Learning Red Hat Enterprise Linux & Fedora, 4th Edition

By Bill McCarty

Publisher: O'Reilly
Pub Date: April 2004
ISBN: 0-596-00589-X
Pages: 352

New in this edition are installation instructions and help with package updating for Red Hat Enterprise Linux and Fedora. Information on the GRUB bootloader, and the CUPS printer system, as well as the Publisher's Edition of Fedora are also included.

Learning Red Hat Enterprise Linux & Fedora, 4th Edition

By Bill McCarty

Publisher: O'Reilly
Pub Date: April 2004
ISBN: 0-596-00589-X
Pages: 352

Printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly & Associates books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (http://safari.oreilly.com). For more information, contact our corporate/institutional sales department: (800) 998-9938 or corporate@oreilly.com.

Preface

You've probably heard about Linux from a magazine, radio or TV program, or a friend. You're wondering what Linux is about and whether you should give it a try. If so, particularly if you currently use Microsoft Windows, this book was written for you.

When the first edition of this book was being written, Linux was a much talked about novelty. Today, Linux has invaded corporate information technology departments, becoming a popular technology used by hobbyists and professionals alike.

As predicted in the first edition of this book, Linux is becoming easier to use. If you work with Microsoft Windows and have dabbled a bit in MS-DOS or are curious about what happens inside Windows, you already have the skills to install and configure Linux. Thousands of people from all walks of life have already done so.

Recently, Red Hat?the distributor of the most popular Linux release, Red Hat Linux?split its product line. This book explains both resulting Linux distributions: Red Hat Enterprise Linux and Fedora Core. It includes the two-CD publisher's edition of Fedora Core, containing everything you need to install and configure your own Linux system. This book will make your Linux journey easier, by giving you the big picture, providing you with step-by-step procedures, and getting you started doing useful or fun activities, such as word processing or games. This book focuses on the needs of the new Linux user and on desktop Linux applications. You'll learn about networks and servers, but the details of those topics are left for more advanced books.

Organization of This Book

Chapter 1

This chapter is designed to introduce you to Linux and help you determine whether Linux is appropriate for you.

Chapter 2

This chapter helps you understand what's involved in installing Linux and guides you through the process of gathering information about your system necessary to successfully install Linux.

Chapter 3

This chapter takes you step-by-step through the installation of Linux.

Chapter 4

Before you can effectively use a desktop environment, you need to know some Linux fundamentals. This chapter explains basic Linux concepts that underlie graphical and nongraphical system use.

Chapter 5

This chapter explains how to configure and use the GNOME and KDE desktop environments.

Chapter 6

This chapter introduces you to OpenOffice.org, Linux's desktop suite, which includes a word processor, spreadsheet, presentation graphics, and more. The chapter also introduces you to Evolution, a popular Linux email client and personal scheduler.

Chapter 7

This chapter digs deeper into the *bash* shell, the Linux command-line interface. You'll learn how to use the Linux command-line interface, which resembles MS-DOS but is much more powerful and sophisticated. Here you'll see firsthand just how powerful and easy to use Linux can be.

Chapter 8

This chapter explains the RPM Package Manager, which helps you manage program and applications. It also explains how to use Red Hat Network to keep your system up to date.

Chapter 9

Administering a multiuser operating system such as Linux is somewhat more complicated than administering a single-user operating system, but Linux includes tools that simplify the work. This chapter shows you how to configure your Linux system, including how to configure sound and printers.

Chapter 10

This chapter shows you how to connect via your Internet Service Provider (ISP) to the Internet. Once connected, you can use your Linux system to surf the Web and access other familiar Internet services.

Chapter 11

This chapter shows you how to connect your Linux system to other systems on your Local-Area Network, and how to install and configure servers such as Apache, the world's most popular web server.

Sources of Information

If you are new to the world of Linux, there are a number of resources to explore and become familiar with. Having access to the Internet is helpful, but not essential.

Red Hat's Web Site

Your primary resource for information on Red Hat Linux is Red Hat's web site, http://www.redhat.com. The Red Hat web site includes more resources than can be mentioned here. The final section of Chapter 3 describes several of the most useful resources. Two particularly important resources are:

The redhat-install-list mailing list

http://www.redhat.com/mailing-lists

Here, you can obtain installation assistance from members of the Red Hat Linux community.

Bugzilla

http://bugzilla.redhat.com

Bugzilla is a database that lists possible bugs affecting Red Hat Linux. The database often gives fixes or workarounds for bugs.

Linux Documentation Project Guides

The Linux Documentation Project (LDP) is a group of volunteers who have worked to produce books (guides), HOWTO documents, and manual pages on topics ranging from installation to kernel programming. More works are in development. For more information about the LDP, consult their web page at http://www.tldp.org, or one of its many mirrors. The LDP works include:

Linux Installation and Getting Started

By Matt Welsh et al. This book describes how to obtain, install, and use Linux. It includes an introductory Unix tutorial and information on systems administration, the X Window System, and networking.

Linux System Administrators Guide

By Lars Wirzenius and Joanna Oja. This book is a guide to general Linux system administration and covers topics such as creating and configuring users, performing system backups, configuring major software packages, and installing and upgrading software.

Linux System Adminstration Made Easy

By Steve Frampton. This book describes day-to-day administration and maintenance issues of relevance to Linux users.

Linux Programmers Guide

By B. Scott Burkett, Sven Goldt, John D. Harper, Sven van der Meer, and Matt Welsh. This book covers topics of interest to people who wish to develop application software for Linux.

The Linux Kernel

By David A. Rusling. This book provides an introduction to the Linux kernel, how it is constructed, and how it works. Take a tour of your kernel.

The Linux Kernel Module Programming Guide

By Ori Pomerantz. This guide explains how to write Linux kernel modules.

Conventions Used in This Book

The following typographical conventions are used in this book:

Bold

Used for commands, programs, and options. All terms shown in bold are typed literally.

Italic

Used to show arguments and variables that should be replaced with user-supplied values. Italic is also used to indicate new terms and example URLs, filenames and file extensions, and directories and to highlight comments in examples.

`Constant Width`

Used to show the contents of files or the output from commands.

`Constant Width Bold`

Used in examples and tables to show commands or other text that should be typed literally by the user.

`Constant Width Italic`

Used in examples and tables to show text that should be replaced with user-supplied values.

`#`, `$`

Used in some examples as the root shell prompt (`#`) and as the user prompt (`$`) under the Bourne or *bash* shell.

This icon signifies a tip, suggestion, or general note.

This icon indicates a warning or caution.

A final word about syntax: in many cases, the space between an option and its argument can be omitted. In other cases, the spacing (or lack of spacing) must be followed strictly. For example, **-wn** (no intervening space) might be interpreted differently from -**w** *n*. It's important to notice the spacing used in option syntax.

Path Notation

I use a shorthand notation to indicate paths. Instead of writing "From the Start menu, choose Find, then Files or Folders," I write: Start → Find → Files or Folders. I distinguish menus, dialog boxes, buttons, or other GUI elements only when the context would otherwise be unclear. Simply look for the GUI element whose label matches an element of the path.

Keyboard Accelerators

In a keyboard accelerator (such as **Ctrl-Alt-Del**), a dash indicates that the keys should be held down simultaneously, whereas a space means that the keys should be pressed sequentially. For example, **Ctrl-Esc** indicates that the Control and Escape keys should be held down simultaneously; **Ctrl Esc** means that the Control and Escape keys should be pressed sequentially.

Where a keyboard accelerator contains an uppercase letter, you should not type the Shift key unless it's given explicitly. For example, **Ctrl-C** indicates that you should press the

How to Contact Us

We have tested and verified the information in this book to the best of our ability, but you may find that features have changed (or even that we have made mistakes!). Please let us know about any errors you find, as well as your suggestions for future editions, by writing:

O'Reilly & Associates, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
(800) 998-9938 (in the U.S. or Canada)
(707) 829-0515 (international/local)
(707) 829-0104 (fax)

You can also send us messages electronically. To be put on the mailing list or to request a catalog, send email to:

info@oreilly.com

To ask technical questions or comment on the book, send email to:

bookquestions@oreilly.com

We have a web site for the book, were we'll list examples, errata, and any plans for future editions. The site also includes a link to a forum where you can discuss the book with the author and other readers. You can access this site at:

http://www.oreilly.com/catalog/learnredhatentlnx

For more information about this book and others, see the O'Reilly web site:

http://www.oreilly.com

Acknowledgments

Thanks to my editor, Andy Oram, who valiantly fought many battles?a few with me?in seeing this book to completion. Andy's concern to make this the best book possible was continually evident and greatly appreciated. Andy suggested many improvements and caught several errors that would have caused me serious embarrassment. Way to go, Andy! Thanks also to Margot Maley of Waterside Productions, Inc., who brought this authorship opportunity to my attention.

Several reviewers, some working for O'Reilly & Associates and some working elsewhere, commented on the manuscript and suggested helpful corrections and improvements. In particular, I'd like to thank the following people for taking time away from their busy schedules to review this latest edition: Jason Hall, Andy Oram, Chip Turner. I greatly appreciate their assistance and readily confess that any errors in the manuscript were added by me after their reviews and so are entirely my responsibility.

My family?Jennifer, Patrick, and Sara?provided their customary compassion and assistance during this latest authorship experience. Thanks, guys.

I also acknowledge the love, concern, and support of my savior, Jesus Christ. His perfect love is entirely undeserved.

PREV < Day Day Up > NEXT

Chapter 1. Why Run Linux?

Welcome to Linux, the operating system everyone's talking about. Unlike the weather?which proverbial wisdom says you can't do anything about?you *can* do something about Linux. You can run it on your own PC, so that you can see firsthand what the talk is about and perhaps contribute suggestions to its future development.

This chapter is the first leg of your journey into the land of Linux. Here, you'll learn whether this particular journey is right for you and what you can expect down the road. If you're impatient to get started, you can jump ahead to the next chapter, which helps you prepare your PC for installing Linux. But, if you'd like to know more about the history and capabilities of Linux, read on.

1.1 Why Red Hat Enterprise Linux and Fedora?

This book explains Release 3 of Red Hat Enterprise Linux WS and Release 1 of Fedora Core. As explained in the following sections, Red Hat offers several Linux products, or *distributions* as they're called. Moreover, other companies sell or freely provide Linux distributions. Why, then, does this book focus on Red Hat Enterprise Linux WS and Fedora Core?

From the standpoint of market share, Red Hat is the leading provider of Linux distributions in the U.S. and worldwide. To many people, Red Hat Linux *is* Linux. And, among the various Linux distributions provided by Red Hat, Red Hat Enterprise Linux WS and Fedora Core stand out as the most appropriate distributions for desktop users, especially those in corporate environments. To understand why this is so, it's necessary to understand more about Linux, operating systems, and open source (http://www.opensourece.org) software.

1.2 What Is Linux?

Linux is an *operating system*, a software program that controls your computer. Most PC vendors load an operating system?generally, Microsoft Windows?onto the hard drive of a PC before delivering the PC; so, unless the hard drive of your PC has failed or you've upgraded your operating system, you may not understand the function of an operating system.

An operating system handles user interaction with a system and provides a comfortable view of the system. In particular, it solves several problems arising from variation among hardware. As you're aware, no two PC models have identical hardware. For example, some PCs have an IDE hard drive, while others have a SCSI hard drive. Some PCs have one hard drive; others have two or more. Most PCs have a CD-ROM drive, but some do not. Some PCs have an Intel Pentium CPU, while others have an AMD Athlon, and so on.

Suppose that, in a world without operating systems, you're programming a new PC application?perhaps a new multimedia word processor. Your application must cope with all the possible variations of PC hardware. As a result, it becomes bulky and complex. Users don't like it because it consumes too much hard drive space, takes a long time to load, and?because of its size and complexity?has more bugs than it should. Operating systems solve this problem by providing a standard way for applications to access hardware devices. Thanks to the operating system, applications can be more compact, because they share the commonly used code for accessing the hardware. Applications can also be more reliable, because common code is written only once?and by expert systems programmers rather than by application programmers.

As you'll soon learn, operating systems do many other things as well; for example, they generally provide a filesystem so you can store and retrieve data and a user interface so you can control your computer. However, if you think of a computer's operating system as its subconscious mind, you won't be far off the mark. It's the computer's conscious mind?applications such as word processors and spreadsheets?that do useful work. But, without the subconscious?the operating system?the computer would cease breathing and applications would not function.

Pronouncing Linux

Internet newsgroup participants have long debated the proper pronunciation of *Linux*. Because Linus Torvalds originated the Linux kernel, it seems reasonable that his pronunciation of the word should reign as the standard. However, Linus is Finnish and his pronunciation of *Linux* is difficult for English speakers to approximate. Consequently, many variations in pronunciation have arisen. The most popular pronunciation sounds as though the word were spelled *Linnucks*, with the stress on the first syllable and a short *i*.

If your computer has a sound card, you can hear how Linus Torvalds pronounces Linux: http://www.ssc.com/lj/linuxsay.html. Linus's personal opinion is that how you pronounce *Linux* matters much less than whether you use it.

1.2.1 Desktop and Server Operating Systems

Now that you know what an operating system is, you may be wondering what operating systems other PC users are using. According to the market research firm IDC, Microsoft products account for over 90 percent of sales of desktop operating systems. Because Linux is a free operating system, Linux sales are a mere fraction of actual Linux installations. Unlike most commercial operating systems, Linux is not sold under terms of a per-seat license; a company is free to purchase a single Linux CD-ROM and install Linux on as many systems as they like.[1] So, sales figures understate the popularity of Linux. Moreover, it's important to consider who uses a product and what they use it for, rather than merely the number of people using it. Linux is particularly popular among power users who run web sites and databases and write their own code. Hence, though Linux *is* popular, its influence is even greater than its popularity suggests.

[1] Recently, some Linux vendors, including Red Hat, have begun bundling services with their Linux distributions. Your service agreement with such a vendor may forbid you to install unlicensed copies of the vendor's distribution or may impose penalties for doing so. See Section 1.2.7 for more information.

1.3 Reasons to Choose or Not Choose Linux

Notwithstanding its high points, Linux is not for everyone. You should approach your decision to use Linux as you'd approach any decision, by evaluating the pros and cons. Here are several reasons to run Linux:

- **You want a stable and reliable computing platform**. No popular operating system is more stable and reliable than Linux. If you're tired of crashes and hangs and the lost time and data they entail, you're a candidate for Linux.
- **You want a high-performance computing platform**. Linux can coax blazingly fast performance out of hardware below the minimum required to load and run other popular operating systems. And with ample memory and a fast CPU, Linux goes toe-to-toe with anything Microsoft or other vendors offer. If speed is your thing, Linux is your hot rod.
- **You need a low-cost or free operating system**. If you're on a budget or if you need to set up many systems, the low cost of Linux will let you reserve your hard-earned capital for hardware or other resources. Linux is the best operating system value on the planet.
- **You're a heavy network or Internet user**. If you use networks, especially the Internet, Linux's advanced support for TCP/IP may light up your life. Linux makes it easy to construct firewalls that protect your system against hackers or routers that let several computers share a single network connection.
- **You want to learn Unix or TCP/IP networking**. The best way?perhaps the only way? to learn more about Unix or TCP/IP networking (or computers generally) is through hands-on experience. Whether you're interested in such experience owing to personal curiosity or career ambition (system administrators are often handsomely paid), Linux affords you the opportunity to gain such experience at low cost, without leaving the comfort of your home.
- **You seek an alternative to Microsoft's vision of computing's future**. If you're tired of marching to the relentless drumbeat of the Redmond juggernaut, Linux offers a viable way to cut the umbilical cord and set about creating a new, open source computing destiny for yourself and others.
- **You want to have fun**. Hopefully, you've discovered that one of the best reasons for doing anything is that it's fun. Many Linux users report that they've never had so much fun with a computer. There's no better reason for running Linux than that.

To be frank, some folks shouldn't run Linux. If one or more of the following are true of you, you should run Linux *only* if you have a good friend who's knowledgeable about Linux, available by phone at odd hours, and works cheap:

- **You're scared of computers**. If you're scared of computers, you should spend more time working with Windows 2000/XP or Mac OS X before venturing into the Linux world. Linux may indeed be right for you, but it's not right just yet.
- **You don't like, or don't have the time and inclination, to tinker and learn**. Setting up and running Linux will require you to learn new concepts and skills. None of these is especially difficult, but unless you're enthusiastic about learning and playing around, setting up and running Linux will stress you out. Instead, you should stick with the familiar.
- **You're married to certain Windows applications**. You can run some Windows applications under Linux's WINE emulation, but this isn't true of every Windows application. Before putting your toe in the Linux waters, you should obtain up-to-date information on the status of WINE emulation of your favorite Windows applications (see http://www.winehq.com). Alternatively, you can purchase the commercial products VMware (see http://www.vmware.com), Win4Lin (see http://www.netraverse.com), or CrossOver Office (http://www.codeweavers.com) that enable you to run Windows applications or Windows itself under Linux.

  Rather than convert your desktop system to run Linux, you may prefer to install Linux

Chapter 2. Preparing to Install Linux

Before installing Linux, you must first gather some data about your system. This chapter presents information you need to know and tasks you need to perform before the installation of Linux. It helps you make certain that your IBM-compatible PC meets the minimum hardware requirements for Linux. It shows you how to document your Windows operating system configuration so that you can respond to questions presented by the Linux install procedure. And, finally, it shows you how to prepare your hard disk for Linux.

> Various Linux distributions have somewhat different hardware requirements. To avoid tedious repetition, I refer to Red Hat Enterprise WS 3 and Fedora as simply "Linux" throughout this chapter. But, please bear in mind that the information in this chapter may not accurately apply to Linux distributions other than Red Hat Enterprise Linux WS 3 and Fedora.

2.1 Minimum Hardware Requirements

Linux supports a wide range of PC hardware, but not even Linux supports every known device and system. Your PC must meet certain minimum requirements in order to run Linux, which I describe in the following sections. For the latest and most complete information, you should check Red Hat's hardware compatibility web site, http://hardware.redhat.com/hcl. This site will also help you determine whether Linux supports the devices installed in your system. If you're not familiar with PC hardware, check out Robert and Barbara Thompson's *PC Hardware in a Nutshell: A Desktop Quick* Reference (O'Reilly), an excellent introduction and reference to PC hardware.

2.1.1 Central Processing Unit (CPU)

Linux does not support the Intel i386 and earlier processors. However, it fully supports the Intel i486, Celeron, Pentium, Pentium Pro, Pentium II, Pentium III, and Pentium IV processors and compatible processors manufactured by others, such as AMD's Athlon, Athlon XP, and Athlon MP. Such processors are members of what is known as the *x86 family* of processors. Linux also supports non-Intel processors such as AMD's AMD64 processor, IBM's PowerPC processor, and certain processors used in IBM mainframes. And, it supports Intel's Itanium processor.

> Although Linux supports processors other than members of Intel's x86 processor family, special measures are required to install it on such processors. This book describes the installation, configuration, and use of Linux on only the x86 family of processors.

Apart from the processor model, there's the issue of processor speed. You should generally have a 400 MHz Pentium II or faster processor for satisfactory results using Linux desktop applications.

2.1.2 Motherboard

The motherboard is the main part of a PC. It holds the CPU, RAM, and other internal computer components, linked by several buses. Linux supports the standard ISA, EISA, PCI, and VESA (VLB) system buses used on most IBM-compatible PCs, as well as the AGP, and USB auxiliary buses. Fedora Core includes limited support for the IEEE 1394 (FireWire) bus. However, the related software components are largely untested.

Your motherboard should include at least 256 MB of RAM for optimum Linux performance. However, this figure is merely a guideline. Many people manage to install Linux on systems having only 192-256 MB of RAM. Some especially determined and skilled users have managed to coax Linux into working on systems with as little as 4 MB of RAM by using their own installation programs or methods. However, unless you're a skilled system administrator and programmer, it's not recommended that you attempt such an exotic installation.

> Red Hat does not support systems having less than the required amount of RAM.

A handful of motherboards present special problems when installing Linux. Generally, problems stem from bad BIOS, for which a fix is often available. Check the Red Hat web site for details; the best way to do so is via the knowledgebase search page at http://www.redhat.com/apps/support.

2.1.3 Drives

An anonymous wag once quipped that one can never be too thin, be too rich, or have too much hard disk space. Fortunately, Linux is not extremely hungry for disk space. To install and use Linux, you should have a minimum of 4000 MB (4 GB) of free hard disk space. More realistically, you should have at least 10 GB.

2.2 Collecting Information About Your System

Before you launch into the installation process, you should collect some basic information about your system. Generally, Red Hat's installer will successfully probe your system and discover its configuration, but when it fails to do so, you must be prepared to supply the required information. Otherwise, you'll be forced to terminate the installation procedure, obtain the information, and then start all over again.

