see this information in megabytes, type **free -m** and press Enter.

Listing 13.1 shows a system with 79 megabytes of RAM, of which about 78MB is being used and 1.5 megabytes are free. It also shows a swap space of 145MB and only 16KB of it is being used. This system is running normally.

Task: Tracking System Resource Usage

The `top` utility offers another way of viewing the processes on your machine. This utility can be used to determine which resources each process is using. It also provides an interactive interface in which you can manage processes such as changing the priority of a process or interactively killing a process. The top utility is somewhat of a resource hog itself so don't be surprised if it shows up at the top of the list. This task shows you how to use the `top` utility:

`top` has many interactive commands that you might want to try. To get the whole rundown on top, look at the manual page (type **man top** at the shell prompt).

1. Start the `top` utility. Open a terminal emulator window if you are using the X Window System. Type **top** and press Enter. An example is shown in Figure 13.3.

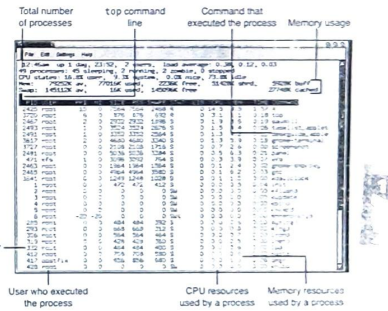


FIGURE 13.3
Use the top utility to see which processes are using valuable system resources.

2. Sort processes according to the amount of system memory being used. Type **M**. At the `top` command line, you'll see a message that says, `Sort by memory usage`. The list of process will be updated and the process that is using the most memory will be shown at the top of the list.
3. Sort processes according to the amount of CPU resources being used. Type **P**. A message that says, `Sort by CPU usage`, appears on the `top` command line. The list of processes will be updated and the process that is using the most CPU resources will be shown at the top of the list.
4. Terminate a task. Type **k** to see the message `PID to kill`. Then type the PID (or process identification number) for the process that you want to kill. Using Figure 13.3, to terminate the `xdaliclock` program, type **k** to display the message, type **3641**, and press Enter. The `top` command then asks if you are sure you want to kill the command by displaying `Kill PID 3641 with signal [15]:` on the `top` command line. Press Enter to end the process.
5. Close `top` when you are finished. Type **q**. The `top` utility will close and you will be returned to the shell prompt.

You can do more with the top utility. To see a list of options, type **h** at the `top` command line.

Other System Monitoring Tools

During the first part of this hour, you learned how to view system resource usage from the command line. If you are using the X Window System with GNOME or KDE, you'll find a number of graphical tools that enable you to view resource usage information.

If you are using the GNOME desktop, you might want to look at the GNOME System Monitor (see Figure 13.4). If you don't see the Gnome System Monitor in the menus, type **`gtop`** and press Enter in a terminal emulator.

If you are using the KDE desktop, you have two different choices. You can use the KDE Task Manager (`ktop`) or the KDE Process management utility (`kpm`). The KDE Task Manager is shown in Figure 13.5.

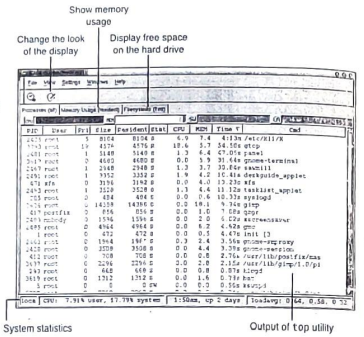


FIGURE 13.4 *The GNOME System Monitor is a graphical front-end for the top utility.*

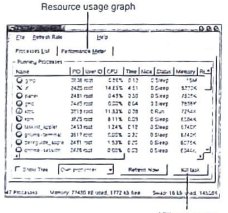


FIGURE 13.5 *The KDE Task Manager displays running processes and graphs CPU and memory usage.*

Managing Disk Space

As you work with Linux, create new files, and install new applications, you may start to run short on hard drive space. Every once in a while, you need to check and see how much free space is left. If room is becoming scarce, you may need to free up some space or maybe even start limiting the space you allocate to users.

As the Linux system administrator, your job is to make sure there is enough free space on the hard drive and in the filesystem to store files and add new applications. Because the filesystem is divided into separate partitions (or devices), you will need to know the names of each of these partitions. The `mount` command tells you which filesystems are available. An example of the `mount` command is shown in Listing 14.1.