2.2.1 Information You Need

Table 2-1 specifies the configuration information you need and gives you space to conveniently record the information as you gather it. If your system currently runs Windows, you can obtain much of the needed information by using Windows utilities, as explained in the next section. To obtain the remaining information, you can consult your system documentation and the documentation for any devices installed by you. If your documentation is missing or incomplete, you may need to contact your hardware vendor or manufacturer. Alternatively, you may be able to find the needed information on the manufacturer's web site; use a search engine such as Google (which is powered by Linux) to discover the URL of the web site.

Sometimes, you'll need to examine your system's BIOS settings or open your system's case and examine the installed hardware; consult your system documentation to learn how to do so. Finally, if you're installing Linux in a large organization such as a business or a university, your system administrator is likely to be an excellent source of the information you need.

Table 2-1. Configuration information needed to install Linux

Device	Information needed	Your information
Hard drive(s)	The number, size, and model of each hard drive. Which hard drive is first, second, and so on. Which adapter type (IDE or SCSI) is used by each drive. For each IDE drive, whether or not the BIOS is set for LBA mode. The number and type of each existing partition and the amount of free disk space.	
CD-ROM drive(s)	Which adapter type (IDE, SCSI, or other) is used by each drive. For each drive using a non-IDE, non-SCSI adapter, the make and model of the drive.	
SCSI adapter (if any)	The make and model of the adapter.	
RAM memory	The amount of installed RAM.	
Video adapter	The make and model of the adapter and the amount of installed video RAM.	
Video monitor	The make and model of the video monitor and the manufacturer's specifications, if available, especially the horizontal and vertical sync (refresh) rates.	
Mouse	The type (serial, PS/2, or bus).The protocol (Microsoft, Logitech, MouseMan, etc.). The number of buttons. For a serial mouse, the serial port to which it's connected (COM1 or COM2).	
Sound adapter (if	The make, chipset, and model of the adapter.	

2.3 Preparing Your Hard Disk

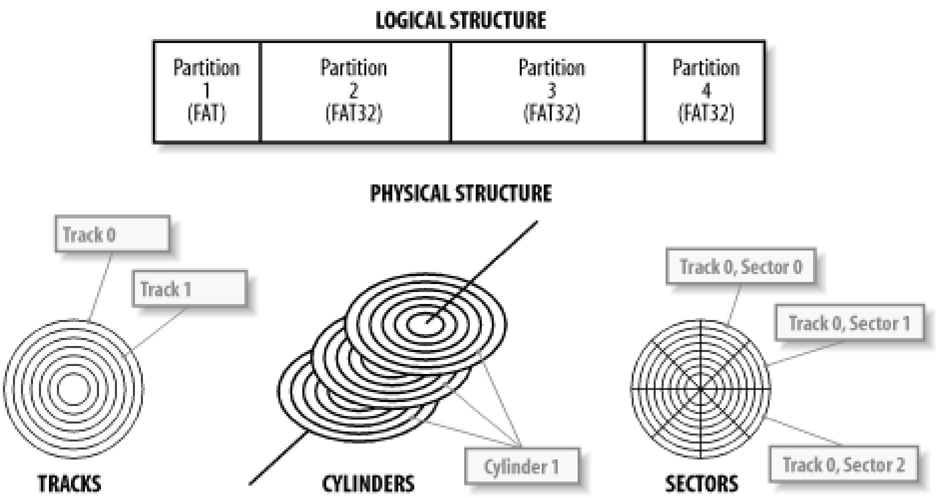
To prepare your hard disk for installing Linux, you must allocate the space in which Linux will reside. You'll learn how to do so in this section. First, I'll explain how hard disks are organized, followed by how to view the structure of a hard disk. Finally, I'll describe how to alter, or *partition*, the structure of your hard disk in preparation for installing Linux.

2.3.1 How Hard Disks Are Organized

Let's start by reviewing facts you've probably learned by working with Windows. Most operating systems, including Windows 95/98, 2000, and XP, manage hard drives by dividing their storage space into units known as *partitions*. So that you can access a partition, Windows associates a drive letter (such as *C*: or *D*:) with it. Before you can store data on a partition, you must *format* it. Formatting a partition organizes the associated space into what is called a *filesystem*, which provides space for storing the names and attributes of files as well as the data they contain. Windows supports several types of filesystems, such as FAT, FAT32, and NTFS.

Partitions comprise the *logical structure* of a disk drive, the way humans and most computer programs understand the structure. However, disk drives have an underlying *physical structure* that more closely resembles the actual structure of the hardware. Figure 2-6 shows the logical and physical structure of a disk drive.

Figure 2-6. The structure of a hard disk



Mechanically, a hard disk is constructed of platters that resemble the phonograph records found in an old-fashioned jukebox. Each platter is associated with a read/write *head* that works much like the read/write head on a VCR, encoding data as a series of electromagnetic pulses. As the platter spins, the heads record data in concentric rings known as *tracks*, which are numbered beginning with zero. A hard disk may have hundreds or thousands of tracks.

All the tracks with the same radius are known as a *cylinder*. Like tracks, cylinders are numbered beginning with zero. The number of platters and cylinders of a drive determine the drive's *geometry*. Some PCs require you to specify the drive geometry in the BIOS setup. Most modern PCs autodetect the drive geometry but let you specify a custom value if you prefer.

Most operating systems prefer to read or write only part of a track, rather than an entire track. Consequently, tracks are divided into a series of *sectors*, each of which holds a fixed number of bytes, usually 512.

To correctly access a sector, a program needs to know the geometry of the drive. Because it's sometimes inconvenient to specify the geometry of a drive, some PC BIOS programs let you specify *logical byte addressing* (LBA). LBA sequentially numbers sectors, letting programs read or write a specified sector without the burden of specifying a cylinder or head number.

2.3.2 Viewing Disk Partitions

Chapter 3. Installing Linux

This chapter shows you how to install Red Hat Enterprise Linux or Fedora Core by following a simple, step-by-step procedure. During the installation, you'll need to refer to the information you collected in Table 2-1 of Chapter 2. Most users will be able to complete the installation procedure without difficulty; however, this chapter includes a section that describes how you can obtain help if you encounter installation problems. Once you successfully complete the installation procedure, you'll have your own working Linux system.

3.1 Installing the Operating System and Applications

To install Linux, attach any optional devices you plan to use, such as a PCMCIA network card or external floppy drive, to your PC. Then, follow this simple step-by-step procedure:

1. Start the installation.
2. Select installation options.
3. Create partitions.
4. Configure the boot loader.
5. Configure networking.
6. Configure language support.
7. Configure the system time.
8. Set the root password.
9. Select packages.
10. Install packages.
11. Configure X (Red Hat Enterprise Linux only).
12. Complete the video configuration (Red Hat Enterprise Linux only).
13. Create boot diskette (Fedora Core only)
14. Complete the installation.

> Although the Linux installation procedure is generally trouble-free, errors or malfunctions that occur during the installation of an operating system can result in loss of data. You should not begin the installation procedure until you have backed up all data on your system and determined that your backup is error-free. You should also create boot floppies or other media that enable you to boot your system even if the boot information on the hard disk is damaged.

3.2 Start the Installation

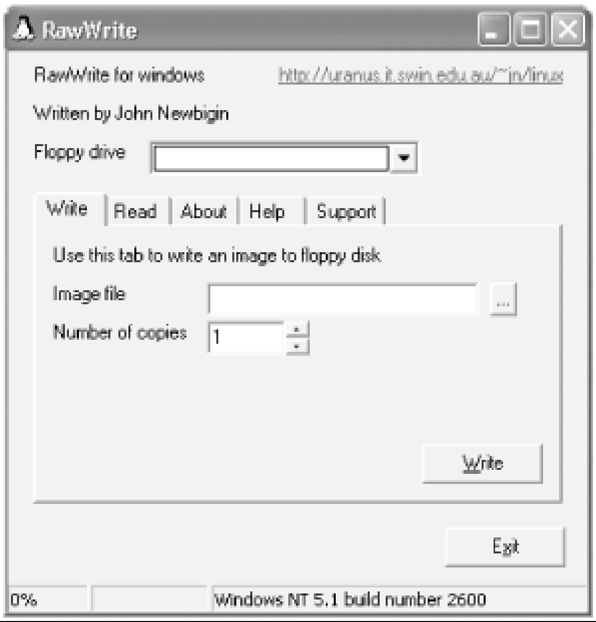
To begin installing Linux, you must boot your system from the installation media. Most recently manufactured PCs can boot the Installation CD 1 CD-ROM. However, unless you generally boot from a CD-ROM?which is quite unlikely?you'll need to reconfigure your PC's BIOS so your PC is able to boot from a CD-ROM. To do so, enter your PC's BIOS screen and look for a configuration item titled something like Boot Order or Boot Priority. Change the configuration so that the CD-ROM drive has the highest boot priority. Consult your PC's documentation for details on entering and using its BIOS configuration screens.

3.2.1 Creating a Boot Floppy

If your PC can't boot from a CD-ROM, you must create a boot floppy disk. Creating a boot floppy requires some special measures; you can't simply copy files onto a disk and then boot it. To create a Linux installation boot floppy by using a PC that runs Microsoft Windows, perform the following steps:

1. Format a floppy.
2. Insert Disc 1 of Linux into your system's CD-ROM drive.
3. Click **My Computer** and then your CD-ROM drive. Navigate to the d *:\dosutils\rawritewin* directory, where d is the drive letter associated with your CD-ROM drive. Double-click the program **rawwritewin**. The RawWrite dialog box appears, as shown in Figure 3-1. Specify the floppy drive and image file (images\bootdisk.img or other), and click **Write**. It takes perhaps a minute or so for the **rawrite** utility to create the floppy diskette.

Figure 3-1. Using rawrite to make a boot diskette



If your PC requires one or more PCMCIA or unusual SCSI devices during boot up, you must follow a somewhat more complicated procedure. See the *README* file on Installation CD-ROM 1 for details.

3.2.2 Boot the Installation Program

To start the installation process, insert Installation CD 1 of Linux into your system's CD-ROM drive. If your system cannot boot from a CD-ROM, insert the boot floppy you created and reboot your PC. When the system reboots, you should see a start-up screen featuring a **boot** : prompt and a series of messages explaining how to invoke the graphical and text mode installation and upgrade facilities, as shown in Figure 3-2. This prompt lets you enter special parameters to work around a variety of installation problems. Generally, it's not necessary to do so. Simply press **Enter** or wait about a minute and the installation program will start.

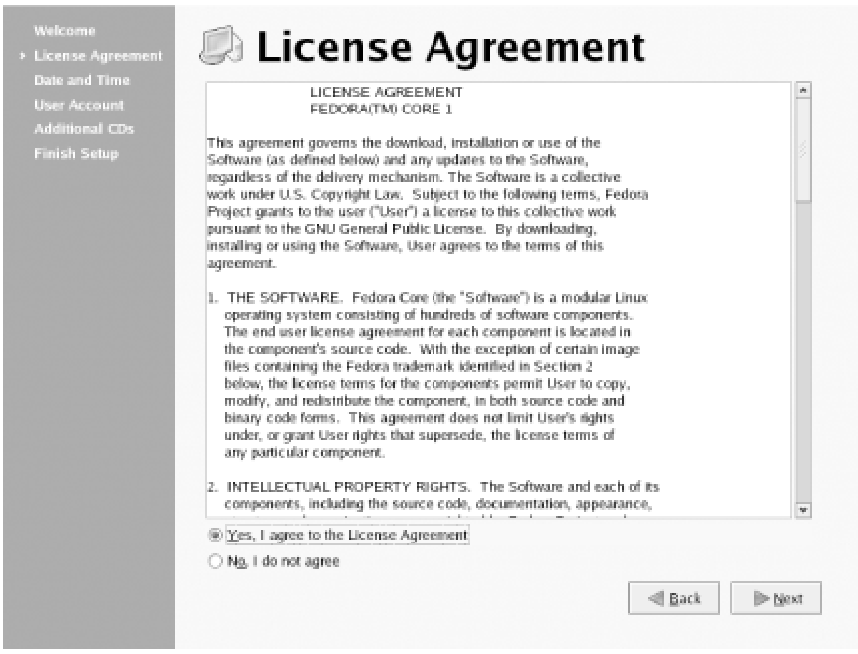
3.3 The Firstboot Service

When you boot your system for the first time, the Firstboot service is launched to help you configure your system. Figure 3-30 shows the Welcome screen the service displays. Click Next to continue.

Figure 3-30. The Firstboot service Welcome screen

The License Agreement screen (Figure 3-31) requires you to consent to Red Hat's or the Fedora team's license agreement, according to the operating system you chose to install; otherwise you'll be unable to complete the configuration of your system. To consent to the agreement, click the radiobutton labeled "Yes, I agree to the License Agreement" and then click Next.

Figure 3-31. The License Agreement screen



The Date and Time screen (Figure 3-32) lets you specify the current date and time. You can also specify a time server to which your system can synchronize, so that your system's time is highly accurate. If necessary, revise the date and time. If you want to synchronize your system's time, enable the checkbox labeled Enable Network Time Protocol and select a time server from the list labeled Server. Click Next to continue.

Figure 3-32. The Date and Time screen

3.4 Logging into the Desktop

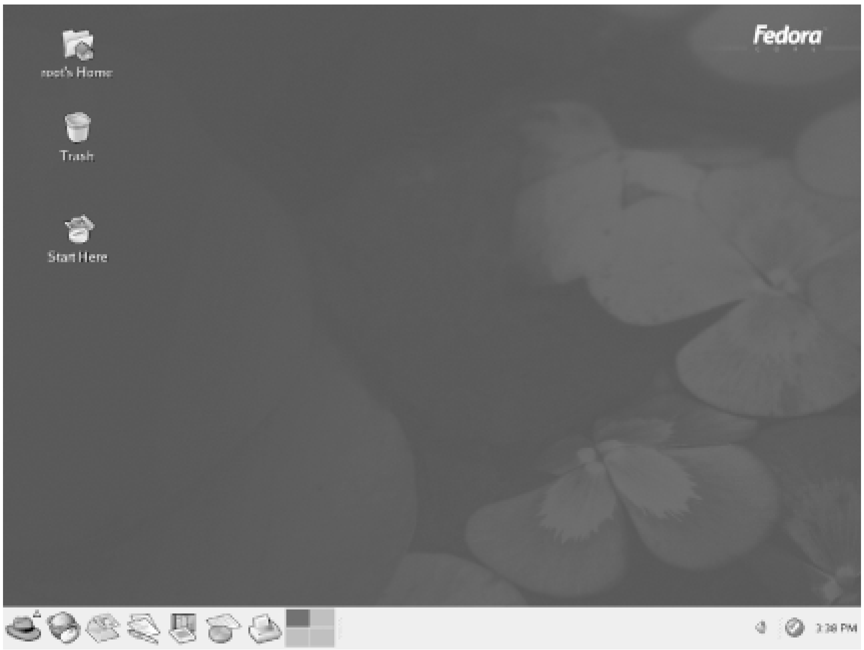
The login screen should resemble the one shown in Figure 3-37. To log in, type **root**, or the name of another user account you created, in the text box labeled Login and press **Enter**. A second login screen appears, requesting your password. Type the password you earlier assigned to the user and press **Enter**.

> If you chose to install the KDE desktop rather than the GNOME desktop, the appearance of your login screen will differ somewhat from that shown in Figure 3-37. However, the GNOME and KDE login screens are functionally similar.

Figure 3-37. The GNOME login screen

The GNOME desktop (Figure 3-38) appears. If you like, click around the desktop and see what you can discover. Chapter 5 explains how to use the GNOME desktop. However, you should read Chapter 4 before reading that chapter.

Figure 3-38. The GNOME desktop



To shut down your system, click the red hat at the lower left of the desktop. Then click Shutdown and click OK. Wait for your system to power down; now you're ready to move on to Chapter 4 to begin learning how Linux works.

> You may see a flashing red ball at the lower right of the GNOME desktop. The ball is associated with the Red Hat Network Alert Notification facility, which you'll learn about in Chapter 8.

3.5 Getting Help

If your system fails to boot or if you're unable to complete the Linux installation process, don't despair: this section will help you troubleshoot your installation.

3.5.1 Failed Graphical Startup or Login

Sometimes, the graphical startup or login fails. This is more likely if your system wasn't manufactured recently or if it's a laptop rather than a desktop.

If you don't see either display manager's screen, it's likely that your X configuration isn't appropriate for your system's video hardware. It could be that your X configuration requires revision. In the worst case, your system's video hardware may not be compatible with X; in that event, you'll nevertheless be able to run nongraphical Linux applications.

> If you see a scrambled image rather than text or images on your monitor, immediately switch off the monitor. If your monitor is an older model, it can be damaged by the incorrect configuration. Using the resources described in the following section, seek help in configuring X to work with your system's video adapter and monitor.

3.5.2 Additional Resources

As much as I'd like to help you solve your problems in installing Linux, I get too much email to be able to respond personally. But don't fret: the help you need is probably close by, in one of these sources (roughly ordered by importance and value):

The Red Hat Enterprise Linux Installation Guide

: http://www.redhat.com/docs

This guide is distributed with boxed copies of Red Hat Enterprise Linux. It's also available online. It provides a step-by-step guide to installing Red Hat Enterprise Linux that includes a few details not given in this chapter. Unfortunately, no similar guide has yet been developed for Fedora Core 1.

RedHat Enterprise Linux General Advisories

: http://www.redhat.com/apps/support/errata

The Red Hat Enterprise Linux General Advisories web page describes problems with, and enhancements to, Red Hat Enterprise Linux and provides links to updated packages. Sometimes, the installation media themselves are found to contain bugs or security problems. In that case, you may be able to find and download fixes from this web page. Chapter 8 explains Red Hat Network, which enables you to configure your system to automatically notify you of available software updates.

Fedora Updates

: http://fedora.redhat.com/download/updates.html

The Fedora project has not yet established a web page for publishing advisories. However, their updates web page will likely link to such information in the future.

Red Hat Global Support Services

: https://www.redhat.com/apps/support

This web page provides hyperlinks to a variety of information sources and resources for Red Hat Enterprise Linux users. Because Red Hat does not support Fedora Core, this resource applies only to Red Hat Enterprise Linux.

Red Hat Linux Product Activation Page

Chapter 4. How Linux Works

Before you can effectively use a desktop environment, you need to know some Linux fundamentals. This chapter explains basic Linux concepts that underlie graphical and nongraphical system use. It describes Linux user accounts and how Linux organizes data as filesystems, directories, and files. This chapter also explains how to use the X Window System (often known simply as X). Because both GNOME and KDE are built on top of X, an understanding of X is central to using either desktop environment. Even though you're probably eager to get working with your new system, I suggest you at least skim this chapter. I also predict that you'll come back to it when you have some more experience and run into something confusing.

4.1 User Accounts

Like other multiuser operating systems, such as Windows NT/2000/XP, Linux uses user accounts to identify users and allocate permissions. Every Linux system has a special user known as the *root user*. The root user is analogous to the Windows user known as Administrator. The root user can perform privileged operations that are forbidden to other users. For instance, only the root user can perform most system administration operations. By default, the username associated with the root user is **root**.

You should be judicious in your use of the root account. For instance, you should safeguard the associated password so that no one uses it to compromise your system. Also, you should log in as the root user only when performing privileged operations; that is, special administrative operations that can be performed only by the root user. Following this advice will help you avoid disasters such as accidentally deleting important files that are protected against access by ordinary, non-root users.

4.2 How Linux Organizes Data

In order to make the most effective use of your Linux system, you must understand how Linux organizes data. If you're familiar with Windows or another operating system, you'll find it easy to learn how Linux organizes data, because most operating systems organize their data in similar ways. This section explains how Linux organizes data and introduces you to several important Linux commands that work with directories and files.

4.2.1 Devices

Linux receives data from, sends data to, and stores data on *devices*. A device generally corresponds to a hardware unit, such as a keyboard or serial port. However, a device may have no hardware counterpart: the kernel creates several *pseudodevices* that you can access as devices but that have no physical existence. Moreover, a single hardware unit may correspond to several devices. For example, Linux defines each partition of a disk drive as a distinct device. Table 4-1 describes some typical Linux devices; not every system provides all these devices and some systems provide devices not shown in the table. The device name often appears in messages and filenames on the system.

Table 4-1. Typical Linux devices

Device	Description
`audio`	Sound card
`cdrom`	CD-ROM drive
`console`	Current virtual console
`fd`*n*	Floppy drive (*n* designates the drive; for example, fd0 is the first floppy drive)
`ftape`	Streaming tape drive, not supporting rewind
`hd`*xn*	Non-SCSI hard drive (*x* designates the drive and *n* designates the partition; for example, hda1 is the first partition of the first non-SCSI hard drive)
`lp`*n*	Parallel port (*n* designates the device number; for example, lp0 is the first parallel port)
`modem`	Modem
`mouse`	Mouse
`nrft`*n*	Streaming tape drive, supporting rewind (*n* designates the device number; for example, nrft0 is the first streaming tape drive)
`nst`*n*	Streaming SCSI tape drive, not supporting rewind (*n* designates the device number; for example, nst0 is the first streaming SCSI tape drive)
`null`	Pseudodevice that accepts unlimited output and throws it away
`printer`	Printer
`psaux`	PS/2 mouse
[illegible]`ft`*n*	Streaming tape drive, not supporting rewind (*n* designates the device number;

4.3 Using X

X is the standard graphical user interface (GUI) for Linux. Like other GUIs, such as Windows and Mac OS, X lets you interact with programs by using a mouse (or other pointing device) to point and click, providing a simple means of communicating with your computer.

Despite its age, X is a remarkable and very modern software system offering a cross-platform, network-oriented GUI. It runs on a wide variety of platforms including essentially every flavor of Unix, such as Solaris, Linux, and the BSDs (FreeBSD, NetBSD, and OpenBSD). X clients are available for use, for example, under Windows 3.*x*, 9*x*, NT, 2000, and XP. The sophisticated networking capabilities of X let you run a program on one computer while viewing the graphical output on another computer via a network connection. X was designed to provide room for experimentation with new interfaces, so a wide variety of X-based window managers and desktops is available for every taste. On the other hand, this variety can provoke some minor confusion, as interfaces and behaviors vary slightly from one system to another.

Most Linux users run XFree86, a freely available software system compatible with X, which is distributed with Red Hat Linux. XFree86 was developed by the XFree86 software team, which began work in 1992. In 1994, the XFree86 Project, Inc. (http://www.xfree86.org) assumed responsibility for ongoing research and development of XFree86.

4.3.1 Window Managers

Using X means interacting with Linux on several different levels. X itself merely provides the graphics facility for displaying components of a GUI: X draws the screen, draws objects on the screen, and tracks user input actions such as keyboard input and mouse operations. To organize the desktop into familiar objects like windows, menus, and scrollbars, X relies on a separate program called a window manager. But even more functionality is required. A window manager alone doesn't provide tight integration between applications of the sort required by drag-and-drop operations; that higher degree of integration comes from what's called a desktop environment. While X itself is a single program, X under Linux supports several popular window managers and two popular desktop environments, GNOME and KDE.

Window managers create the borders, icons, and menus that provide a simple-to-use interface. Window managers also control the look and feel of X, letting you configure X to operate almost any way you desire. At one time, it was common for Linux users to separately choose a window manager and desktop environment. However, today most users retain the window manager with which their preferred desktop environment is initially configured. Under Red Hat Linux Enterprise 3, GNOME uses the Metacity window manager and KDE uses Kwin, formerly known as the K Window Manager, or simply K. Because of the variety of window managers, scrollbars and other widgets may behave differently from one system to another. But, the differences are minor and determined clicking generally discovers the proper method of interacting with a widget. See Section 4.3.5.2 later in this chapter.

4.3.2 Desktop Environments

A *desktop environment* is a set of desktop tools and applications. The Windows desktop includes applications such as the Windows Explorer, accessories such as Notepad, games such as FreeCell and Minesweeper, and utilities such as the Control Panel and its applets. Although you can run X without a desktop, having a desktop helps you work more efficiently. Both GNOME and KDE are free software and are developed by teams in an open, collaborative manner.

The default Red Hat Linux desktop environment is GNOME. However, you can easily reconfigure KDE as the desktop, if you prefer. The choice between GNOME and KDE is now not so important as in the past. Red Hat has reworked GNOME and KDE to give them a consistent look and feel. Moreover, almost every GNOME application can now be run under KDE and almost every KDE application can now be run under GNOME. So your Linux experience will be similar whether you're using GNOME or KDE.

4.3.2.1 GNOME

GNOME stands for the GNU Network Object Model Environment (pronounced as *guh-nome* or *gee-nome*). One of GNOME's most interesting features is session awareness. When you

Chapter 5. Using the GNOME and KDE Desktops

Red Hat Enterprise Linux and Fedora Core support two desktops, GNOME and KDE. This choice is consistent with the Linux philosophy of having it your own way. But the reasons behind having multiple desktops have more to do with history and law than technology.

At one time, parts of KDE were distributed under a license that some believed required commercial users to pay a license fee. Because Red Hat wanted Red Hat Linux to be freely redistributable and usable, Red Hat included only GNOME in the Red Hat Linux distribution. Red Hat also assisted in the development of the GNOME desktop. However, the KDE license was eventually clarified. At that point, Red Hat warmed toward KDE and included it in the Red Hat Linux distribution. Many users prefer KDE to GNOME, finding it in many ways more mature than GNOME. But GNOME retains a somewhat favored status in Red Hat's eyes, as indicated by the installation program's default choice of GNOME as the desktop. If you want to install KDE, you must manually select the KDE package group.

To minimize the confusion that might otherwise result from dueling desktops, Red Hat has worked to give GNOME and KDE a more consistent look and feel. This has upset many KDE fans, who prefer KDE's native look and feel to that imposed by Red Hat. An advantage of Red Hat's decision is that most applications work properly under both GNOME and KDE. However, a disadvantage of Red Hat's decision is that KDE now includes applications that lack the distinctive look and feel that unites the KDE desktop.

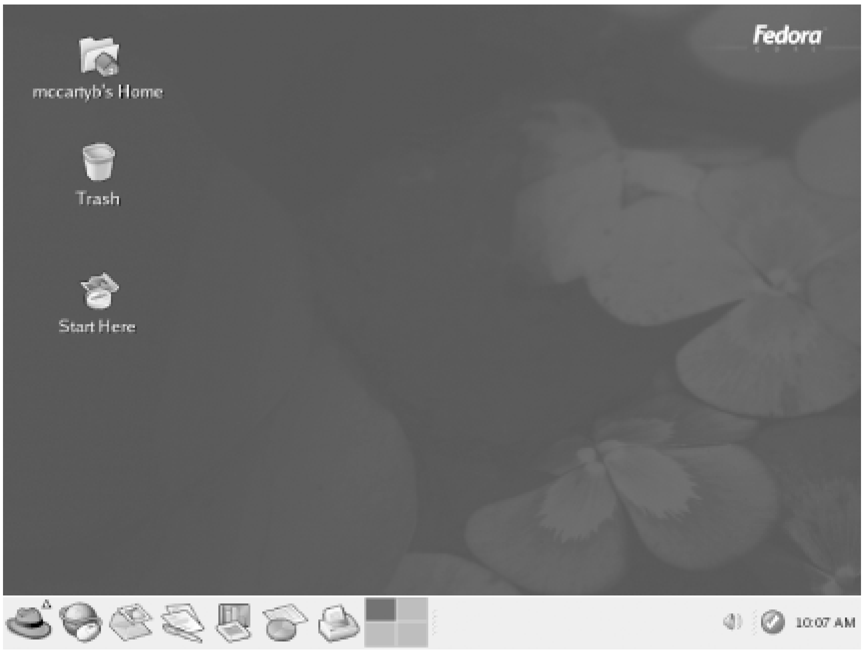
I'm not much interested in justifying or attacking Red Hat's decision. The decision has been made, and the goal of this book is to describe Red Hat Enterprise Linux as it is, rather than as it might be. Therefore, this chapter describes both desktops, devoting roughly equal space to each. My personal recommendation is that you try each desktop for a while and use the one you prefer.

Some readers of earlier editions of this book complain that I favor one desktop over the other. Yes, I do have a personal favorite. However, almost all such readers have incorrectly identified my preference. And, my preference has recently changed. So, I believe that my presentation of the desktops is reasonably fair and unbiased.

5.1 Using the GNOME Desktop

When you first log in to your Red Hat Enterprise Linux or Fedora Core system, you will see the GNOME desktop (Figure 5-1). The contents of your desktop may be slightly different, of course.

Figure 5-1. The GNOME desktop

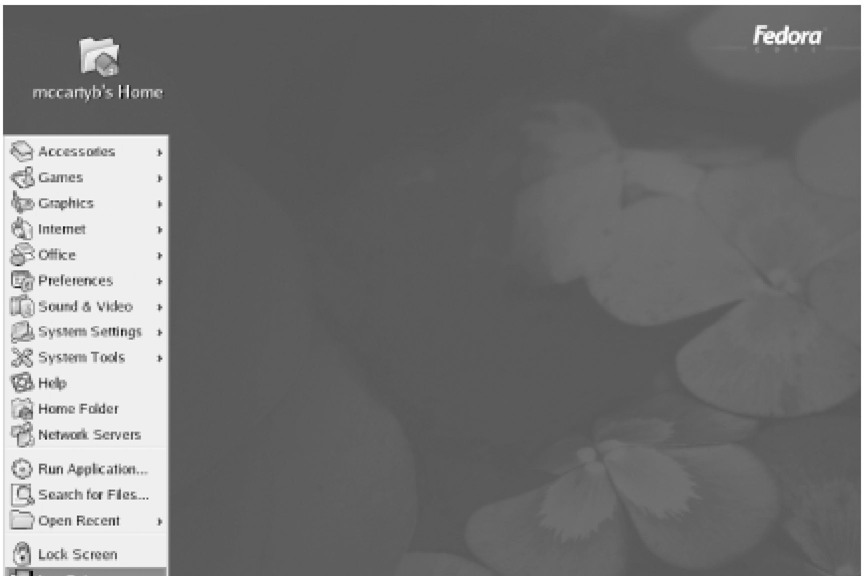


> If you want to launch a GNOME session, but KDE is configured as the default desktop environment, select Gnome from the Session menu of the system login screen. Of course, GNOME must be installed in order for this to work.

To log out of GNOME, left-click on the main menu, which resembles a red hat. From the pop-up menu that appears, select the Log Out menu item (Figure 5-2). A Log Out dialog box (Figure 5-3) appears and asks you to confirm your decision to log out. Clicking OK terminates your GNOME session. If you enable the checkbox titled Save Current Setup, the GNOME session manager will save the state of your desktop and restore it when you log in again to GNOME.

There are two other options in the Log Out dialog box. Select the Shut Down button to shut down your system, or the Restart the computer button to restart it.

Figure 5-2. Logging out of GNOME



5.2 Using the KDE Desktop

As explained at the beginning of this chapter, Red Hat Linux initially included only GNOME; however, Red Hat Enterprise Linux and Fedora Core support both GNOME and KDE. Figure 5-13 shows KDE's desktop. If your system is configured to use GNOME and you want to launch a KDE session, select KDE from the Session menu of the system login screen. Of course, KDE must be installed in order for this to work.

Figure 5-13. The KDE desktop

5.2.1 The KDE Desktop

KDE has a main menu icon in its panel, at the lower left of the screen. The icon is identical to that associated with GNOME's main menu, a red hat. Clicking the icon reveals a menu that includes a Logout menu item. You can use the Logout menu item to terminate KDE.

Right-clicking the KDE desktop causes a pop-up menu to appear. From this menu, you can create desktop shortcuts and perform a variety of other functions. The desktop includes a variety of icons and folders. The specific icons and folders that appear may vary depending on the software installed on your system and your KDE configuration. The most common icons are described in the following subsections.

5.2.1.1 Start Here icon

By double-clicking the Start Here icon, you can launch Konqueror, KDE's file manager, to view a folder that contains several useful icons. Double-clicking any icon in the folder launches a window containing icons that provide access to KDE facilities. You can access the same facilities by using the KDE menu. The icons within the Start Here folder include:

Applications

The Applications icon lets you launch various applications.

Preferences

The Preferences icon provides access to a folder containing icons that enable you to view and modify a variety of preferences, including those for the desktop, document handlers, user interface look and feel, multimedia, and peripherals.

System Settings

The System Settings icon provides access to tools for viewing and modifying the system configuration.

5.2.1.2 Home Directory icon

The Home Directory icon enables you to view your home directory by using KDE's file

Chapter 6. Using Linux Applications

Red Hat Enterprise Linux and Fedora Core include a plethora of applications. This chapter introduces you to several of the most popular and useful applications. These include: OpenOffice.org, a desktop suite; Evolution, an email client and personal scheduler; and Kpilot, an application for syncing a personal digital assistant (PDA) with your system. This chapter also explains how to use the Nautilus file manager to burn CDs.

PREV < Day Day Up > NEXT

6.1 OpenOffice.org

OpenOffice.org is a desktop suite that functionally resembles Microsoft Office. That is, OpenOffice.org can perform many of the functions performed by Microsoft Office and includes many of the familiar features of Microsoft Office, along with a few features not found in Microsoft Office. The distinctive advantage of a desktop suite is that its component applications are designed to work together. The applications of a desktop suite have a similar look and feel, which makes them easy to learn and use.

Linux users have long had access to applications that help them prepare documents. However, development of Linux desktop suites has lagged behind that of Microsoft Office. The applications and suites have tended to be somewhat clumsy to use, unreliable, and poor in features. OpenOffice.org sets a new standard for Linux desktop suites, providing features and capabilities that are adequate to satisfy most computer users, not merely Linux fans.

OpenOffice.org began as a commercial desktop suite known as StarOffice, created by StarDivision. When Sun Microsystems acquired StarDivision in 1999, Sun soon thereafter released a freely available version of StarOffice. More recently, Sun has made certain StarOffice technologies available to the open source community, which created the freely redistributable OpenOffice.org desktop suite. Sun plans to continue development of StarOffice, which is a component of Sun's Java Desktop System, as a commercial product. At the same time, the open source community plans to continue development of OpenOffice.org.

OpenOffice.org is a multi-platform product, and is currently available for Linux, PPC Linux, Solaris, Windows, and Mac OS X (under Apple's X11). Work is underway to support other platforms, including FreeBSD, OpenVMS, and IRIX. OpenOffice.org is also a global product, currently supporting 32 languages. Support for new languages is added regularly.

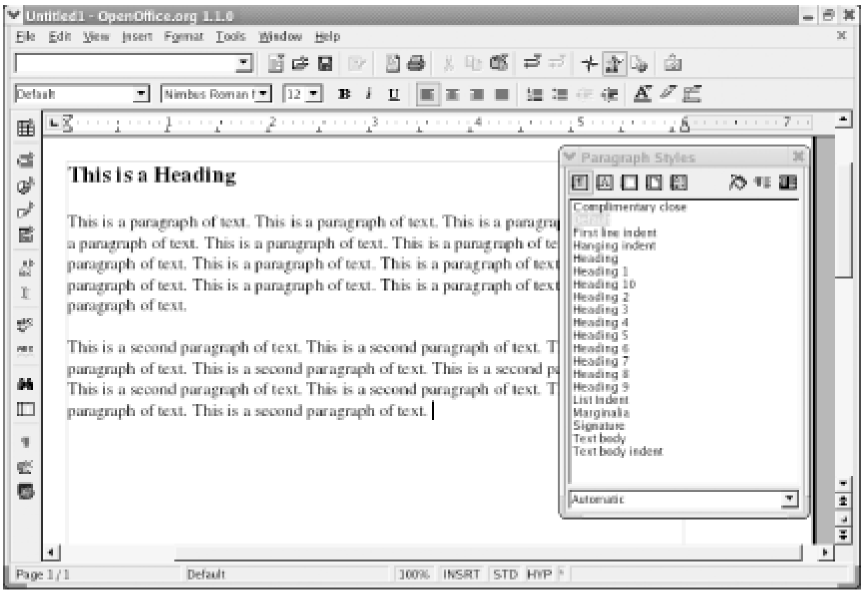
OpenOffice.org includes translation filters that let you share documents with users of Microsoft Office and other popular applications. It also includes convenient features such as Print to PDF (Adobe Portable Document Format); AutoPilot, which assists you in creating complex documents; and Stylist, which helps you take control of the look of your document.

OpenOffice.org includes word processor, spreadsheet, graphics, presentation manager, and drawing applications. The next several sections describe these applications. You can learn more about OpenOffice.org at http://www.openoffice.org.

6.1.1 Writer: The OpenOffice.org Word Processor

Writer is available via the Office menu item of the GNOME and KDE menus and, still more conveniently, has its own panel icon, which resembles a pen superimposed on two sheets of paper. When you launch Writer, you'll see a window resembling that shown in Figure 6-1.

Figure 6-1. Writer: the OpenOfice.org word processor



If you're launching Writer for the first time, a dialog box invites you to register as an OpenOffice.org user. Your registration is invited?and appreciated by the developers of

6.2 Evolution

Ximian's Evolution is an email client and personal scheduler. Previously, Evolution was available from Ximian as an add-on to Red Hat Linux. However, conflicts between Ximian's RPM packages and those distributed by Red Hat sometimes made life complicated for Ximian users. Evolution is now part of Red Hat Enterprise Linux and Fedora Core, so Ximian users can expect more trouble-free operation and fewer problems when upgrading to new Linux versions. You can learn more about Evolution at http://www.ximian.com.

Evolution has four main functions:

Email client

Receives email from POP and IMAP servers and sends email via SMTP servers.

Calendar

Provides daily and monthly calendars to help you plan your time.

Task list

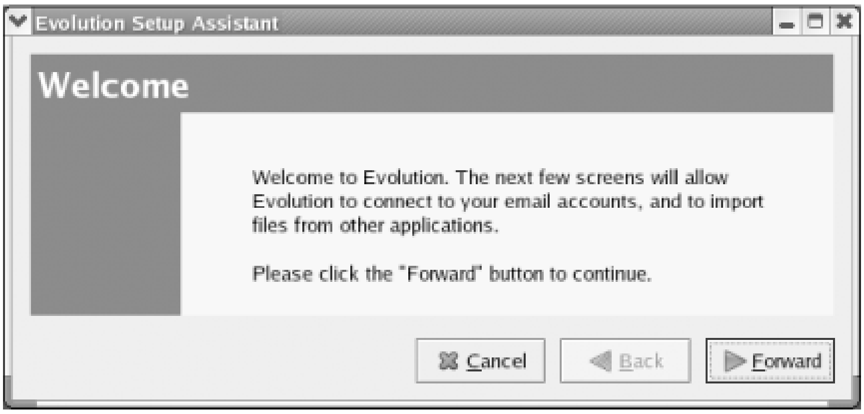
Provides a to-do list that helps you keep track of projects and deadlines.

Contact database

Provides a contact list that you can conveniently use when composing email.

Evolution is available via the Internet ⟶ Email submenu of the GNOME and KDE menus. It also has a convenient panel icon, which resembles a postage stamp superimposed on an envelope. When launched for the first time, Evolution provides a wizard to help you configure its operation, as shown in Figure 6-7.

Figure 6-7. The Welcome panel of the Evolution setup wizard



Click Forward to move on to the Identity panel. There, you specify your full name and email address. Optionally, you can specify the name of your organization, a plain text email signature file, and an HTML email signature file.

Click Forward to move on to the Receiving Email panel. There, you specify the type of server you use to receive email, the hostname of the server, your username, and information on the security and authentication characteristics of the server. You can obtain this information from your email provider or determine the correct values by experimentation.

Click Forward to move on to a second Receiving Email panel. This panel lets you specify options about checking for new email and using folders and filters. You can specify the values according to your preferences.

Click Forward to move on to the Sending Email panel. There, you specify the type of server you use to send mail, its hostname, and security and authentication options. You can obtain this information from your email provider or determine the correct values by experimentation.

Clicking Forward takes you to the Account Management panel. This panel lets you associate a name with the account, so that you can distinguish from accounts you may add later. You

6.3 Pilot/Handspring Tool

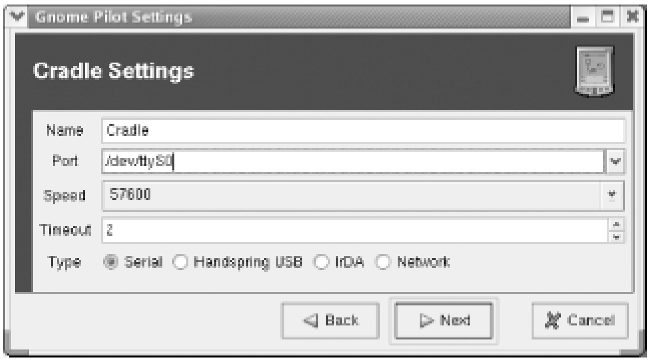
Red Hat Enterprise Linux includes gnome-pilot, a tool that lets you synchronize your Palm or Handspring PDA (personal digital assistant) with your Linux system using your system's serial or USB port and your PDA's hot sync cradle. By default, Red Hat Linux is configured to enable you to back up your PDA to your Linux system or restore a backup from your Linux system to your PDA. You can also synchronize Evolution's calendar, task list, and contact list with your PDA.