LISTING 14.1 The Output of the *mount* Command

```
[root@localhost /root]# mount
/dev/hda1 on / type ext2 (rw)
proc on /proc type proc (rw)
/dev/hda3 on /home type ext2 (rw)
none on /dev/pts type devpts (rw,mode=0622)
```

You'll find quite a bit of information in the output of the `mount` command. For each mounted filesystem, the device name appears followed by the mount point, the filesystem type, and the access privileges. In Listing 14.1, the first filesystem (where the root partition is mounted) is located at `/dev/hda1`, it is a Linux Extended Filesystem (ext2), and has read and write privileges. To maintain this system, close attention needs to be given to `/dev/hda3`, because this is where each user's home directory is stored, and to `/dev/hda1`, because that is where the rest of the filesystem is stored.

Once you know which filesystem you need to watch, you can use the `df` command to find out the size of each partition and the `du` command to see how much room individual files and directories take up within the filesystem. Once you learn how much room is being used, you can then take corrective measures to eliminate wasted space.

The Syntax for the `df` Command

SYNTAX

The `df` command lists the amount of available (or free) disk space for each filesystem mounted on the system. The `df` command displays the name of each mounted filesystem. It displays the size of the partition, the amount of used and free space for each filesystem, the percentage of spaced used, and the mount point. Here's the syntax for the `df` command:

```
df [options] [filename]
```


When the `df` command is used by itself, it displays information for each mounted filesystem. You can use the `df` command with a filename to list the information for a single file, a device (such as a partition on the hard drive), a mount point, or a specific directory within the filesystem. Free space can be calculated for separate filesystems in these different ways:

- **List the disk space information for a specific device.** In Listing 14.1, to see the information for the device where the root filesystem is mounted, type **`df /dev/hda1`** and press Enter.
- **List the disk space information for a specific mount point.** Using Listing 14.1 again, to view information for the `/home` mount point (which is stored in `/dev/hda3`), type **`df /home`** and press Enter.

Several command options can be used with `df` to change how used and available space are displayed:

- To display the total free and used space in kilobytes, type **`df -k`** and press Enter.
- To display the total free and used space in megabytes, type **`df -m`** and press Enter.

The Syntax for the `du` Command

SYNTAX

The `du` command displays the amount of space a directory (and any subdirectories contained within the directory) occupies within the filesystem. By default, the `du` command lists the directories contained within the current working directory. Here's the syntax for the `du` command:

```
du [options] [filename]
```

When you want to display the space used by a particular file or directory, specify a filename after the `du` command. For example, to see the amount of space used by each directory within the `/home` directory and the total space used by the `/home` directory, type **`du /home`** and press Enter.

Use command options with the `du` command to change how the used space is displayed:

- To display the total used space in kilobytes, type **`du -k`** and press Enter.
- To display the total used space in megabytes, type **`du -m`** and press Enter.

Task: Monitoring Used Disk Space

You have several tools at your disposal to help you monitor how quickly your hard drive is filling up. These tools should be used on a regular basis to keep track of disk space usage and to help you decide when it is time to make room for more files. Here are a few guidelines for monitoring disk space:

1. Display the free disk space on each of the mounted filesystems. Type **df** and press Enter. An example of a system with two mounted partitions is shown in Listing 14.2.

LISTING 14.2 The Output of the df Command

```
Filesystem            Size  Used Avail Use% Mounted on
/dev/hda1             1961M 1376M  485M  74% /
/dev/hda3              988M    4M  984M   0% /home
```

2. Determine which filesystem is getting too full. A filesystem should contain between 5 and 30 percent free space. In Listing 14.2, the filesystem on `/dev/hda1` only has 26 percent free space; it is time to create some extra space. The filesystem on `/dev/hda3` is only 0 percent full (our users haven't been very busy).
3. View the amount of space being used by a particular directory. This can help you track down oversized directories. Type **du *filename*** and press Enter.

> Several directories tend to fill up fast. One is the `/home` directory, where system users store their files. The other is the `/usr` directory, where programs that are available to all users are stored. To see the disk usage for a particular user, type **du /home/*username*** and press Enter.