To set up your system to communicate with your PDA, choose Tools ⟶ Pilot Settings from the Evolution menu. The Welcome panel appears, as shown in Figure 6-9.

Figure 6-9. The gnome-pilot Welcome panel

Plug your PDA into its cradle and plug the cradle into your system's serial or USB port, according to the type of cradle you're using. Click Forward to continue. The Cradle Settings panel (Figure 6-10) appears. Specify the port to which the cradle is attached.

Figure 6-10. The Cradle Settings panel



If you've synched your PDA using a Microsoft Windows host, you can use the information in Table 6-1 to determine the serial port. If you're using a USB port, specify a port having the form */dev/ttyUSBn*, where n is the number of the USB port.

Table 6-1. Linux and Windows serial port designations

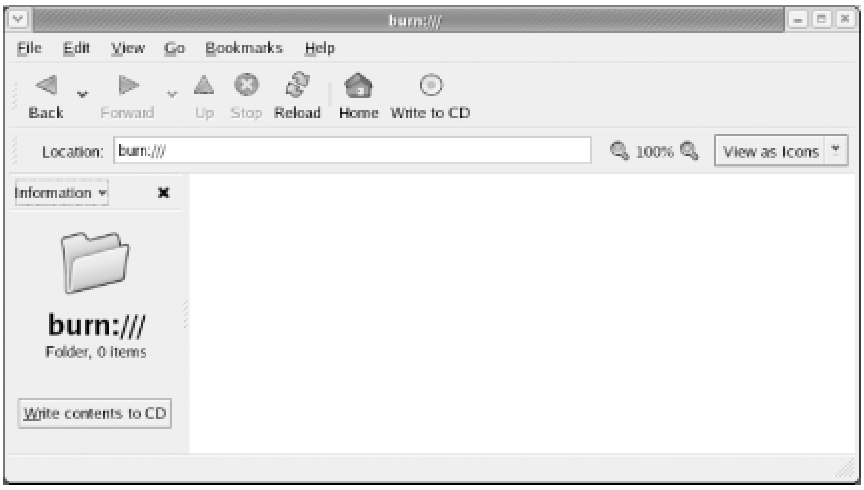
Windows designation	Linux designation
COM1	/dev/ttyS0
COM2	/dev/ttyS1
COM3	/dev/ttyS2

6.4 CD Creator

At one time, writing a CD under Linux was a common rite of passage for new Linux users. The command used to write a CD, **cdrecord**, is a sophisticated command with many options. Getting all the options just right was a challenge for many.

However, Creator, a program built into the Nautilus file manager, makes it easy to write CDs. To launch CD Creator, simply insert a blank CD into your CD-R or CD-RW drive. Alernatively, you can launch Nautilus and select Go ⟶ CD Creator from the Nautilus menu. Nautilus presents the burn window shown in Figure 6-14.

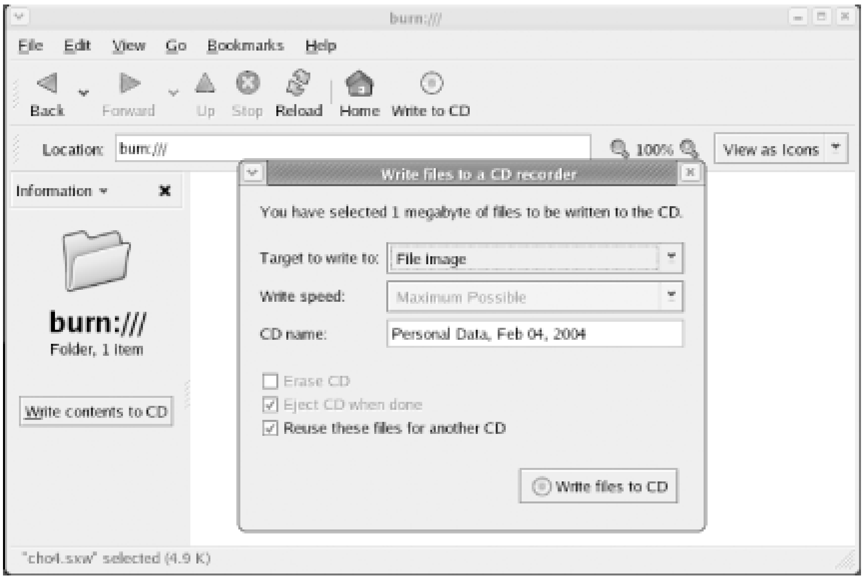
Figure 6-14. The burn window



To specify the files to be written to the CD, open a second Nautilus window. Then simply drag the desired files or directories into the CD Creator window.

When you've specified all the files to be written, click the Write to CD icon on CD Creator's toolbar. The Write Files to a CD Recorder dialog box appears (Figure 6-15).

Figure 6-15. The Write Files to a CD Recorder dialog box



Specify the target device, write speed, and other options. You can create an ISO image file?a disk file that contains the same data that would be found on a CD-ROM disk?by writing to the special File image target. However, in the more likely case that you want to burn a physical CD, you should choose your CD-R drive as the target device. Click the Write files to CD button to begin burning the CD.

Chapter 7. Conquering the bash Shell

Linux provides two user interfaces: the graphical user interface (GUI) hosted by X and an older, command-line interface (CLI) called the shell.

Those familiar with the MS-DOS command-line interface will recognize the shell, which you use by typing text commands to which the system responds by displaying text replies. But the comparison with the MS-DOS command line doesn't do justice to the Linux shell, which is vastly more powerful. And, *older* doesn't necessarily imply *inferior*.

GUIs are stylish primarily because they're easy to learn and use. But they're not always the most efficient way of operating a computer. A skilled user of the shell can often outrace a competitor using a GUI. Moreover, a GUI enables its user to perform only the functions provided by the GUI's programmers. In contrast, the shell is expandable. The shell enables users to define entirely new operations based on sequences of existing operations.

The real power of Linux lies in the shell. So, if you aspire to master Linux, you must conquer the shell. Even if your ambition falls short of gurudom, you'll find knowledge of the shell helpful. Many procedures from sources other than this book assume that you know how to use the shell. And, if X fails, you can't easily repair it without knowing how to use the shell.

Linux supports a variety of shells, but the most popular is the *bash* shell, described in this chapter. The chapter explains how to issue shell commands, and how to use shell commands to manipulate files and directories, work with removable media, and launch programs. The chapter also explains nano, a simple text editor that operates in text mode.

7.1 Issuing Shell Commands

The most common way to access the shell is via a terminal window, as explained in Chapter 5 and Chapter 6. However, a terminal window isn't the only way to access the shell. The Section 4.3.3.2 in Chapter 4 explains how to access the shell with a virtual console.

As your first Linux command, launch a terminal window, type **w**, and press **Enter**. Your contents of the terminal window should look something like this:

```
[bill@home bill]$ w
11:12am  up 6 min,  1 user,  load average: 0.00, 0.08, 0.05
USER     TTY      FROM       LOGIN@   IDLE   JCPU   PCPU  WHAT
bill     tty1               11:11am  0.00s  0.20s  0.11s  -bash
```

The **w** command tells Linux to display the system status and a list of all system users. In the example, the output of the command tells you that it's now 11:12 a.m., that the system has been up for six minutes, and that only one user?*bill*?is currently logged in. Notice that the command output is very terse, packing much information into a few lines. Such output is typical of Linux commands. At first, you may find Linux output cryptic and difficult to read, but over time you'll grow to appreciate the efficiency with which Linux communicates information.

> Linux command output is not terse owing to an oversight or laziness on the part of the creators of Linux. Instead, Linux command output is designed so that it can be processed by programs as well as by humans. The structure of the output simplifies the task of programmers who write programs to process command output.

Linux provides many commands besides the **w** command?so many that you may despair of learning and recalling them. Actually, the number of commands you'll use regularly is fairly small. Soon, they will become second nature to you.

Try a second command, the **date** command:

```
[bill@home bill]$ date
Fri Oct 5 11:15:20 PST 2004
```

The **date** command displays the current date and time.

If you find working with MS-DOS distasteful or intimidating, you may not immediately enjoy working with the Linux command line. However, give yourself some time to adjust. The Linux command line has several features that make it easier to use, and more powerful, than MS-DOS.

7.1.1 Correcting Commands

Sometimes you may type a command incorrectly, causing Linux to display an error message. For example, suppose you typed **dat** instead of **date**:

```
[bill@home bill]$ dat
bash: dat: command not found
```

In such a case, carefully check the spelling of the command and try again. If you notice an error before pressing **Enter**, you can use the **Backspace** or **Left** arrow key to return to the point of the error and then type the correct characters. The **Backspace** key erases characters whereas the **Left** arrow key does not. You can also use the **Delete** key to delete unwanted characters.

Just as a web browser keeps track of recently visited sites, the *bash* shell keeps track of recently issued commands in what's known as the *history list*. You can scroll back through *bash*'s history by using the Up arrow key, or back down using the Down arrow key, just as you would with the Back and Forward buttons on a web browser. To reissue a command, scroll to it and press **Enter**. If you like, you can modify the command before reissuing it. When typing shell commands, you have access to a minieditor that resembles the DOSKEY editor of MS-DOS. This minieditor lets you revise command lines by typing key commands. Table 7-1 summarizes some useful key commands interpreted by the shell. The key commands

7.2 Working with the Linux Command Prompt

Linux commands share a simple, common structure. This section describes their common structure and explains how you can obtain helpful information about the commands available to you.

7.2.1 Commands and Arguments

The general form of a shell command line is this:

```
command [options] [arguments]
```

The **command** determines what operation the shell will perform and the *options* and *arguments* customize, or fine-tune, the operation. The *options* and *arguments* may or may not appear, as indicated by the square brackets. Sometimes the **command** specifies a program file that will be launched and run; such a command is called an *external command*. Linux generally stores these files in */bin*, */usr/bin*, or */usr/local/bin*. System administration commands are generally stored in */sbin* or */usr/sbin*, which are included by default in the path of the root user. When a command specifies a program file, the shell passes any specified arguments to the program, which scans and interprets them, adjusting its operation accordingly.

Some commands are not external program files; instead they are built-in commands interpreted by the shell itself. One important way in which shells differ is in the built-in commands that they support. Later in this section, you'll learn about some of *bash*'s built-in commands.

The name of a Linux command almost always consists of lowercase letters and digits. Most commands let you specify options or arguments. However, in any given case, you may not need to do so. For example, typing the **w** command without options and arguments causes Linux to display a list of current users.

> Remember, Linux commands are case sensitive; be sure to type each character of a command in the proper case.

Options modify the way that a command works. Many options consist of a single letter, prefixed by a dash. Often, you can specify more than one option; when you do so, you separate each option with one or more spaces. For example, the **-h** option of the **w** command causes the output of the command to omit the header lines that give the time and the names of the fields:

```
[bill@home bill]$ w -h
```

Arguments specify filenames or other targets that direct the action of the command. For example, the **w** command lets you specify a username as an argument, which causes the command to list logins that pertain only to the specified user:

```
[bill@home bill]$ w bill
```

Some commands let you specify a series of arguments; you must separate each argument with a space between them. For example, the following command prints a list of logins by the root user, without header lines:

```
[bill@home bill]$ w -h bill
```

When a command includes several arguments, a command may not fit on a single line. However, you can continue typing when you reach the end of a line, because the shell will automatically wrap your input to the next line. If you find line wrapping disconcerting, you can type a backslash (\) at the end of a line, press **Enter**, and continue typing on the next line. The backslash is the shell's line continuation character; the shell sees lines joined by a backslash as though they were a single line. Don't type anything after the backslash or the continuation feature won't work correctly.

7.2.2 Getting Help

Because Linux provides so many commands and because Linux commands provide so many possible options, you can't expect to recall all of them. To help you, Linux provides the **man**

7.3 Useful Linux Programs

This section presents several programs you may find helpful in working with your Linux system. You'll learn several commands that report system status and you'll learn how to use *nano*, a simple text editor.

7.3.1 Viewing System Information

Linux provides a number of commands that report system status. The most commonly used commands are shown in Table 7-8. These commands can help you troubleshoot system problems and identify resource bottlenecks. Although each command can be used without options or arguments, each supports options and arguments that let you customize operation and output; consult the appropriate manpage for details.

Table 7-8. Useful system commands

Command	Function
df	Shows the amount of free disk space (in 1 KB blocks) on each mounted filesystem.
du	Shows the amount of disk space (in 1 KB blocks) used by the working directory and its subdirectories. With the **-s** option, displays just a summary without listing all the subdirectories and files.
free	Shows memory usage statistics, including total free memory, memory used, physical memory, swap memory, shared memory, and buffers used by the kernel.
ps	Shows the active processes (instances of running programs) associated with this login session. Use the **-a** option to list all processes.
top	Shows a continually updated display of active processes, and the resources they are using. Type the **q** key to exit.
uptime	Shows the current time, the amount of time logged in, the number of users logged in, and system load averages.
users	Shows each login session.
w	Shows a summary of system usage, currently logged-in users, and active processes.
who	Shows the names of users currently logged in, the terminal each is using, the time each has been logged in, and the name of the host from which each logged in (if any).

7.3.2 Using the nano Editor

If you're working under X, you have access to a variety of GUI text editors. However, GUI text editors cannot be used from a virtual console. The *nano* editor is a simple text editor that you can think of as the Linux equivalent of the Windows program named Edit, because it can be used in graphical or text mode.

To start *nano*, simply type **nano** at the shell prompt, or if you want to create or edit a particular file, type **nano** followed by the name of the file (or the file's path, if the file is not in the working directory). For example, to create or edit the file *mydata*, type:

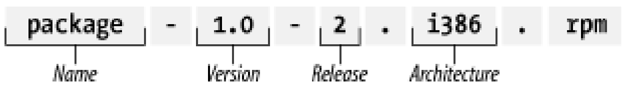
```
[bill@home bill]$ nano mydata
```

Chapter 8. Installing Software Using the RPM Package Manager

This chapter explains the Red Hat Package Manager (RPM), a tool that facilitates installing, uninstalling, and upgrading software for your Red Hat Linux system. The chapter explains how to use RPM to find the package associated with an application and how to quickly and easily install the package. It also explains how to use RPM to upgrade packages and query the status of installed packages.

An RPM package (or more simply, an *RPM* or a *package*) is a file that contains executable programs, scripts, documentation, and other files needed by an application or software unit. RPM packages are generally named using a convention that lets you determine the name of the package, the version of the software, the release number of the software, and the system architecture for which the application is intended. Figure 8-1 shows how the components of a package name are arranged.

Figure 8-1. The structure of a package name

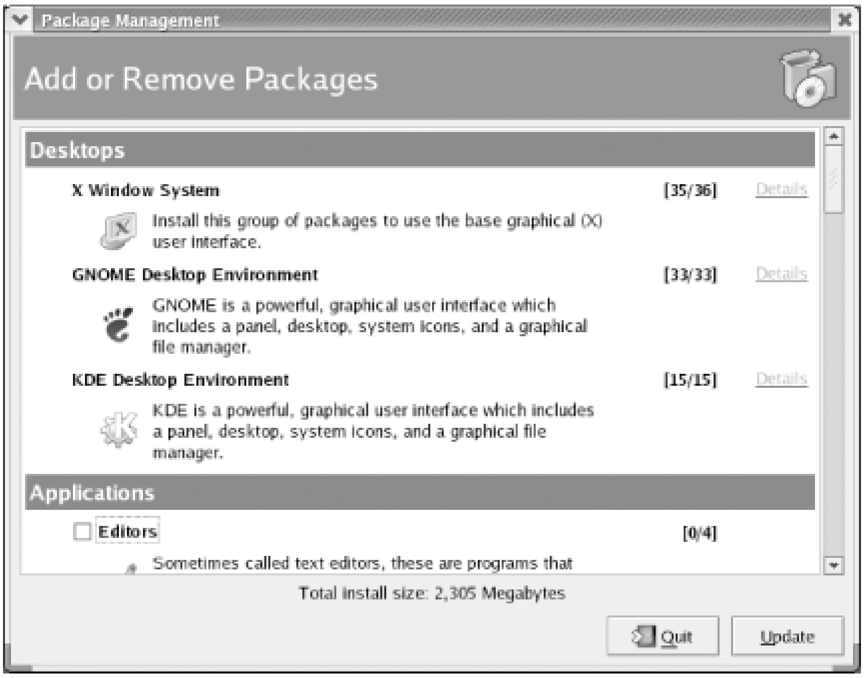


8.1 The Package Management Tool

Red Hat Enterprise Linux and Fedora Core feature a package management tool that's much easier to use than GnoRPM, the tool included in previous versions of Red Hat Linux. To launch the package management tool, choose System Settings ⟶ Add/Remove Applications from the main menu. Only the root user can manipulate packages. If you're not logged in as the root user, the tool helpfully prompts you for the root user's password. The tool checks the status of installed packages, as shown in Figure 8-2. This process may require a minute or two. Then, the Add or Remove Packages window (Figure 8-3) appears.

Figure 8-2. Checking system package status

Figure 8-3. The Add or Remove Packages window



8.1.1 Installing Package Groups and Packages

To install a package package group, select the checkbox next to the name of an uninstalled package group. In Figure 8-3, the Editors package group is not installed, as indicated by the unchecked checkbox to the left of the package group's name. By enabling the checkbox, you can specify that you want to install the package group. Installation does not begin until you click the Update button.

Most package groups contain optional packages that you can install or omit, according to your preference. The numbers to the right of the package group name indicate how many of the optional packages are installed. If you've just specified that a package group should be installed, the numbers indicate how many of the optional packages will be installed when you click the Update button. For example, in Figure 8-3, the 0/4 associated with the Editors package group indicates that none of the four associated packages is currently installed. If

8.2 The redhat-install-packages Command

Many GUI tools use shell commands behind the scenes. The package management tool uses a special command, **redhat-install-packages**, to install packages. Although the package management tool can install only packages that reside on the installation media, the **redhat-install-packages** command can install RPM packages from any source. For example, you can use this command to install packages that you've downloaded from a web or FTP server.

RPM package names generally end with a *.rpm* extension. To install a package using the **redhat-install-packages** command, issue the command:

```
redhat-install-packages  package-file.rpm
```

where *package-file.rpm* is the name of the package you want to install. You can install multiple packages by listing several filenames, separating each from the next with a space, like so:

```
redhat-install-packages  package-file1.rpm package-file2.rpm
package-file3.rpm
```

If a desired package requires other packages in order to operate correctly, the other packages must already be installed or must be installed at the same time as the desired package. Otherwise, installation of the desired package will fail. See the upcoming Section 8.5.1 for more information on this potential problem. However, the **redhat-install-packages** command will attempt to automatically install necessary packages that reside on the installation media.

> Several Linux distributions publish their software as RPMs. Although the **redhat-install-packages** command is generally capable of installing foreign packages?that is, packages not created for Red Hat Enterprise Linux or Fedora Core?installing foreign packages may leave your system in an unusable state. If you're using Red Hat Enterprise Linux, you should generally install only packages created for Red Hat Enterprise Linux, Version 3. Similarly, if you're using Fedora Core, you should generally install only packages created for Fedora Core 1. You can safely use the **redhat-install-packages** command to install such packages.

8.3 The rpm Command

RPM packages are built, installed, uninstalled, and queried with the **rpm** command. Like other command-line facilities, the **rpm** command gives you almost complete control over its operation. So, you can use the **rpm** command to accomplish feats not possible using the package management tool. For instance, you can use the **rpm** command to install or remove individual packages, install packages from non-Red Hat media, and install packages other than those built by Red Hat. On the other hand, the **rpm** command is sometimes much less convenient to use than the package management tool. So, you use can either the GUI or command line, according to your needs and preferences.

rpm has several modes, each with its own options. The general format of the **rpm** command is:

rpm [*options* **] [** *packages* **]**

The first option generally specifies the **rpm** mode (e.g., install, query, update, build, etc.); any remaining options pertain to the specified mode.

The **rpm** command has built-in FTP and HTTP clients. So, you can specify an *ftp://* or *http://* URL to identify an RPM package stored on a remote host.

Unless the system administrator has specially configured the system, any user can query the RPM database. Most other RPM functions require root privileges. Strictly speaking, it's not necessary for you log in as root to install an RPM package; however, your user account must be authorized to access and modify the files and directories required by the package, including the RPM database itself. Generally, the easiest way to ensure such access is by logging in as root.

8.3.1 General rpm Options

The following **rpm** options can be used with all modes:

--ftpport *port*

Use *port* as the FTP port.

--ftpproxy *host*

Use *host* as a proxy server for all transfers. Specified if you are FTPing through a firewall system that uses a proxy.

--help

Print a long usage message (running **rpm** with no options gives a shorter usage message).

--quiet

Display only error messages.

--version

Print the version number of **rpm**.

-vv

Print debugging information.

8.4 Querying the RPM Database

You can query RPM's database, which lists the packages installed on your system. For example, to display a simple description of an installed package, use a command like this one:

```
rpm -q  package
```

In this command, *package* is the name of the package you want RPM to describe. In response, RPM prints the package name, version, and release number.

Rather than use the **-q** option and the package name, you can use any of the following alternative options:

-a

Causes RPM to display information about all installed packages

-f *file*

Causes RPM to display information about the package that owns *file*

-p *packagefile*

Causes RPM to display information about the package contained in *packagefile*

You can also tailor the output of an RPM query, by specifying one or more of the following options:

-c

Causes RPM to display a list of configuration files included in the package

-d

Causes RPM to display a list of documentation files included in the package

-i

Causes RPM to display the package name, description, release number, size, build date, installation date, vendor, and other information

-l

Causes RPM to display the list of files that the package owns

-s

Causes RPM to display the state of all the files in the package?normal, not installed, or replaced

For example, the command:

```
rpm -qid rhide
```

displays information about the *rhide* package, including a list of documentation files included in the package.

8.5 Installing a Package

To install a package, log in as root and issue the following command from a shell prompt:

```
rpm -ivh  package
```

where *package* specifies the name of the file that contains the package. You can specify multiple packages, as long as you include a space to separate each package name from its neighbor. For example, the following command installs both the *nano* and *mutt* packages from files in the current directory:

```
rpm -ivh nano-1.2.1-4.i386.rpm mutt-1.4.1-3.i386.rpm
```

The options used with the **rpm** command include:

-i

This option specifies that RPM should install the package or packages given as arguments.

-h

This option specifies that RPM should print hash marks (`#`) as it installs the package as a visible indication of progress.

-v

The verbose option specifies that RPM should print messages that summarize its actions and progress.

Generally, RPM will successfully install the specified package. However, errors can occur. RPM may report:

- That the package is already installed
- That a package file conflicts with a file from another package
- A failed dependency

The next three sections explain how to resolve these errors.

8.5.1 Package Is Already Installed

If a package has already been installed, RPM will not overwrite the package without your permission:

```
# rpm -ivh bad-1.0-1.i386.rpm
bad package bad-1.0-1 is already installed
```

If you want to overwrite the package, add the **?replacepkgs** option to your command:

```
rpm -ivh --replacepkgs bad-1.0-1.i386.rpm
```

It may be more appropriate to update the package. Updating the package leaves its configuration files intact, whereas overwriting the package replaces the configuration files with files containing default options. An upcoming section shows you how to update a package.

Of course, it's also possible that you should do nothing. You may have attempted to install the package without first checking whether it's already installed and operational. In that case, you can use RPM to verify that the package is installed correctly and update or overwrite the package only if RPM reports problems.

To verify a package against a package file, issue the following command from a shell prompt:

```
rpm -Vp  package
```

In the command, *package* specifies the name of the file that contains the package; for example, *basesystem-8.0-1.rpm*. In verifying a package, RPM compares the installed files with the original package contents. If RPM detects no discrepancies, no output will appear. Otherwise, RPM displays a line for each file that differs from the original package contents.

8.6 Uninstalling a Package

To uninstall a package, type:

rpm -e *package*

In this command, *package* is the name of the package, not the name of the package file. The name should omit the architecture; it can also omit the package version or package version and release number. For example, you can erase the **nano** package by issuing either of the following commands:

rpm -e nano-1.2.1-4rpm -e nano

If you attempt to uninstall a package on which another package depends, RPM will report a dependency error and terminate without uninstalling the package. You can force RPM to uninstall the package by using the **?nodeps** option:

rpm -e --nodeps *package*

However, doing so will probably cause the dependent package to cease working properly. Therefore, you shouldn't use the **?nodeps** option very often.

8.7 Updating a Package

When you update (upgrade) a package, RPM installs the new version of the software but attempts to leave your existing configuration files intact. You can update a package by using the **-U** option of the **rpm** command:

```
rpm -Uvh  package
```

When you update a package, RPM automatically uninstalls the old version of the package before installing the new one.

> If no old version of the specified package exists, RPM simply installs the new version. Therefore, you can use the **-U** option to install or update a package; many Linux users avoid the **-i** (install) option, always using the **-U** option instead. However, you should use the **-i** option rather than the **-U** option when installing a package containing an updated Linux kernel. By using the **-i** option, you leave the original kernel in place, so that you can use it to boot your system if the new kernel does not work correctly with your system.

If RPM determines that your existing configuration files may be incompatible with those of the new version of the package, RPM will save a copy of the existing files. In that case, you need to examine the old and new files and determine what the proper configuration should be. The documentation that accompanies the package should assist you in this process.

If you attempt to update an existing package using an older version of the package, RPM will report an error and terminate without performing the update. To force RPM to perform the update, use the **?oldpackage** option:

```
rpm -Uvh --oldpackage  package
```

8.8 Freshening a Package

From time to time, Red Hat issues updated packages that correct functional or security-related problems with released packages. You can use Red Hat Network to obtain and install updated packages. However, if you prefer, you can download updated packages via FTP and install them using the **rpm** command. Red Hat makes updated packages available on the public FTP servers *ftp.redhat.com* and *updates.redhat.com*.

To install an updated package, use the **-F** flag, which stands for *freshen*. For instance, to install an updated version of the **gnupg** package, issue the command:

```
rpm -Fvh gnupg-1.2.1-10.i386.rpm
```

By specifying **-F** rather than **-U**, you instruct RPM to install the updated package only if an earlier version of the package is already present. This lets you use wildcards to specify entire sets of updated packages:

```
rpm -Fvh *.rpm
```

This command will not install packages that aren't already installed. But, it will install updated versions of any existing packages.

8.9 Advanced RPM Techniques

Because you invoke the **rpm** command by using the shell, just as you do any other program, you can combine options and arguments to perform a variety of useful tasks. Consider the following examples:

`rpm -Va`

Verifies every installed package. You might find this command useful if you accidentally deleted some files. The output of the command would help you determine what packages, if any, suffered damage.

`rpm -qf /usr/bin/mystery`

Displays the name of the package that owns the specified file.

`rpm -Vf /usr/bin/mystery`

Verifies the package that owns the file */usr/bin/mystery*.

`rpm -qdf /usr/bin/puzzle`

Lists the documentation files associated with the package that owns the file */usr/bin/puzzle*. This could be helpful, for example, if */usr/bin/puzzle* is a program you're having difficulty using.

8.10 Keeping Your Red Hat Enterprise Linux System Up to Date

Red Hat provides a service known as Red Hat Network (http://rhn.redhat.com), designed to help you keep your Red Hat Enterprise Linux system up-to-date and secure. Access to Red Hat Network is included with your subscription to Red Hat Enterprise Linux. If you purchase Red Hat Professional Workstation, you receive a one-year subscription to Red Hat Network. If you're using Fedora Core, you can keep your system up to date by using Yum, which is described in Section 8.11.

The Red Hat Network provides access to security alerts, bug fix alerts, and enhancement alerts published by Red Hat. Updated packages can be downloaded or automatically installed via the Red Hat Update Agent. The Red Hat Network is of significant benefit to administrators of multiple systems, who might otherwise have difficulty applying patches to close security loopholes and fix problems quickly.

8.10.1 Using Red Hat Network

To use Red Hat Network, your computer must be able to access the Internet. To set up your computer, you must perform two steps:

1. Sign up for Red Hat Network.
2. Create a system profile for your computer.

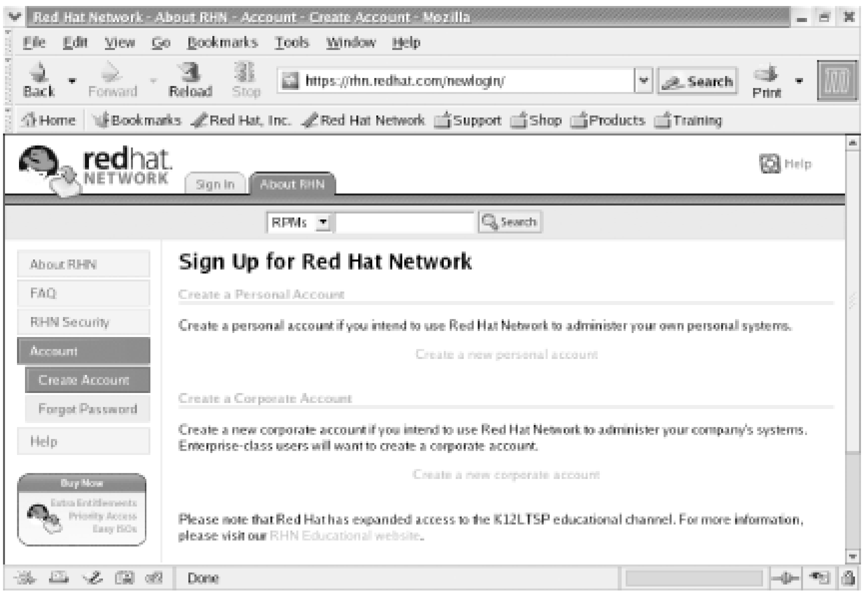
Once your computer is set up to use Red Hat Network, updates can be accomplished automatically based on a schedule or manually, by using the Red Hat Update Agent.

The following sections describe these procedures as they existed at the time of writing. However, Red Hat may revise the operation of Red Hat Network at any time. For more information on using Red Hat Network, see the *Red Hat Network User Reference Guide*, available at http://www.redhat.com/docs/manuals/RHNetwork.

8.10.1.1 Signing up

To sign up for Red Hat Network, point your browser to the main Red Hat Network web page, http://rhn.redhat.com. Then click the link marked "Create Account." The web page shown in Figure 8-6 appears.

Figure 8-6. The Red Hat Network Create Account web page



Click the link marked "Create a Personal Account." The web page shown in Figure 8-7 appears. Fill in the requested information and then click the Create Account button at the bottom of the form.

Figure 8-7. The Red Hat Network personal account sign-up page

8.11 Keeping Your Fedora Core System Up to Date

Since Fedora Core isn't officially supported by Red Hat, Fedora Core users don't have access to Red Hat Network. However, Fedora Core includes the Alert Notification and up2date tools, which can automatically install software updates.

You don't need to activate a software license in order to update your Fedora Core system, but you do need to configure your system. The procedure used to configure and update your system closely resembles that appropriate for updating Red Hat Enterprise Linux. Here are the steps you should follow:

1. Launch the Alert Notification Tool configuration sequence by right-clicking the Alert Notification Tool icon. The series of dialog boxes that appears closely resembles that related to Red Hat Enterprise Linux, described in the preceding section.

2. Launch the up2date tool by clicking the Alert Notification Tool icon. Complete the series of dialog boxes, beginning with the Welcome to Red Hat Update Agent dialog box. The series closely resembles that of Red Hat Enterprise Linux, described in the preceding section. However, it omits several dialog boxes that relate to Red Hat Network.

> If you're prompted to install the Red Hat, Inc. public GPG key, be sure to click Yes. Otherwise, system updates will fail.

Once you've configured the Alert Notification tool, its color indicates whether updates are available. If the tool is red, click the tool icon, review the available updates, and install those you consider appropriate. In general, you should install all available updates as soon as they become available. However, you may prefer to postpone installing available updates?especially large updates that entail time-consuming downloads?to a convenient time.

Chapter 9. Configuring and Administering Linux

GNOME and KDE include two menus that provide access to utilities for configuring and administering your system: System Settings and System Tools. Most of these utilities are also available via the Start Here folder. This chapter explains the utilities, equipping you to perform common system administration tasks. In particular, the chapter explains how to manage user accounts, how to configure a printer, how to configure your system's sound adapter, how to view system log files, and how to administer services. Many of these operations require root access. If you launch one of the tools when not logged in as root, the tool will conveniently ask you for the root password. Once you've logged in as the root user, the desktop will display an icon resembling a set of keys. So long as this icon appears, the system will automatically extend you root privileges when necessary. You can cancel this authorization by clicking on the icon and then clicking Forget Authorization.

9.1 Configuring Your System by Using the System Settings Menu

The GNOME and KDE System Settings menu provides access to 15 utilities that help you configure your system. Chapter 8 explained one of these, the package management tool. The tools available are:

Add/Remove Applications

Provides access to the Package Management tool, which lets you install and remove RPM packages

Authentication

Provides access to the Authentication Configuration tool, which lets you configure password settings

Date & Time

Provides access to the Date/Time Properties tool, which lets you configure the time zone, date, and time

Display

Provides access to the Display Settings tool, which lets you configure monitor resolution, color depth, and other display characteristics

Keyboard

Provides access to the Keyboard tool, which lets you choose the keyboard appropriate to your system

Language

Provides access to the Language Selection tool, which lets you choose the current language from among those for which you've installed support

Login Screen

Provides access to the GDM (GNOME Display Manager) Setup tool, which lets you configure how people log in under X

Mouse

Provides access to the Mouse Configuration tool, which lets you choose the mouse appropriate to your system and enable or disable 3-button emulation

Network

Provides access to the Network Configuration tool, which lets you configure network, modem, virtual private network (VPN), and wireless connections and related settings

Printing

Provides access to the Red Hat Printer Configuration tool, which lets you configure printers and control print queues

Root Password

Provides access to a tool that enables you to change the root password

Security Level

Provides access to the Security Level Configuration tool, which lets you configure a firewall to protect your system from network attacks

9.2 Administering Your System by Using the System Tools Menu

The System Tools menu of the GNOME and KDE desktops provides access to a set of tools that help you administer your system. The distinction between these tools and the tools provided by the System Settings menu is rather arbitrary. That is, any given tool could as likely have been placed on one menu rather than the other. Generally, items on the System Tools menu perform an action, whereas items on the System Settings menu let you configure a facility. But, exceptions to this rule of thumb abound.

The tools provided by the System Tools menu are:

Disk Management

Provides access to the User Mount Tool, which lets users mount, unmount, and format filesystems

Floppy Formatter (GNOME only)

Provides access to the Format a Floppy tool, which formats floppies

Hardware Browser

Provides access to the Hardware Browser tool, which displays information about system hardware

Internet Configuration Wizard

Provides access to a wizard that assists you in creating new network connections

Kickstart (Red Hat Enterprise Linux only)

Provides access to the Kickstart Configurator tool, which lets you specify the parameters for automated installation of Red Hat Linux

Network Device Control

Provides access to the Network Device Control tool, which lets you configure and administer network devices and connections

Printing Notification Icon

Provides an alert icon resembling a small printer. The icon appears on the panel when one or more print jobs are pending.

Print Manager

Provides a tool for managing pending print jobs.

Red Hat Network

Provides access to the Red Hat Update Agent, which assists Red Hat Network subscribers in updating their systems

Red Hat Alert Icon

Provides a panel icon that changes shape and color when updates are available for your system.

System Logs

Provides access to the System Logs tool, which lets you view the contents of system log files

System Monitor

9.3 Controlling Services by Using the Server Settings Menu

Services are generally processes that wait for a request to perform an operation or provide information and then do so. The System Settings ⟶ Server Settings menu provides menu items that let you administer certain services. The menu also provides the Services menu item associated with the Service Configuration tool, which lets you start and stop services and configure which services run at various runlevels. Red Hat Enterprise Linux and Fedora Core support seven runlevels:

0

Runlevel 0 is associated with a system shutdown.

1

Runlevel 1 is associated with the state known as single-user mode. It is generally used to troubleshoot or configure the system from a state that does not provide network services. It vaguely resembles Windows safe mode.

2

Runlevel 2 is associated with a state that provides access to the network but does not provide network services to remote hosts. Like runlevel 1, it's primarily used for troubleshooting and configuration.

3

Runlevel 3 is a normal system operating mode. It features a text-based login prompt.

4

Runlevel 4 is reserved; that is, it is not used.

5

Runlevel 5 is a normal system operating mode. It features a graphical login screen.

6

Runlevel 6 is associated with a system reboot.

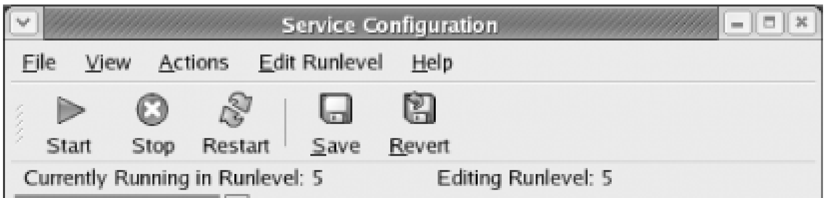
Runlevels 3 and 5 are normal system operating modes. Other modes are used transiently (levels 0, 6), for troubleshooting (levels 1, 2), or not used at all (level 4). When you associate a service with a runlevel, the service is automatically started whenever the runlevel is entered. You can set the current runlevel by issuing the command:

init *n*

where *n* is the number of the desired runlevel. However, it's seldom necessary to change the runlevel of a running system. You should do so judiciously, since services may be abruptly terminated during transition from one runlevel to another.

To launch the Service Configuration tool, select System Settings ⟶ Server Settings ⟶ Services from the main menu. Figure 9-29 shows the appearance of the tool. Note that it reports the current runlevel. It also reports the runlevel to which changes apply, called the *editing runlevel*.

Figure 9-29. The Service Configuration tool



Chapter 10. Connecting to the Internet

This chapter explains how to use Red Hat Enterprise Linux or Fedora Core to connect to the Internet via a telephone dialup, Integrated Services Digital Network (ISDN), Digital Subscriber Line (DSL), Ethernet, or wireless connection. First, it provides an overview of networking and explains how to use the Network Administration Tool, an X-based program that makes it easy to connect to the Internet via an Internet service provider (ISP). The chapter also explains how to use **wvdial**, a program that can establish a telephone dialup connection to the Internet but doesn't require X. Next, the chapter describes several popular network client applications available under Linux, including the Mozilla web browser, the Ximian Evolution email client, and a graphical FTP client. Finally, the chapter gives some tips on how to configure Linux to work with your cable or DSL modem if the Network Administration Tool is unable to do so.

10.1 Networking Overview

Computers handle network traffic much as the post office handles mail. Think, for example, of the steps involved in sending and receiving a letter. Your postal carrier must know where to drop off and where to pick up mail. So your home must have some kind of recognizable interface; we call this a mailbox. And whereas your postal carrier may know your neighborhood quite well, delivery in other areas requires other carriers. Mail is passed to these other carriers through a gateway; we call this the post office. Although you can think of the whole postal system as one big network, it's easier to understand if you think of it as a hierarchy of subnetworks (or subnets): the postal system is divided into states, states are divided into counties and cities with a range of Zip Codes, Zip Codes contain a number of streets, and each street contains a unique set of addresses.

Computer networking mirrors this model. Let's trace an email message from you to a coworker. You compose the message and click Send. Your computer passes the message to a network interface. This interface may be a modem by which you dial up an Internet service provider (ISP), or it may be via an Ethernet connection on a LAN. Either way, on the other side of the interface is a gateway machine. The gateway knows how to look at the address of the recipient of the email message and interpret that message in terms of networks and subnets. Using this information, the gateway passes the message to other gateways until the message reaches the gateway for the destination machine. That gateway in turn delivers the message via a recognizable interface (such as a modem or Ethernet link) to the recipient's inbox.

If you review this analogy, you can easily see which parts of networking you'll need to configure on your Linux system. You'll need to know the address of your machine. Just as the town name Sebastopol and the Zip Code 95472 are two different names for the same location, you may have both a name, called a hostname, and a number, called an IP number or IP address, that serve as the address for your machine.

To translate between these two notations, you may need to know the address of a Domain Name Server (DNS). This is a machine that matches IP addresses with hostnames. You'll also need to know the address of a gateway machine through which network traffic will be routed. Finally, you'll need to be able to bring up a network interface on your system, and you'll need to assign a route from that interface to the gateway.

While all of this can seem complex, it really isn't any more complex than the postal system, and it functions in much the same way. Fortunately, Linux comes with tools to help you automate network configuration.

10.2 Configuring an Internet Connection

The Network Administration Tool simplifies configuration of your system to access the Internet via a telephone dialup, ISDN, DSL, Ethernet, or wireless connection. The Network Administration Tool requires you to follow a three-step process:

1. Set up the hardware device associated with the connection.
2. Specify DNS settings and hostnames.
3. Activate the device, if necessary.

The following subsections explain how to perform these steps.