4. List all the directories on the system according to size. Type **du / | sort -rn** and press Enter. If the listing is too long, view the output in the `less` pager (type **du / | sort -rn | less** and press Enter). Figure 14.1 shows an example of the directory size listing displayed in the `less` pager.
5. Take the first step needed to increase free space on your system. Determine those places where there is not enough room and identify those files that could be deleted, archived, or compressed.
6. Create a regular schedule on which you check for free and used disk space. If the system users are quite active, you may want to look at their disk usage on a weekly basis.

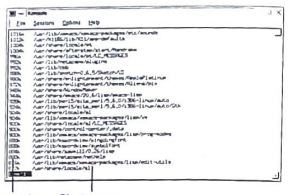


FIGURE 14.1 *The largest directories are displayed at the top of the list.*

Creating Additional Free Disk Space

When you find the filesystem filling up and you think you'll need more hard drive space, you need to take some sort of action to increase the space available for storing files and installing new applications. One solution is to add a second (or third) hard drive to the system. Here are some other ideas to help you keep as much free hard disk space on hand as possible:

- **Compress files.** There may be files in your home directory (or the home directories of other system users) that are used infrequently. Compress these files (using a utility such as `gzip`, covered in Hour 10, "Backing Up and Restoring the Filesystem") when they aren't needed. If you need to work with the file, it only takes a second to uncompress the file.
- **Delete files.** Delete files that you no longer need.
- **Archive files.** Archive files you do not use on a regular basis on a removable storage medium, such as a floppy or Zip disk.
- **Clean out any temporary directories.** Clean out directories such as the `/tmp` directory and reduce the size of system log files (these are found in the `/var/log` directory). Also, delete backup files that are automatically created by any programs you use.

> You'll learn how to reduce the size of system log files in Hour 17, "Automating System Administration Tasks."

- **Move files and delete directories.** Linux sees directories as empty files. When a directory has a considerable amount of activity, such as when files are moved around, the directory size might not shrink when the directory is empty. To get around this, move all the files out of the directory and then delete the directory.
- **Remove programs you don't use.** When you installed Linux, quite a number of applications were installed along with the operating system. If you find that you don't use some of these applications, get rid of them. Hour 15, "Installing Software," tells you how to uninstall applications.
- **Limit the amount of space.** Limit how much space users can use to store files on the hard drive.

Locating Unused Files on the System

One way to regain valuable hard drive space is to delete or back up files that have not been used in the recent past. Here's how to use the `find` command to list those files that are good candidates for the archives (or the trashcan):

1. Look for files in the entire filesystem that have not been accessed in the last 180 days. Type **find / -atime +180 -print** and press Enter. This locates all the files on the system not accessed in the last six months and displays the list on the screen. Some sample output is shown in Figure 14.2.

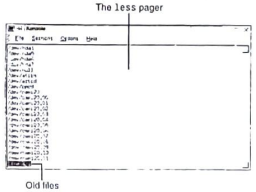


FIGURE 14.2 *Use the **find** command to locate files that have not been used recently.*

2. Search for user files that have not been used in the past 30 days. Type **find /home -atime +30 -print** and press Enter.

> If the file listing is too long, direct the output to the `less` pager or save the output to a file. To save the output to a file named `oldfiles`, type **find /home -atime +30 -print > oldfiles** and press Enter.

3. Locate backup files. Most backup files have a filename that begins with either `.BAK`, or ends with a tilde (the ~ character). To list backup files on the system, type **find / -name *.BAK.*" -o -name "*~" -print** and press Enter.

Watching User Activity

One way to keep plenty of free space in the filesystem is to manage system users more effectively. Here are a few tips that you can use to train users to become more efficient system users:

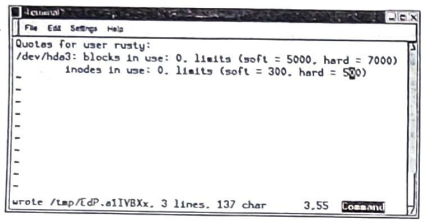
- Train users on the importance of backing up their files.
- Teach users to compress files that are used infrequently.
- Implement a system where users archive files that they are not using and delete the original files from the filesystem.
- Discourage users from installing too many applications in their home directories. Applications should be installed in the `/usr` directory so that every user has access to the program. This prevents several users from installing the same program unnecessarily.

Earlier in this hour, you learned how to look at user home directories to determine how much space they're using in the filesystem. If you have a very large hard drive and only a few users, the amount of space that is used for user files may not be a problem. When space becomes limited, you may also want to limit the space in which users can store their files.

Task: Setting Disk Quotas for System Users

Disk quotas provide one way to help prevent the filesystem from becoming overcrowded. Here's how to limit the amount of disk space in which a user can store files in his or her home directory

FIGURE 14.5
Limit storage space and the number of files for each user in the system.



12. Save your changes and close the file.
13. View your changes. Type `quota` *`username`* and press Enter. To see the disk quota for user rusty, type `quota rusty` and press Enter. The quota for rusty appears as follows:

```
Disk quotas for user rusty (uid 501):
     Filesystem  blocks   quota   limit   grace   files   quota   limit   grace
      /dev/hda3    1560    5000    7000             191     300     500
```

Task: Setting the System Clock

When you installed Linux, you told the installation program in which time zone you live. This is the information that the Linux kernel uses to display the time using the Hardware Clock. If you set the computer's clock correctly during the installation, chances are that you're staying on time. What do you do if your time is wrong? Here are a few easy tricks you can use to reset the time:

1. Log into the root account. Only the superuser can change the system time.
2. Check the time. Type `hwclock` and press Enter. The time displays in the following format:

   ```
   Sat Jul  1 19:58:28 2000  -0.261406 seconds
   ```

   This example shows the current date as July 1, 2000 and the current time of 7:58 p.m. The date is correct, but the time displayed by the hardware clock is incorrect. Oops!
3. Set the clock so that it matches the time stored in the BIOS clock. Type `hwclock -w` and press Enter. The clock is now set so that it matches the BIOS clock.

4. Check the time again. Type `hwclock` and press Enter. The time kept by the hardware clock is now set to the time stored in the BIOS clock:

   ```
   Sat Jul  1 10:00:37 2000  -0.693272 seconds
   ```

   The time that displays in this example is almost the correct time, at 10 a.m. on Saturday, July 1, 2000. The clock is just a few minutes off.
5. If the time is still wrong, set the time manually. Use the following command syntax:

   `hwclock --set --date="`*`MM/DD/YY HH:MM:SS`*`"`

   For example, to set the time to 10:15 a.m. on July 1, 2000, type `hwclock --set --date="07/01/00 10:15:00"` and press Enter.
6. Check the time again. Type `hwclock` and press Enter.

Summary

You now have a few tools under your belt to help you manage your hard drive space. In the past hour, you learned how to determine which files and users were using more than their fair share of disk space. You also learned a few tricks for keeping free space at a desirable level.

Q&A

Q How do I scan my hard drive for errors?

A Every time you boot the Linux operating system, the startup process automatically checks the hard drive for errors with the `fsck` command. If you want to use the `fsck` command, you need to unmount the filesystem that you want to check before running this command. To learn more about the `fsck` command, read the manual page (type `man fsck` at the shell prompt).

Q If I limit the amount of disk space that is available to an individual user, is there any way I can notify him or her?

A You can always tell users that you are limiting the amount of space available to them to store files and let them know what limits you set. The users can then use the `du` command to see how close they are coming to their limit. Another way is to use the `edquota -t` command. This allows you to automatically notify users that they have used up their disk space quota. The `edquota -t` command opens a file in the `vi` text editor, and you will need to specify the number of days the users can go over their limit. After the grace period, users will not be able to add any new files to their home directories.

A less common way to network your computer is by attaching one or more terminals to a computer, as shown in Figure 20.2. Using this configuration, a single computer controls more than one computer monitor through a serial port connection. A variety of terminals can be connected to a standalone computer this way—dumb terminals, text terminals, and graphics terminals. This is a very cost effective way of setting up a small network. For more information on how to connect terminals to a computer, check out the "Text Terminal" and "Serial" HOWTOs.

A serial connection is not the only way to do this. You can also use a network card and category five Unshielded Twisted Pair (UTP) wiring.

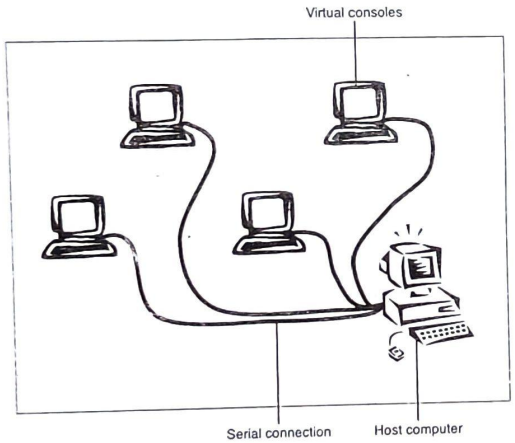


FIGURE 20.2
Attach several terminals to a computer through the serial port.