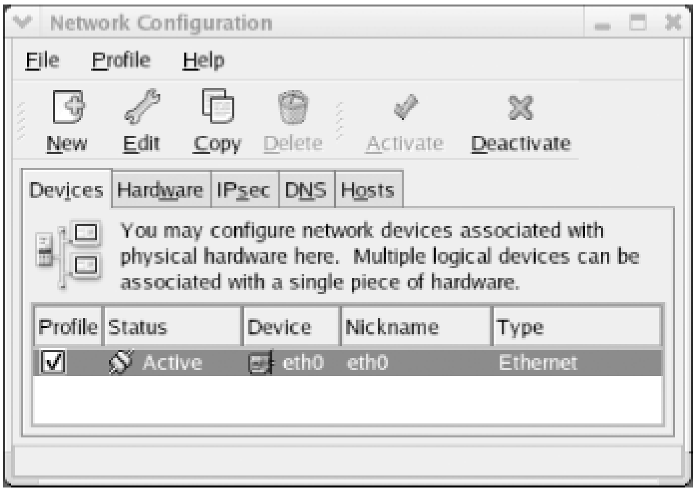
> The Network Administration tool supports all these methods of connecting to the Internet. However, some hardware devices are not compatible with Red Hat Enterprise Linux and Fedora Core. And, some Internet service providers insist that their customers use only Windows. In either case, you may experience difficulties in connecting to the Internet. The final section of this chapter explains some means of last resort that might help you resolve problems.

10.2.1 Setting Up Hardware Devices

In the past, most computer users connected to the Internet via a plain old telephone service (POTS) dial-up modem. However, today, many means of connecting to the Internet are available. For example, many home computer users have high-speed connections using ISDN or DSL. Corporate computer users often connect to the Internet via a local area network, using an Ethernet adapter. And wireless network adapters are becoming quite popular.

To set up a hardware device using the Network Administration Tool, choose System Settings Network from the main menu. The Network Administration Tool appears, as shown in Figure 10-1. Although the screen is titled "Network Configuration," the tool is nevertheless commonly known as the Network Administration Tool.

Figure 10-1. The Network Administration Tool



The Network Administration Tool has five tabs:

Devices

Used to associate a physical device with a network connection

Hardware

Used to set up a physical device

IPsec

10.3 The Mozilla Web Browser

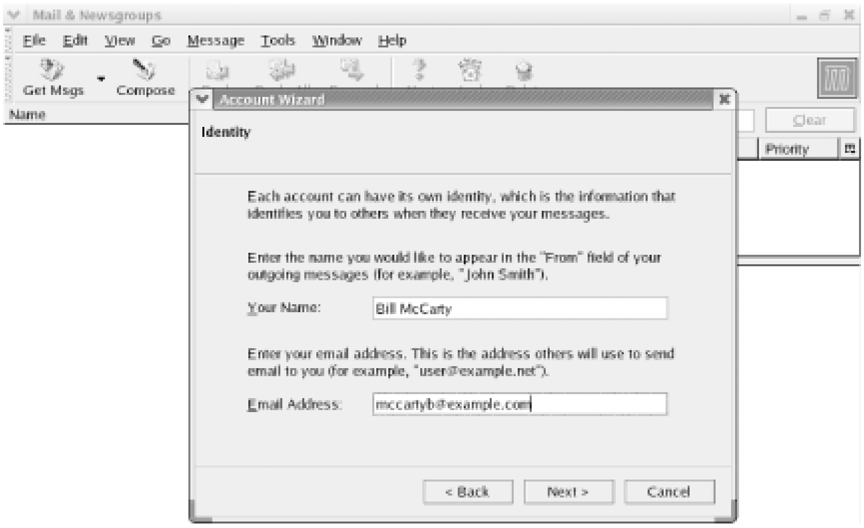
Once you've established a connection to the Internet, you can surf the Web using Mozilla, the default Red Hat Enterprise Linux and Fedora Core web browser. To launch Mozilla, choose Internet ⟶ Web Browser from the main menu. Mozilla resembles its closed source ancestor, Netscape Navigator. So, if you've used Navigator, you'll feel at home in Mozilla.

> The Web contains quite a bit of malicious software. You should log in as an ordinary user?that is, a user other than the root user?to surf the Web. That way, malicious software is less likely to be able to compromise your system.

When you launch Mozilla for the first time, you may see a dialog box explaining that a Mozilla profile is being created from existing Netscape 4 files. Click Convert Profile to allow Mozilla to create the profile.

Mozilla includes email and news clients that are easily configured. It also includes a web page composer and address book. To configure Mozilla email, choose Window ⟶ Mail & Newsgroups from the Mozilla menu. An Account wizard appears, asking whether you want to configure an email or newsgroup account. Select Email Account and click Next. The Identity dialog box (Figure 10-15) appears. Specify your name and email address and click Next.

Figure 10-15. The Identity dialog box



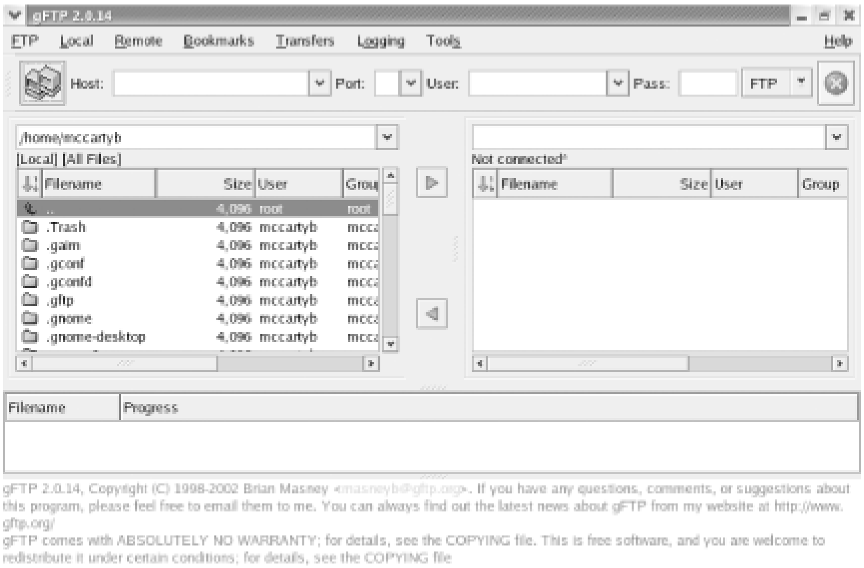
The Server Information dialog box (Figure 10-16) appears. Specify which protocol your mail server uses, POP (Post Office Protocol) or IMAP (Interim Mail Access Protocol). POP servers require you to download email. IMAP servers let you read email that resides on the server. Most up-to-date ISPs support IMAP, the newer protocol. Also specify the hostnames of your incoming and outgoing mail services. A single host may fill both roles. If you don't know this information, you can obtain it from your ISP. Click Next to continue.

Figure 10-16. The Server Information dialog box

10.4 gFTP FTP Client

You can use your web browser to download files from an FTP server, but to upload files you need an FTP client. The gFTP client, included with Red Hat Enterprise Linux and Fedora Core, is an excellent choice, because its user interface resembles that of popular Windows FTP clients, such as WS-FTP. Figure 10-18 shows the gFTP client, which can be launched by choosing Internet ⟶ More Internet Applications ⟶ gFTP from the main menu.

Figure 10-18. The gFTP FTP client



To connect to a remote system, specify the hostname, username, and password in the textboxes appearing on the toolbar. If the server permits anonymous logins, you can omit the username and password. To connect, click the Connect icon resembling a pair of computers at the left of gFTP's toolbar. To upload a file, click on the name of the file in the local list box at the left of the window and then click on the right-pointing arrow. To download a file, click on the name of the file in the list box at the right of the window and then click on the left-pointing arrow. When you've transferred all your files, choose Remote ⟶ Disconnect or click again on the Connect icon.

> You can access an FTP server in command-line mode, if you prefer. Chapter 12 explains how to do so.

10.5 Using wvdial

If you have your dial-up network connection working perfectly, you may have little interest in exploring **wvdial**. However, there are two reasons you should consider learning more about **wvdial**: you can use **wvdial** even if X isn't working or isn't installed, and you can use **wvdial** in shell scripts of your own design. Chapter 13 includes an example script.

10.5.1 The /etc/wvdial.conf File

To configure **wvdial**, become the root user and issue the following command:

```
# wvdialconf /etc/wvdial.conf
```

This command analyzes your system and creates a template configuration file, */etc/wvdial.conf*. You must edit this file to specify the username and password your ISP expects.

The contents of the template file look something like this:

```
[Dialer Defaults]
Modem = /dev/modem
Baud = 115200
Init1 = ATZ
Init2 = ATQ0 V1 E1 S0=0 &C1 &D2 S11=55 +FCLASS=0
ISDN = 0
; Phone = <Target Phone Number>
; Username = <Your Login Name>
; Password = <Your Password>
```

Edit the last three lines of the file, deleting the leading semicolon and space and substituting the proper phone number, username, and password required to connect to your ISP. When you're done, your file should look something like this:

```
[Dialer Defaults]
Modem = /dev/modem
Baud =  115200
Init1 = ATZ
Init2 = ATQ0 V1 E1 &C1 &D2 +FCLASS=0
ISDN = 0
Phone = 15625551100
Username = bill100
Password = donttell
```

> Be sure that only the root user can read the *wvdial.conf* file. Use a file manager or the shell to change the file's permissions, if necessary. Otherwise, someone who uses your system may discover your password.

Now, you're ready to make a connection by issuing the following command:

```
# wvdial &
```

The command generates quite a bit of output, which makes further use of this virtual terminal distracting. The simplest solution is to switch to another terminal window, or to another virtual terminal by pressing **Alt-***n*, where *n* stands for the virtual terminal (1-7). Alternatively, you can direct the output of the command to a file, by typing this command in place of the one given earlier:

```
# wvdial 2>/tmp/wvdial.messages &
```

Of course, you'll need to consult the file if something goes wrong with **wvdial**. Do so by using the **less** command:

```
# less /tmp/wvdial.messages
```

Once your connection is up, you can browse the Web and access other Internet services. For now, simply verify that your connection is working by issuing the command:

```
# ping www.redhat.com
```

The **ping** command should report that echo packets were successfully received from the

10.6 Configuring Linux to Use a Cable or DSL Modem

At one time, setting up a cable or DSL modem for use with Linux was difficult. The new Network Administration Tool included in Red Hat Enterprise Linux and Fedora Core often makes it a snap. But sometimes, the Tool fails to successfully configure the cable or DSL modem.

In such a case I recommend using a cable/DSL gateway router. Netgear and Linksys, among others, manufacture popular models. These inexpensive devices?often costing less than $50 retail?sit between your cable or DSL modem and your home network. Cable/DSL gateway routers generally provide a masquerading firewall and DNS proxy services. Better models have multiple ports so that you can connect several PCs without buying additional hubs or switches. Some recent models even provide a wireless LAN. I recently purchased a very satisfactory gateway router having four Ethernet ports and a built-in wireless adapter for $30, net of rebates.

Because such devices are designed to work with as many cable and DSL configurations as possible, they work right out of the box most of the time. It's true that they generally provide no function that couldn't be provided?at least in principle?by a Linux PC. But they consume less power, occupy less space, make less noise, and require less configuration and administration than a Linux PC. I retired a quite venerable Pentium 166 Linux PC from cable modem gateway duty several years ago and have never regretted the decision.

Some useful resources when setting up Linux to access a cable or DSL modem include:

Hal Burgiss' DSL HOWTO for Linux, http://www.tldp.org/HOWTO/DSL-HOWTO

> This somewhat dated HOWTO examines the technology behind DSL and offers advice on subscription, installation, configuration, and troubleshooting for Linux users.

Paul Ramey's Red Hat Linux 6.X as an Internet Gateway for a Home Network, http://www.tldp.org/HOWTO/mini/Home-Network-mini-HOWTO.html

> This HOWTO relates to an obsolete version of Red Hat Linux, but nevertheless contains much useful information.

These resources are both a bit out of date, but are the most current available. That they haven't been recently updated seems to demonstrate that most Linux users are using gateway routers rather than Linux hosts as their means of connecting to their cable or DSL provider.

Chapter 11. Setting Up Network Services

Linux's greatest strength is its powerful and robust networking capabilities. The good news is that everything about Linux's networking setup is open to inspection and completely configurable. Nothing is hidden from the user, and no parameters are forced on you. The challenge is to get the most out of this setup.

Basic networking principles don't differ much between Windows and Linux, and indeed the principles aren't unfamiliar. This chapter begins with an overview of networking and then looks in more detail at Linux networking on a local area network (LAN) and the Intertnet.

This chapter explains how to set up a LAN that includes a Linux Samba server, which lets Microsoft Windows and Unix systems access shared files and printers across the network. Samba not only lets you share files and printers, it can also be used to back up and restore files via the network.

This chapter also explains how to set up and configure the Apache web server and an SSH server, which you can use to remotely administer your system. These applications let you and others access data on your Linux system via the Internet. These applications will be most useful if your system is connected to the Internet 24/7. But, even if your connection is intermittent, you and others can access the services these applications provide whenever the connection is active.

Finally, the chapter also explains how to implement a basic host firewall to help protect your systems from unauthorized access via the Internet. Security is best when it consists of multiple layers. So, you're strongly urged not to connect any host to the Internet except through a network firewall. To help you ensure that your host and network firewalls are working, the chapter explains how to use *Nmap* to test your security.

If your host is protected?as it should be?by a host or network firewall, hosts on the other side or the firewall will not be able to connect to services offered by your host. To permit such hosts to access services, you must disable or customize the network and host firewalls. Ask your network firewall administrator to enable access to your host's services. Also, you must disable your host firewall, as described in Section 11.5.2. However, you should disable the firewall only if your host resides on a trusted LAN that is protected from access by Internet hosts and other untrusted hosts.

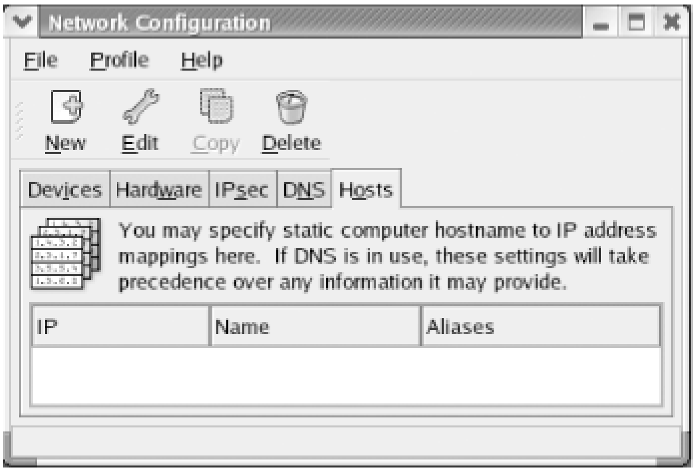
> Previous editions of this book provided instructions on setting up a DHCP server that enables you to manage network configurations centrally. However, Red Hat Enterprise Linux WS does not provide a DHCP server. If you need a DHCP server for your LAN, you may find it convenient to purchase a gateway router of the sort described in Chapter 10. Many gateway routers provide a simple DHCP server adequate for use on home and small business LANs.
>
> Most Internet services are configurable only by the root user. So, most of the operations in this chapter require that you be logged in as root, or possess temporary root privileges as indicated by the keys icon.

11.1 Configuring Hosts

Most systems attached to a network use DNS services to determine the IP address associated with a hostname. However, private hosts on your local network won't be known to your ISP's DNS server and therefore can be accessed only by IP address, not by hostname. Moreover, Samba and other local area network services won't operate correctly unless they can resolve hostnames.

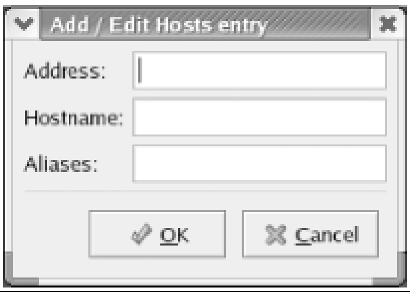
Fortunately, using the Network Administration Tool, you can configure your system to determine the IP address associated with a hostname even when DNS services are not available. To do so, use the Hosts tab of the Network Administration Tool, shown in Figure 11-1.

Figure 11-1. The Hosts tab of the Network Administration Tool



To specify host information, launch the Network Administration Tool by choosing System Settings ⟶ Network from the main menu. Select the Hosts tab of the Tool, click New. The Add/Edit Hosts entry dialog box appears (Figure 11-2). Specify the IP address and name of the host. If desired, you can specify one or more aliases or abbreviated names for the host. By default, the host information includes an entry for the IP address 127.0.0.1, which is associated with the hostname *localhost*. The Add/Edit Hosts entry dialog box does not display this entry, which protects it from accidental modification.

Figure 11-2. The Add/Edit Hosts entry dialog box



Linux stores configured hostnames in the file */etc/hosts*, which you can edit by using a text editor, if you prefer. Windows has a similar file that you must revise in order for Windows hosts to interoperate with Samba and other Linux-based services. The file is named *hosts* and resides in an *etc* subdirectory of your Windows directory. Under Windows 2000, the path of the file is often *c:\windows\system32\drivers\etc\hosts*; the location varies slightly depending on the installation options.

Each line of the file describes a single host. The left column contains the host's IP address and the right column contains the associated hostname or names. If multiple names (aliases) appear, the principal name is given first. Simply use a text editor, such as Notepad, to add the appropriate lines describing your local hosts.

11.2 Samba

Once you've configured your hostnames, you're ready to provide services to hosts on the network. To provide printer and file sharing, Windows uses a facility known as the Server Message Block (SMB). This same facility is sometimes known as the Common Internet File System (CIFS), NetBIOS, or LanManager. Thanks to Andrew Tridgell and others, Linux systems provide support for SMB via a package known as Samba. Like SMB, Samba lets you:

- Authorize users to access Samba resources
- Share printers and files among Windows, OS/2, Netware, and Unix systems
- Establish a simple name server for identifying systems on your LAN
- Back up PC files to a Linux system and restore them

Samba has proven its reliability and high performance in many organizations. According to the survey at http://www.samba.org/pub/samba/survey/ssstats.html, Bank of America is using Samba in a configuration that includes about 15,000 clients, and Hewlett-Packard is using Samba in a configuration that includes about 7,000 clients.

11.2.1 Installing the Samba Server

If you've never installed and configured a network server, Samba is a good place to begin; its installation and configuration are generally straightforward.

The Samba server includes the **nmbd** and **smbd** programs (which run as daemons), several utility programs, manpages and other documentation, and three configuration files: */etc/samba/smbusers*, */etc/samba/smb.conf*, and */etc/samba/lmhosts*. The *smbusers* file associates several user accounts that are special to Samba with Linux user accounts; for example, it associates the Samba user IDs, *administrator* and *admin*, with *root*. Generally, you don't need to change *smbusers*. Likewise, you don't generally need to revise *lmhosts*. You'll learn how to configure the *smb.conf* file shortly.

The simplest way to install Samba is to select the Windows File Server package group during system installation. However, if you failed to do so, you can install Samba by using the Package Management Tool.

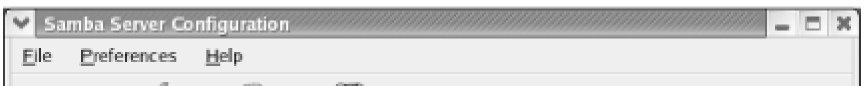
11.2.2 Configuring Samba

The */etc/samba/smb.conf* file lets you specify a variety of options that control Samba's operation. You can edit the file by using your favorite text editor; however, the Samba Server Configuration Tool enables you to view and change options using your web browser, which is generally much easier than using a text editor. The Samba Server Configuration tool verifies the values of parameters you enter and provides online help.

To install the Samba Server Configuration Tool, select System Settings ⟶ Add/Remove Applications from the main menu. The Package Management Tool appears. Select Server Configuration Tools ⟶ Details ⟶ redhat-config-samba. Select Close ⟶ Update, and insert the appropriate installation CDs when prompted. To launch the tool, select System Settings ⟶ Server Settings ⟶ Samba from the main menu. Figure 11-3 shows the Samba Server Configuration screen.

> The Samba menu item may not appear immediately following the installation of the Windows File Server package. If your system's menu lacks the Samba menu item, log out and log in again. Doing so should cause your desktop manager to regenerate the menus, causing the Samba menu item to appear.

Figure 11-3. The Samba Server Configuration screen



11.3 Running Apache

Installing and configuring the Apache web server is not difficult. Once your web server is up and running, other Internet users can view and download documents within the web-enabled directories on your Linux system. This section explains the installation and configuration of Apache, the most popular web server on the Internet.

11.3.1 Installing Apache

Use the Package Management Tool to install the Web Server package group, which contains the Apache web server. This package group contains over two dozen optional packages. Unless you specifically require the support they provide, you should deselect them before initiating the installation. However, you should not generally deselect the optional *redhat-config-httpd* package, which is helpful in configuring Apache.