Locating Users on a Network

Because Linux is a multi-user, multitasking operating system, many people can share the resources of one computer. If the computer is attached to a network, there will be many users sharing the same network resources. If the computer is a standalone computer, several users can share the computer simultaneously using connected dumb terminals.

When all these people are logged into the system, how do you keep track of them all? An easy way to find out which users are logged into the system is with the `who` command. At a minimum, the `who` command will tell you which users are logged in, the terminals at which they're working, and how long each user has been logged in.

Listing Users on the Network

When you only need to know which users are logged in, the terminals at which they are working, and the times that they logged in, simply type `who` at the shell prompt. Listing 20.1 shows the output of the `who` command when there are five users using the system.

LISTING 20.1 Sample Output from the `who` Command

```
[rusty@localhost rusty]$ who
root     tty1     Jun 12 03:00
rusty    tty2     Jun 11 11:35
mona     tty3     Jun 11 12:09
dezi     tty4     Jun 11 11:36
lucy     tty5     Jun 11 11:36
```

Another way to find out who is using the network is with the `finger` command. The `finger` command displays the same information as the `who` command and adds the user's real name, how long the account has been idle, an office number, and a phone number. Listing 20.2 shows the logged-in users displayed by the `finger` command.

LISTING 20.2 Sample Output from the `finger` Command

```
[rusty@localhost rusty]$ finger
Login    Name         Tty  Idle    Login Time   Office     Office Phone
root     C Withers    1    05:53   Jun 12 03:00
rusty    R Spoon      2            Jun 11 11:35
mona     M Rossman    3            Jun 11 12:09
dezi     D Forearm    4     22:57  Jun 11 11:36
lucy     L Amber      5     22:53  Jun 11 11:36
```

You may wonder where `finger` finds this extra information. The user's real name is displayed if the name is contained in the user's password file. The idle time is the amount of time that has passed since the user did anything with the account (such as type something at the keyboard or click the mouse).

The Syntax for the who Command

When you want to know which users are logged into the system and how long their accounts have been active, use the who command. The following code shows the syntax for the who command:

```
who [options]
```

The who command can be used without any additional parameters, or you can add one or more of the following command options to display additional information about the logged-in users:

- When you need a quick reminder of the options available for the who command, use the shell's `--help` command option.
- When you aren't sure which user account you are logged into, use the `am i` command option to display the username, the hostname, the terminal logged into, and the amount of time you have been logged in.
- To add column headings to the output of the who command, use the `-H` command option.
- A list of logged-in users and the total number of users logged in can be displayed with the `-q` command option.
- Sometimes a user may be logged in but not actively working in his or her account. To see how long accounts have been in use or idle, use the `-i` command option.
- To see whether a user is accepting messages (which you'll learn about later in this chapter), use the `-w` command option.

Displaying User Information

The `finger` command can also be used to display specific information about an individual user. Listing 20.3 shows the information displayed for the user "dezi."

LISTING 20.3 The Results of the `finger dezi` Command

```
[rusty@localhost rusty]$ finger dezi
Login: dezi                             Name: D Forearm
Directory: /home/dezi                   Shell: /bin/bash
On since Sun Jun 11  11:36 (EST) on tty4    22 hours 57 minutes idle
No mail.
Plan:
I'll be out of the office next week. I'm taking a vacation.
Time to pull weeds and plant seeds.
```


If you want others to know a little more about you, create a Plan file using your favorite text editor. This file needs to be located in your home directory (dezi would store this file in `/home/dezi`), and the file is named `.plan`.

LISTING 20.3 The Results of the `finger dezi` Command

Sending Messages to Other Users

One way to send a quickie message to another user on the system is with the `write` command. The `write` command allows you to leave a message for another user who is logged into the system. When the other user is ready to respond to your message, he or she will also use the `write` command to send the reply.

If you are not able to use the `write` command, the superuser will need to enable messages. From the root shell prompt, type `mesg y` and press Enter.

The Syntax for the write Command

One way to send messages across a network without having to set up an email system is to use the `write` command. The following code shows the syntax for the `write` command:

```
write username [ttyname]
Your message here
Ctrl+D
```

Here, *username* is the name of the user to whom you want to send a message and the *ttyname* parameter indicates the terminal at which the user logged in. Use the *ttyname* parameter when a user is logged into more than one terminal to specify which terminal to send the message.

Sending Messages

The `write` command allows two users to communicate across the network without using email or voice mail. Before you use the `write` command to send a message to another user, you'll need to make sure that the other user is logged into the system. For example, users rusty and mona are working together on a project and rusty needs to ask mona a question. To see if mona is logged in, rusty would use the who command (by typing `who` and pressing Enter at the shell prompt) to check and see whether mona is working at the computer.

Once rusty has determined that mona is logged in, rusty would send the message by typing `write mona` and pressing Enter. The cursor appears on the next line and rusty can type the message. When rusty has finished composing the message, pressing Ctrl+D sends the message to mona. Here's what displays on the terminal at which rusty is working:

```
[rusty@localhost rusty]$ write mona
Do you have the schedule for the next two weeks?
[rusty@localhost rusty]$
```

Now rusty just sits back and relaxes until mona sends a reply.

Replying to Messages

When a message arrives from another user, you'll again use the `write` command to respond to the user. In the previous message that rusty sent to mona, the message appears as follows on the terminal at which mona is working:

```
Message from rusty@localhost.localdomain on pts/1 at 02:16
Do you have the schedule for the next two weeks?
EOF
```

Before mona can respond, the Enter key must be pressed in order to display the shell prompt. Then, mona can reply by executing a command similar to the following:

```
[mona@localhost mona]$ write rusty
You'll need to ask Dezi for the updated schedule.
```

After mona has finished composing the message, pressing Ctrl+D (the Linux end-of-file character) sends the message back to rusty.

You may want to use communication commands (such as `write` and `talk`, discussed later) sparingly when communicating with other users. Some people do not like to be interrupted while they are working. You may also want to keep your messages short; a line or two is usually enough to give short instructions, leave requested information, or request a meeting.

Sending a Message to All Users

There may be times when you need to send an important message to each and every user who is logged into the system. A system administrator may want to let everyone know when the system will be going down for repair, or the boss may want to let everyone know about an emergency staff meeting. The `wall` (write all) command sends a message to every user on the system. To send a message to every user, use the following command:


```
[rusty@localhost rusty]$ wall
Staff meeting at 3pm today.
Ctrl+D
```

As you might have noticed, the `wall` command works the same way as the `write` command. You execute the command, type a message, and press Ctrl+D to send the message. The message that is sent displays on each user's terminal, as follows:

```
Broadcast message from rusty (tty2) Mon Jun 12 02:38:37 2000...

Staff meeting at 3pm today.
```

If a user wants to reply to the broadcast message, he or she can use the `write` command to send a message to the sender.

Turning Off Messages

There may be times when you don't want to receive messages from other users. You may be talking on the telephone and don't want any distractions, or you may be processing a large file and you don't want to add a heavier load to the system resources.

When you don't want to receive messages from other users, type `mesg n` and press Enter. Whenever someone tries to contact you using the `write` or `talk` command, a message similar to the following appears on their screen:

```
write: rusty has messages disabled
```

When you are ready to accept messages again, type `mesg y` and press Enter.

Having a Two-way Conversation with Another User

Sometimes it may be more efficient to have a direct conversation with another user on the network. By using the `talk` command, you can call up another user and have a one-on-one conversation. If you visit the Internet chat rooms, you'll find the `talk` command works in somewhat the same way.

Once again, let's go back to users rusty and mona. These two users need to have a meeting but neither is able to leave his or her desk, and mona is expecting an important phone call. For these situations, mona can use the `talk` command and have a real-time, interactive conversation with rusty. To start the conversation, mona would type the following command:

```
talk rusty
```

After mona has sent the request to rusty, the screen will look like the one shown in Figure 20.3. The screen will remain like this until a response is received from rusty.

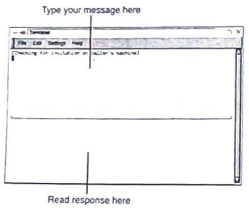


FIGURE 20.3
You will have to wait until your invited guest joins the conversation.