> When you install Apache, the procedure may fail owing to missing packages. It appears that Red Hat's developers erroneously specified some required packages in the Apache web server package group as optional. Select the following packages for installation and try again:

- *commons-logging*
- *xerces-j*
- *jakarta-regexp*
- *bcel*
- *redhat-java-rpm-scripts*
- *xalan-j*

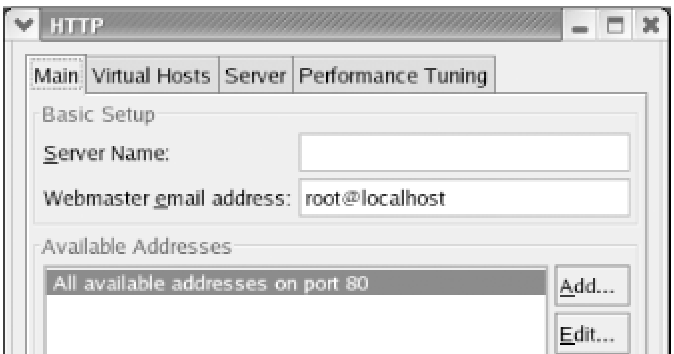
11.3.2 Configuring Apache

Configuring a web server can be as easy or as difficult as you choose. Like other web servers, Apache provides seemingly countless options. As distributed with Red Hat Enterprise Linux and Fedora Core, Apache has a default configuration that generally requires only a little tweaking before use. Apache's configuration files reside in the directory */etc/httpd/conf*. For historical reasons that no longer apply, Apache has three configuration files:

- *access.conf*
- *httpd.conf*
- *srm.conf*

However, the only configuration file that's currently used is *httpd.conf*. The easiest way to perform a basic configuration of Apache is with the Apache Configuration Tool. To configure Apache, choose System Settings ⟶ Server Settings ⟶ HTTP from the main menu. The main configuration screen (Figure 11-10) appears.

Figure 11-10. The Apache Configuration Tool



11.4 The Secure Shell

The Secure Shell (SSH) lets you connect to a system from another system via TCP/IP and obtain a shell prompt, from which you can issue commands and view output in a secure fashion. SSH works similarly to the older and possibly more familiar Telnet service, but differs in that conversations between SSH and its clients are sent in encrypted form so hackers cannot easily discover private information, including user account names and passwords.

11.4.1 Installing SSH

The installation procedure automatically installs an SSH client and server and associates the *sshd* service with runlevels 3-5. You can start, stop, and restart the *sshd* service and changes its associations with runlevels by using the Service Configuration Tool. The service must be running in order to respond to clients.

> The SSH service has several configuration files, residing in */etc/ssh*. You don't have to modify them to get SSH running. If you're curious about them, view the *sshd* manpage.

11.4.2 Using SSH

To verify that the SSH server is properly running, you can access it via a client on the local system by issuing the following command:

$ **ssh localhost**

The client will attempt to log you on to the local system using your current user account and will prompt you for your password. If you supply the correct password, you should see a shell prompt, indicating that the client and server are functioning correctly. Type **exit** and press **Enter** to exit SSH.

To log on to a remote system, simply specify the hostname or IP address of the remote system in place of *localhost*. If you want to log in to a user account other than one named identically to the account you're using on the local system, issue the command:

$ **ssh** *userid* @ *host*

where *host* is the hostname or IP address of the remote host and *userid* is the name of the user account you want to use. For example:

$ **ssh billmccarty@example.com**

You can use the SSH client's **scp** command to transfer files to or from a remote system running an SSH server. To transfer a file to a remote system, issue a command such as this one:

$ **scp** *file* *userid* @ *host* : *destination*

where *file* is the path of the file to be transferred, *host* is the hostname or IP address of the remote host, *destination* is the directory to which the file should be transferred, and *userid* is your user account on the remote system. If given as a relative path, the destination path is understood as being relative to the home directory of the specified user. For example:

$ **scp rhbook_rev.txt billmccarty@example.com:files**

To transfer files to your home directory on the remote system, omit the **path** argument; however, retain the colon or the command will be misinterpreted.

You can specify multiple files to be transferred if you like. You can use shell metacharacters to specify a set of files to be transferred. You can also specify the **-r** flag, which specifies that **scp** should recursively copy a directory. For example, the following command copies an entire directory to the remote system:

$ **scp -r Desktop billmccarty@example.com:files**

To transfer files from a remote system, issue a command based on this pattern:

$ **scp** *userid* @ *host* : *file* *path*

where *host* is the hostname or IP address of the remote system, *file* is the path of the file to

11.5 Implementing a Basic Firewall

Sometimes you may want a host to provide certain services to only local clients or clients on other hosts of a network that you control. If your network is connected to the Internet, you can use a *firewall* to prevent undesired access to services. A Linux firewall depends on certain kernel facilities to examine incoming and outgoing packets. Packets that fail to pass specified rules can be rejected, preventing undesired access to private services. Unlike TCP Wrappers, a firewall does not require special support from applications or services it protects. And, a firewall can work with protocols other than TCP, such as UDP and ICMP.

11.5.1 Configuring the Firewall

To configure a firewall, launch the Security Level Tool by choosing System Settings ⟶ Security Level from the main menu.

The Security Level Tool (Figure 11-17) appears.

Figure 11-17. The Security Level Configuration dialog box

The Security Level Configuration dialog box lets you enable or disable the firewall. In addition, you can specify that requests for any of several predefined services are allowed to freely transit the firewall. To do so, simple enable the checkbox associated with the name of the service.

> By allowing service requests to transit the firewall, you may expose your system to network-based attacks. Therefore, don't specify trusted services casually or unnecessarily.

You can use the Trusted devices checkbox to specify that packets originating from the specified device will not be blocked by the firewall. This facility is useful when a host has two network adapters: one associated with a public network, such as the Internet, and another associated with a private network. By specifying the network adapter associated with the private network as a trusted device, you permit clients on the private network free access to services, while blocking clients on the public network from access other than that permitted by the firewall configuration.

11.5.2 Controlling the Firewall

For the firewall to be effective, it must be enabled and the associated *iptables* service must be running. To start, stop, or restart the *iptables* service, you can use the Service Configuration Tool. Generally, you should use the Tool to associate the *iptables* service with runlevels 2-5, so that your system is protected when networking is active.

11.6 Checking Your Security by Using Nmap

Many users have misconfigured firewalls such that they are all but useless in defending against attacks. You can determine the state of your firewall by using the same tool hackers use to find its weaknesses, Nmap. To install Nmap, use the Package Management Tool to install the System Tools package group, and the extra package *nmap*.

Unfortunately, Nmap doesn't have a place on the GNOME or KDE menus. Instead, you use Nmap by issuing shell commands from a terminal window.

Nmap has enough features to be the topic of a book. You can learn more about Nmap from its author's web site, http://www.insecure.org/. A vanilla use of Nmap involves configuring it to send TCP or UDP packets to every important port of a specified system. The target system's responses reveal whether it has services listening on scanned ports.

To scan your system, open a terminal window, become the root user and issue a command having the form:

```
# nmap -sT -sU -p 1-1023 xxx.xxx.xxx.xxx
```

where *xxx.xxx.xxx.xxx* is the IP address of your system. After a few seconds, Nmap reports the ports on which services are listening:

```
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on  (192.168.83.131):
(The 2042 ports scanned but not shown below are in state: closed)
Port       State       Service
22/tcp     open        ssh
68/udp     open        dhcpclient
80/tcp     open        http
123/udp    open        ntp


Nmap run completed -- 1 IP address (1 host up) scanned in 5 seconds
```

Some ports listed in Nmap's report may not be remotely accessible. Therefore, to accurately determine the state of a system, you must scan it remotely. If a remote scan of your system discloses unexpected open ports, you should check your firewall and service configurations to determine whether anything is amiss.

Open ports above port 1023 that come and go with successive scans are not generally cause for concern. Usually, these are associated with established connections and are not really open. However, they can also be an early indication of the presence of a Trojan horse installed by an attacker. It's good practice to scan your hosts after you configure them so that you can later distinguish normal from abnormal behavior.

> Don't scan someone else's system without permission. In some jurisdictions, it's a crime to do so. Even when scanning is legal, it may violate your ISP's terms of use. And, even if your ISP doesn't object, the scanned host's administrator may believe that the host is under attack, diverting his or her attention from other business. So, it's best to scan only systems on your local network, under your direct administrative control.

11.7 Network Security Tips

Anyone who administers a system connected to the Internet needs to know something about network security. It's not uncommon for systems connected to the Internet to be probed by would-be hackers several times daily. If a would-be hacker manages to detect a vulnerability, the hacker can often exploit it in a matter of seconds. Therefore, it's almost certain that a system administrator ignorant of network security will eventually suffer a system break-in.

Network security is a large and sophisticated topic that can be only cursorily surveyed in a book such as this. Concerned readers should consult books such as the following:

- *Building Internet Firewalls*, Second Edition, by Elizabeth D. Zwicky, Simon Cooper, and D. Brent Chapman (O'Reilly)
- *Building Secure Servers with Linux*, by Michael D. Bauer (O'Reilly).
- *Computer Security Basics*, by Deborah Russell and G.T. Gangemi, Sr. (O'Reilly)
- *Linux Security Cookbook*, by Daniel J. Barrett, Richard Silverman, Robert G. Byrnes (O'Reilly).
- *Linux Server Hacks*, by Rob Flickenger (O'Reilly).
- *Practical Unix & Internet Security, 3rd. ed*., by Simson Garfinkel, Gene Spafford, and Alan Schwartz (O'Reilly).
- *Red Hat Linux Firewalls*, by Bill McCarty (Red Hat Press).

If a sufficiently skilled hacker is intent on compromising a system you administer, the hacker will probably succeed. However, here are some tips that can help you avoid falling victim to amateur hackers:

- Use a network or host firewall to prevent outsiders from accessing services you don't need to make publicly available.
- Monitor security web sites and mailing lists so that you're aware of recent threats and the associated countermeasures. The CERT Coordination Center, http://www.cert.org, provides many useful resources.
- Apply bug fixes promptly after Red Hat Network advises you that they are available.

Chapter 12. Advanced Shell Usage and Shell Scripts

Like an MS-DOS prompt window, the Unix shell is a command interpreter that lets you issue and execute commands. By means of the shell, you use and control your system. If you're accustomed to the point-and-click world of graphical user interfaces, you may question the value of learning to use the Linux[1] shell. Many users initially find the shell cumbersome, and some retreat to the familiar comfort of the graphical user interface (GUI), avoiding the shell whenever possible. However, as this chapter explains, the shell unlocks the true power of Linux.

[1] Most of the statements in this chapter are true of Linux generally, rather than merely true of Red Hat Enterprise Linux WS and Fedora Core. However, in this chapter, I use the term *Linux* as a concise way of referring to Red Hat Enterprise Linux WS and Fedora Core.

12.1 The Power of the Unix Shell

While it's true that the shell is an older style of interacting with a computer than the GUI, the graphical user interface is actually the more primitive interface. The GUI is easy to learn and widely used, but the shell is vastly more sophisticated. Using a GUI is somewhat like communicating in American Indian sign language. If your message is a simple one, like "We come in peace," you can communicate it by using a few gestures. However, if you attempted to give Lincoln's Gettysburg address ?a notably brief public discourse ?you'd find your task quite formidable.[2]

[2] American Sign Language, used to communicate with those who have a hearing impairment, is a much richer language than American Indian sign language.

The designer of a program that provides a GUI must anticipate all the possible ways in which the user will interact with the program and provide ways to trigger the appropriate program responses by means of pointing and clicking. Consequently, the user is constrained to working only in predicted ways. The user is therefore unable to adapt the GUI program to accommodate unforeseen tasks and circumstances. In a nutshell, that's why many system administration tasks are performed using the shell: system administrators, in fulfilling their responsibility to keep a system up and running, must continually deal with and overcome the unforeseen.

The shell reflects the underlying philosophy of Unix, which provides a wide variety of small, simple tools (that is, programs), each performing a single task. When a complex operation is needed, the tools work together to accomplish the complex operation as a series of simple operations, one step at a time. Many Unix tools manipulate text, and since Unix stores its configuration data in text form rather than in binary form, the tools are ideally suited for manipulating Unix itself. The shell's ability to freely combine tools in novel ways is what makes Unix powerful and sophisticated. Moreover, as you'll learn, the shell is extensible: you can create *shell scripts* that let you store a series of commands for later execution, saving you the future tedium of typing or pointing and clicking to recall them.

The contrary philosophy is seen in operating systems such as Microsoft Windows, which employ elaborate, monolithic programs that provide menus, submenus, and dialog boxes. Such programs have no way to cooperate with one another to accomplish complex operations that weren't anticipated when the programs were designed. They're easy to use so long as you remain on the beaten path, but once you step off the trail, you might find yourself in a confusing wilderness.

Of course, not everyone shares this perspective. The Usenet newsgroups, for example, are filled with postings debating the relative merits of GUIs. Some see the Unix shell as an arcane and intimidating monstrosity. But, even if they're correct, it's inarguable that when you learn to use the shell, you begin to see Unix as it was intended (whether that's for better or for worse).

When you are performing common, routine operations, a GUI that minimizes typing can be a relief, but when faced with a complex, unstructured problem that requires a creative solution, the shell is more often the tool of choice. Creating solutions in the form of shell scripts allows solutions to be stored for subsequent reuse. Perhaps even more important, shell scripts can be studied to quickly bone up on forgotten details, expediting the solution of related problems.

12.2 Filename Globbing

Before the shell passes arguments to an external command or interprets a built-in command, it scans the command line for certain special characters and performs an operation known as *filename globbing*. Filename globbing resembles the processing of wildcards used in MS-DOS commands, but it's much more sophisticated. Table 12-1 describes the special characters, known as filename metacharacters, used in filename globbing.

Table 12-1. Filename metacharacters

Metacharacter	Function
*	Matches a string of zero or more characters
?	Matches exactly one character
[abc...]	Matches any of the characters specified
[a-z]	Matches any character in the specified range
[!abc...]	Matches any character other than those specified
[!a-z]	Matches any character not in the specified range
~	The home directory of the current user
~userid	The home directory of the specified user
~+	The current working directory
~-	The previous working directory

In filename globbing, just as in MS-DOS wildcarding, the shell attempts to replace metacharacters appearing in arguments in such a way that arguments specify filenames. Filename globbing makes it easier to specify names of files and sets of files.

For example, suppose the current working directory contains the files *file1*, *file2*, *file3*, and *file04*. Suppose you want to know the size of each file. The following command reports that information:

```
ls -l file1 file2 file3 file04
```

However, the following command reports the same information and is much easier to type:

```
ls -l file*
```

As Table 12-1 shows, the `*` filename metacharacter can match any string of characters. Suppose you issued the following command:

```
ls -l file?
```

The `?` filename metacharacter can match only a single character. Therefore, *file04* would not appear in the output of the command.

Similarly, the command:

```
ls -l file[2-3]
```

would report only *file2* and *file3*, because only these files have names that match the specified pattern, which requires that the last character of the filename be in the range 2-3.

You can use more than one metacharacter in a single argument. For example, consider the following command:

12.3 Shell Aliases

Shell aliases make it easier to use commands by letting you establish abbreviated command names and by letting you pre-specify common options and arguments for a command. To establish a command alias, issue a command of the form:

```
alias name='command'
```

where *command* specifies the command for which you want to create an alias and *name* specifies the name of the alias. For example, suppose you frequently type the MS-DOS command **dir** when you intend to type the Linux command **ls -l**. You can establish an alias for the **ls -l** command by issuing this command:

```
alias dir='ls -l'
```

Once the alias is established, if you mistakenly type **dir**, you'll get the directory listing you wanted instead of the default output of the **dir** command, which resembles **ls** rather than **ls -l** . If you like, you can establish similar aliases for other commands.

Your default Linux configuration probably defines several aliases on your behalf. To see what they are, issue the command:

```
alias
```

If you're logged in as *root*, you may see the following aliases:

```
alias cp='cp -i'
alias l.='ls -d .* --color=tty'
alias ll='ls -l --color=tty'
alias ls='ls -color=tty'
alias mv='mv -i'
alias rm='rm -i'
alias vi='vim'
alias which='alias | /usr/bin/which -tty-only -read-alias -show-dot
-show-tilde'
```

Notice how several commands are self-aliased. For example, the command **rm -i** is aliased as **rm**. The effect is that the **-i** option appears whenever you issue the **rm** command, whether or not you type the option. The **-i** option specifies that the shell will prompt for confirmation before deleting files. This helps avoid accidental deletion of files, which can be particularly hazardous when you're logged in as *root*. The alias ensures that you're prompted for confirmation even if you don't ask to be prompted. If you don't want to be prompted, you can issue a command like:

```
rm -f files
```

where *files* specifies the files to be deleted. The **-f** option has an effect opposite that of the **-i** option; it forces deletion of files without prompting for confirmation. Because the command is aliased, the command actually executed is:

```
rm -i -f files
```

The **-f** option takes precedence over the -**i** option, because it occurs later in the command line.

If you want to remove a command alias, you can issue the **unalias** command:

```
unalias alias
```

where *alias* specifies the alias you want to remove. Aliases last only for the duration of a login session, so you needn't bother to remove them before logging off. If you want an alias to be effective each time you log in, you can use a shell script, which we'll discuss later in the chapter.

12.4 Using Virtual Consoles

You can use a terminal window to issue shell commands. However, you can issue shell commands even when X is not running or available. To do so, you use the Linux virtual console feature.

Linux provides six virtual consoles for interactive use; a seventh virtual console is associated with the graphical user interface. You can use special keystrokes to switch between virtual consoles. The keystroke **Alt-F***n*, where *n* is the number of a virtual console (1-6), causes Linux to display virtual console *n*. For example, you can display virtual console 2 by typing **Alt-F2**. You can view only a single console at a time, but you can switch rapidly between consoles by using the appropriate keystroke. The keystroke **Alt-F7** causes Linux to enter graphical mode using virtual console 7.

Virtual consoles also have a screensaver feature like that found on Windows. If a virtual console is inactive for an extended period of time, Linux blanks the monitor. To restore the screen without disturbing its contents, simply press the **Shift** key.

12.4.1 Logging In

To log in using a virtual console, respond to the login prompt by typing your user ID and pressing **Enter**. The system prompts you for the password associated with your account. Type the proper password and press **Enter**. To prevent anyone nearby from learning your password, Linux does not display your password as your enter it. If you suspect you've typed it incorrectly, you can either hit the **Backspace** key a number of times sufficient to delete the characters you've entered and type the password again or simply press **Enter** and start over. If you type the user ID or password incorrectly, Linux displays the message "login incorrect" and prompts you to try again.

When you've successfully logged in, you'll see a command prompt that looks something like this:

```
[bill@home bill]$
```

If you logged in as the root user, you'll see a prompt that contains a hash mark (`#`); if you've logged in as an ordinary user, you'll see a dollar sign (`$`). The prompt tells you that the Linux *bash* shell is ready to accept your commands.

12.4.2 Logging Out

When you're done using a virtual console, you should log out by typing the command **exit** and pressing **Enter**. When you log out, the system frees memory and other resources that were allocated when you logged in, making those resources available to other users and processes.

When the system logs you out, it immediately displays a login prompt. If you change your mind and want to access the system, you can log in simply by supplying your username and password.

12.5 X and the Shell

You can configure your system to boot into nongraphical mode, if you prefer. If your video adapter is not compatible with X, you have no alternative but to do so. However, some Linux users having X-compatible video adapters prefer to configure their system to boot into nongraphical mode. A simple command lets such users launch an X session whenever they wish.

12.5.1 Configuring a Nongraphical Login

Linux provides several runlevels. Each runlevel has an associated set of services. For instance, runlevel 3 is associated with a text-based login and run level 5 is associated with an X-based, graphical login. Changing runlevels automatically starts and stops services associated with the old and new run levels.

You can determine the current runlevel by issuing the following command:

```
# runlevel
```

The output of the command shows the previous and current runlevels. For example, the output:

```
3 5
```

indicates that the current run level is 5 and that the previous run level was 3.

To change the current runlevel, issue the **init** command. For example, to enter runlevel 3, issue the following command while logged in as *root*:

```
# init 3
```

In response to this command, the system will start and stop services as required to enter runlevel 3.

The */etc/inittab* file specifies the default runlevel, which the system enters when booted. By changing the default run level to 3, you can configure your system to provide a nongraphical login when it boots. To do so, log in as *root* and load the */etc/ inittab* file into the *nano* editor by issuing the command:

```
# nano /etc/inittab
```

Find the line that reads:

```
id:5:initdefault:
```

Change the 5 to a 3:

```
id:3:initdefault:
```

Save the file and exit *nano*. The next time you boot your system, it will automatically enter runlevel 3 and provide a nongraphical login screen.