When rusty receives the request from mona for a little talk, a message like the following will appear on the screen:

```
Message from Talk_Daemon@localhost at 17:37 ...
talk: connection requested by mona@localhost.
talk: respond with:  talk mona@localhost
```

The last line of the message tells rusty what command to use to connect with mona. In order for rusty to chat with mona, rusty would type **`talk mona@localhost`** at the shell prompt. When rusty has responded, both terminals display a message saying that the connection was established. Now that the two users are connected, they can begin their conversation. The user who is sending the message will type the message in the top half of the screen (don't worry about placing the cursor; the `talk` command does that for you). Responses from the recipient user appear in the bottom half of the screen. An example of a `talk` conversation is shown in Figure 20.4.

The `talk` command sends everything you type to the recipient as you type it. There are no special characters to type to display the message on the recipient's terminal. When the talk conversation is concluded, press Ctrl+D to end the session.

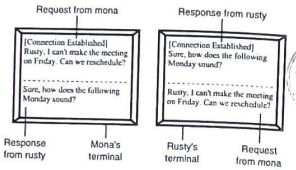


FIGURE 20.4
Looking at the terminals for mona and rusty after their conversation has started.

Using an Office Message Board

In addition to the `write` and `talk` commands, you can communicate with users over the network in other ways. The first way is to change the message that appears before the login prompt. Another way to leave messages for all users is to change the message that appears after a user logs into his or her account.

Changing the Login Message

There may be times when you want to relay information to users before they log into the system. Linux uses the `/etc/issue` file to change the information that displays on an unused terminal. You can use this file to add your company name to the login screen, to notify users of company policy regarding logins, or to post the hours that the system is available for use.

The login screen for each distribution is different. The login screen for LinuxMandrake contains the following information:

```
LinuxMandrake release 7.1B (hydrogen)
Kernel 2.2.15-0.25mdk on an 8586 / tty1
localhost login:
```

On the Slackware distribution, the login screen may look like:

```
Welcome to Linux 2.2.13
darkstar login:
```

Even though these login screens are different, each distribution uses the `/etc/issue` file to control the login message, and only the superuser can make changes to the file.

Task: Customizing the Login Screen

When you want to display a message on an unused terminal that anyone can read, you'll need to edit the `/etc/issue` file. Here's how to open the `vi` text editor and make a few changes:

1. Log in as superuser.
2. Open the `/etc/issue` file in the `vi` text editor. Type **`vi /etc/issue`** and press Enter. The file appears in the text editor and may look something like the `/etc/issue` file shown in Figure 20.5.

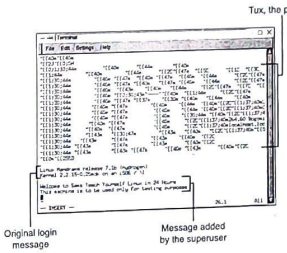


FIGURE 20.5
The original /etc/issue file from the Linux-Mandrake distribution with an added login message.

3. Place the cursor at the end of the file. Press the J key to move the cursor down a row.
4. Type the message that you want to appear on the login screen.
5. Save the file and exit `vi`. Press Esc, type **`:wq`**, and press Enter.
6. Your message won't update until you log out and back in again. Change to an unused terminal (press Alt plus a Function key) and take a look at your changes.

Creating a Message of the Day

Another way to leave messages for users is to create a message of the day that users will see as soon as they log into their accounts. This message appears after a user logs in and just before the shell prompt. In order to create a "message of the day" file, open the file named `/etc/motd` in a text editor and add any text you want. If this file does not exist, you'll need to create it. Figure 20.6 shows a sample `/etc/motd` file.

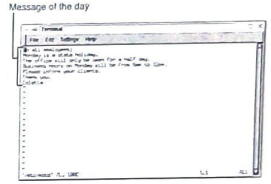


FIGURE 20.6
Create a message of the day file to inform employees or family members of important events.

When creating a message of the day file, keep the message short so that it will fit on a terminal screen. Also, remember that this message appears after the user has logged in.

Summary

During the past hour, you learned how to be nosy and find out who's using the same computer/network as you are. Once you know who's on the system, you can use a couple of easy commands to get in touch with those users or leave them a message. You also learned how to edit the `/etc/issue` and `/etc/motd` files to leave messages for other users.