12.5.2 Starting and Stopping X from a Text-Based Login

To start X from a text-based login, type the command:

```
$ startx
```

Your system's screen should briefly go blank and then you should see X's graphical desktop.

> If the screen is garbled or remains blank for more than about 30 seconds, your X configuration may be faulty. Immediately turn off your monitor or terminate X by pressing **Ctrl-Alt-Backspace**.

When running X from a text-based login, you can gracefully quit X by merely exiting the desktop. To do so, select Logout from the main menu. To quit X abruptly, press **Ctrl-Alt-Backspace**.

12.6 Shell Scripts

A shell script is simply a file that contains a set of commands to be run by the shell when invoked. By storing commands as a shell script, you make it easy to execute them again and again. As an example, consider a file named *deleter*, which contains the following lines:

```
echo -n Deleting the temporary files...
rm -f *.tmp
echo Done.
```

The **echo** commands simply print text on the console. The **-n** option of the first **echo** command causes omission of the trailing newline character normally written by the **echo** command, so both **echo** commands write their text on a single line. The **rm** command removes all files having names ending in *.tmp* from the current working directory.

You can execute this script by issuing the **sh** command, as follows:

```
$ sh deleter
```

> If you invoke the sh command without an argument specifying a script file, a new interactive shell is launched. To exit the new shell and return to your previous session, issue the exit command.

If the *deleter* file were in a directory other than the current working directory, you'd have to type an absolute path, for example:

```
$ sh /home/bill/deleter
```

You can make it a bit easier to execute the script by changing its access mode to include execute access. To do so, issue the following command:

```
$ chmod 555 deleter
```

This gives you, members of your group, and everyone else the ability to execute the file. To do so, simply type the absolute path of the file, for example:

```
$ /home/bill/deleter
```

If the file is in the current directory, you can issue the following command:

```
$ ./deleter
```

You may wonder why you can't simply issue the command:

```
$ deleter
```

In fact, this still simpler form of the command will work, so long as *deleter* resides in a directory on your search path. You'll learn about the search path later.

Linux includes several standard scripts that are run at various times. Table 12-2 identifies these and gives the time when each is run. You can modify these scripts to operate differently. For example, if you want to establish command aliases that are available whenever you log in, you can use a text editor to add the appropriate lines to the *.profile* file that resides in your */home* directory. Since the name of this file begins with a dot (.), the **ls** command won't normally show the file. You must specify the **-a** option in order to see this and other hidden files.

Table 12-2. Special shell scripts

Script	Function
/etc/profile	Executes when the user logs in
~/.bash_profile	Executes when the user logs in
~/.bashrc	Executes when *bash* is launched
~/.bash_logout	Executes when the user logs out

12.7 Understanding Shell Scripts

This section explains how more advanced shell scripts work. The information is also adequate to equip you to write many of your own useful shell scripts. The section begins by showing how to process a script's arguments. Then it shows how to perform conditional and iterative operations.

12.7.1 Processing Arguments

You can easily write scripts that process arguments, because a set of special shell variables holds the values of arguments specified when your script is invoked. Table 12-6 describes the most popular such shell variables.

Table 12-6. Special shell variables used in scripts

Variable	Meaning
`$#`	The number of arguments
`$0`	The command name
`$1, $2, ... ,$9`	The individual arguments of the command
`$*`	The entire list of arguments, treated as a single word
`$@`	The entire list of arguments, treated as a series of words
`$?`	The exit status of the previous command; a value of 0 denotes a successful completion
`$$`	The ID of the current process

For example, here's a simple one-line script that prints the value of its second argument:

```
echo My second argument has the value $2.
```

Suppose you store this script in the file *second*, change its access mode to permit execution, and invoke it as follows:

```
$ ./second a b c
```

The script will print the output:

```
My second argument has the value b.
```

Notice that the shell provides variables for accessing only nine arguments. Nevertheless, you can access more than nine arguments. The key to doing so is the **shift** command, which discards the value of the first argument and shifts the remaining values down one position. Thus, after executing the **shift** command, the shell variable `$9` contains the value of the 10th argument. To access the 11th and subsequent arguments, you simply execute the **shift** command the appropriate number of times.

12.7.2 Exit Codes

The shell variable `$?` holds the numeric exit status of the most recently completed command. By convention, an exit status of zero denotes successful completion; other values denote error conditions of various sorts.You can set the error code in a script by issuing the **exit** command, which terminates the script and posts the specified exit status. The format of the command is:

```
$ exit status
```

where *status* is a nonnegative integer that specifies the exit status.

Appendix A. Linux Directory Tree

Table A-1 describes the directories in the Linux directory tree.

Table A-1. The Linux directory tree

Directory	Description
/bin	User programs and scripts essential to system startup
/boot	Boot information, including the kernel
/dev	Device files
/etc	Host-specific configuration files
/etc/sysconfig	Stores configuration files specific to Red Hat Linux
/home	Users' home directories
/initrd	Used during boot process as a mount point for a directory containing special device drivers
/lib	Libraries, modules, and other object files
/lib/modules	Loadable kernel modules
/lost+found	Recovered data from bad clusters
/mnt	Temporarily mounted filesystems
/opt	Used to store large applications
/proc	Kernel pseudo-directory that provides access to kernel information and configuration items
/root	System administrator's home directory
/sbin	System administration programs and scripts essential to system startup
/tmp	Temporary files, which are automatically deleted by Red Hat Linux
/usr	Files needed for system operation but not needed to boot system (can be mounted read-only, except when being changed by root user)
/usr/bin	Programs and scripts not essential to system startup
/usr/dict	System dictionaries for spell checking
/usr/etc	Configuration files

Appendix B. Principal Linux Files

Table B-1 describes the principal Linux files. You can use it, for example, to help you locate configuration files quickly.

Table B-1. Principal Linux files

File(s)	Description
/boot/grub/grub.conf	GRUB configuration file
*/boot/module-info-**	Module information for the Linux kernel
*/boot/System.map-**	Map of the Linux kernel
*/boot/vmlinuz-**	Linux kernel
/etc/aliases	Mail aliases
/etc/at.deny	User IDs of users forbidden to use the `at` command
/etc/auto.master	Configuration file for the *autofs* daemon, which automatically mounts filesystems
/etc/auto.misc	Automounter map file
/etc/bashrc	Systemwide functions and aliases for the *bash* shell
*/etc/cron.daily/**	Daily *cron* jobs
*/etc/cron.hourly/**	Hourly *cron* jobs
*/etc/cron.monthly/**	Monthly *cron* jobs
*/etc/cron.weekly/**	Weekly *cron* jobs
/etc/crontab	System *cron* file
*/etc/cups/**	Printer configuration files
/etc/default/useradd	Defaults for the `useradd` command
/etc/DIR_COLORS	Directory listing colors
/etc/exports	NFS exported directories
/etc/filesystems	Supported filesystem types
/etc/fstab	Filesystems mounted or available for mounting
/etc/group	System group definitions

Appendix C. Managing the Boot Process

In this appendix, you'll learn more about how to boot a Linux system; in particular, you'll learn more about configuring your computer system to boot any of several operating systems. The appendix focuses on GNU GRUB, the Grand Unified Bootloader, the most popular utility for booting Linux systems. Most users won't need to use the information in this appendix. But, this information can help you troubleshoot and resolve boot-related problems.

> Red Hat Enterprise Linux and Fedora Core can be booted using bootloaders other than GRUB, such as LILO. Occasionally, LILO is able to boot a system that can't be booted using GRUB. To learn more about LILO, see the related section of the *Red Hat Linux 9 Reference Guide*, available at http://www.redhat.com/docs/manuals/linux/RHL-9-Manual/ref-guide/s1-grub-lilo.html.

C.1 Booting Linux

When you boot a PC, you cause it to execute a small program known as a *boot loader*. The purpose of the boot loader is to locate and read into memory the first stage of an operating system and transfer control to it. The operating system then locates and reads its remaining components as needed.

The simplest way to boot Linux is by using a floppy diskette. By doing so, you're able to leave the boot information on your hard drive untouched, ensuring that you can still boot Microsoft Windows or another operating system stored on the same hard drive. Moreover, some operating systems and virus protection programs prevent modification of the boot information on your hard drive. By booting from a floppy diskette, you avoid several potential problems.

However, many users find booting from a floppy disk slow or inconvenient. You don't have to boot Linux from a floppy diskette; you can boot it in any of several other ways. The most popular method is by using GRUB, which replaces the boot loader stored on your hard drive.

This appendix cannot describe the entire range of issues involved in booting Linux. Much of the information here is taken from several Linux HOWTOs that contain additional useful information on booting Linux:

- *BootPrompt-HOWTO*
- *CD-Writing-HOWTO*
- *CDROM-HOWTO*
- *Ethernet-HOWTO*
- *Ftape-HOWTO*
- *Hardware-HOWTO*
- *Multi-Disk-HOWTO*
- *PCI-HOWTO*
- *PCMCIA-HOWTO*

C.2 Boot Floppies

Even if you don't want to boot Linux from a floppy diskette, you should create and keep on hand a Linux boot floppy. If something goes wrong with your system, preventing you from booting in the normal way, you may be able to boot your system by using the floppy. Then, you can diagnose and repair the problem and get back to business as usual.

C.2.1 Creating a Boot Floppy

You can easily create a boot diskette that can be used to start your system. To do so, insert a blank floppy diskette into your system's floppy drive. Log on as *root* and issue the following command:

```
# /sbin/mkbootdisk version
```

For *version*, supply the version number of your kernel. If you don't recall the version, execute the following command, which reports it:

```
# uname -a
```

The version number resembles the version number of an RPM package. For example, a typical kernel version number is 2.4.18-14.

The **mkbootdisk** command creates a boot floppy that uses the same kernel running when the command is issued. It also configures the boot floppy to load any necessary SCSI modules, so that your SCSI drives will be accessible after booting from the floppy.

C.2.2 Using a Boot Floppy

Insert the boot floppy into your system's floppy drive. If your system is turned off, power up your system. If your system is turned on, first shut down the active operating system in the proper manner, then restart the system. Linux should then boot from the floppy.

> To use your boot floppy, your system's CMOS must be configured to allow booting from the floppy drive. If your system boots from its hard drive even when the boot floppy is present, you must change your system's CMOS configuration. The relevant option is generally named Boot Sequence, Boot Order, or something similar. The value you want is generally labeled *A*:, *C*:, or something similar. Consult your system's documentation for further information.

C.3 The GRUB Loader

Most PCs can be booted from a floppy drive or hard drive; most recently manufactured computers can be booted from a CD-ROM drive. The first sector of a disk, diskette, or partition is known as the *boot sector*. The boot sector associated with a disk or diskette (the first sector of the disk or diskette) is known as the *Master Boot Record* (MBR). In order for a diskette or disk to be bootable, it must contain a boot loader, which can reside in:

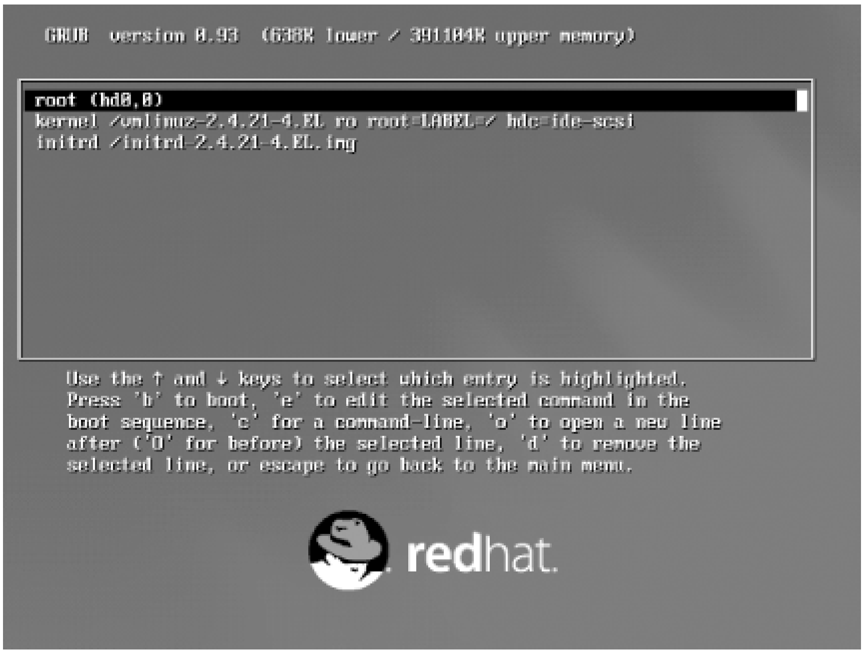
- The boot sector of the floppy diskette
- The MBR of the first hard disk or the first CD-ROM drive, if the PC supports booting from a CD-ROM
- The boot sector of a Linux filesystem partition on the first hard disk
- The boot sector of an extended partition on the first hard disk

GRUB is a sophisticated boot loader that can load Linux, Microsoft Windows 3.*x* and 9*x*, NT, 2000, XP, and other popular operating systems. Most users install GRUB on the MBR of their system's first hard disk. That way, when the system is started, it boots GRUB, which can be used to load Linux, Microsoft Windows, or another operating system.

Unless you direct otherwise, the Red Hat Linux installation procedure automatically installs GRUB. So you don't need to install GRUB; you just need to configure it.

Similarly, when you boot by using GRUB, you can also boot parameters to control the boot process; you can specify GRUB's boot parameters by selecting an operating system from GRUB's menu and pressing **e**. In response, GRUB displays an editor screen that shows the commands associated with the selected operating system, as shown in Figure C-1.

Figure C-1. The GRUB editor screen



Commands that can be used in the editor screen are listed in Table C-1.

Table C-1. GRUB editor commands

Command	Meaning
B	Boot the currently selected operating system.
E	Edit the currently selected GRUB command.
C	Open a screen for interactively entering and executing GRUB commands.

C.4 Boot Parameters

Boot parameters are specified using a three-part directive that includes the name of the parameter and an optional list of options, which consists of an equal sign (`=`) followed by a comma-separated list of option values

No spaces may appear in the directive. As an example, the following directive specifies the identity of the Linux root partition:

```
root=/dev/hda1
```

The installation program generally refers to partitions by using labels, so that you can boot a system even if you move the partitions around. A root directive referring to a label looks like this:

```
root=LABEL=/usr
```

You can specify multiple directives by separating them with a space. For example, the following specifies the identity of the Linux root partition and that the root partition is initially mounted read-only, so that a thorough check of its filesystem can be performed:

```
root=/dev/hda1 ro
```

Most directives are interpreted by the kernel, though GRUB is also capable of processing directives. If you specify a directive that neither the kernel nor GRUB understands (assuming you're using GRUB), a directive that includes an equal sign is passed to the *init* process as an environment variable. You learned about environment variables in Chapter 12. A nonkernel directive that doesn't include an equal sign is passed to the *init* process. An example of this usage is specifying the directive `single`, which causes *init* to start your system in single-user mode:

```
root=/dev/hda1 ro single
```

C.4.1 General Boot Arguments

Table C-2 describes some of the most popular and useful boot arguments. These arguments apply to your system as a whole; in subsequent sections, you'll learn about other boot arguments that apply to specific devices or functions. In addition to boot arguments previously introduced, the table describes the `reserve` argument, which is helpful in avoiding system memory conflicts.

Table C-2. Selected general boot arguments

Argument	Description and options
`init=`	Specifies arguments passed by the kernel to the `init` process.
`mem=`	Specifies the amount of physical memory available to Linux; lets you instruct Linux to avoid high memory areas used by some systems for BIOS or caching. You can specify the amount as a hexadecimal number or as a decimal number followed by `k` or `M`, denoting kilobytes or megabytes, respectively.
`reserve=`	Specifies I/O ports that must not be probed. The port number is specified by using a hexadecimal number, and the range is specified by using a decimal number. For example, `reserve=0x320,32` specifies that I/O ports 320-33f must not be probed.
`ro`	Initially mounts the root filesystem in read-only mode, so that a more effective filesystem check can be done.
	Identifies the root filesystem:/dev/fdn Floppy disk *n* (0 or 1) */dev/hdxn*

Appendix D. Linux Command Quick Reference

The following list describes some of the most useful and popular Linux commands. Consult the manpage for each command to learn about additional arguments and details of operation.

adduser *userid*

Creates a new *userid*, prompting for necessary information (requires *root* privs).

alias name='*command*'

Defines name as an alias for the specified command.

apropos *keyword*

Searches the manual pages for occurrences of the specified keyword and prints short descriptions from the beginning of matching manual pages.

at *time*

at -f *file time*

Executes commands entered via STDIN (or by using the alternative form, the specified file) at the specified time. The time can be specified in a variety of ways; for example, in hour and minute format (*hh:mm*) or in hour, minute, month, day, and year format (*hh:mm mm/dd/yy*).

atq

Displays descriptions of jobs pending via the **at** command.

atrm *job*

Cancels execution of a job scheduled via the **at** command. Use the **atq** command to discover the identities of scheduled jobs.

bg

bg *jobs*

Places the current *job* (or by using the alternative form, the specified jobs) in the background, suspending its execution so that a new user prompt appears immediately. Use the **jobs** command to discover the identities of background jobs.

cal *month year*

Displays a calendar for the specified month of the specified year.

cat *files*

Displays the contents of the specified files.

cd

cd *directory*

Changes the current working directory to the user's home directory or the specified directory.

chgrp *group files*

chgrp -R *group files*

Colophon

Our look is the result of reader comments, our own experimentation, and feedback from distribution channels. Distinctive covers complement our distinctive approach to technical topics, breathing personality and life into potentially dry subjects.

The cover image of a man wearing a wide-brimmed hat is a 19th-century engraving from *Marvels of the New West: A Vivid Portrayal of the Stupendous Marvels in the Vast Wonderland West of the Missouri River*, by William Thayer (The Henry Bill Publishing Co., 1888).

Philip Dangler was the production editor and copyeditor for *Learning Red Hat Enterprise Linux and Fedora, Fourth Edition*. Darren Kelly, Matt Hutchinson, and Claire Cloutier provided quality control. Angela Howard wrote the index.

Edie Freedman designed the cover of this book. Emma Colby produced the cover layout with QuarkXPress 4.1 using Adobe's ITC Garamond font. David Futato designed the CD labels.

David Futato designed the interior layout. The chapter opening images are from the Dover Pictorial Archive, *Marvels of the New West, and The Pioneer History of America: A Popular Account of the Heroes and Adventures*, by Augustus Lynch Mason, A.M. (The Jones Brothers Publishing Company, 1884). This book was converted by Julie Hawks to FrameMaker 5.5.6 with a format conversion tool created by Erik Ray, Jason McIntosh, Neil Walls, and Mike Sierra that uses Perl and XML technologies. The text font is Linotype Birka; the heading font is Adobe Myriad Condensed; and the code font is LucasFont's TheSans Mono Condensed. The illustrations that appear in the book were produced by Robert Romano and Jessamyn Read using Macromedia FreeHand 9 and Adobe Photoshop 6. The tip and warning icons were drawn by Christopher Bing.

The online edition of this book was created by the Safari production group (John Chodacki, Becki Maisch, and Madeleine Newell) using a set of Frame-to-XML conversion and cleanup tools written and maintained by Erik Ray, Benn Salter, John Chodacki, and Jeff Liggett.

[SYMBOL] [A] [B] [C] [D] [E] [F] [G] [H] [I] [J] [K] [L] [M] [N] [O] [P] [Q] [R] [S] [T] [U] [V] [W] [X] [Y] [Z]

[SYMBOL] [A] [B] [C] [**D**] [E] [F] [G] [H] [I] [J] [K] [L] [M] [N] [O] [P] [Q] [R] [S] [T] [U] [V] [W] [X] [Y] [Z]

[SYMBOL] [A] [B] [C] [D] [E] [F] [G] [H] [I] [J] [K] [L] [M] [N] [O] [P] [Q] [R] [S] [T] [U] [V] [W] [X] [Y] [Z]

[SYMBOL] [A] [B] [C] [D] [E] [F] [G] [H] [I] [J] [K] [L] [M] [N] [O] [P] [Q] [R] [S] [T] [U] [V] [W] [X] [Y] [Z]

[SYMBOL] [A] [B] [C] [D] [E] [F] [G] [H] [I] [J] [K] [L] [M] [**N**] [O] [P] [Q] [R] [S] [T] [U] [V] [W] [X] [Y] [Z]

[SYMBOL] [A] [B] [C] [D] [E] [F] [G] [H] [I] [J] [K] [L] [M] [N] [O] [P] [Q] [R] [S] [T] [**U**] [V] [W] [X] [Y] [Z]