

Red Hat® Linux® Networking and System Administration

3rd Edition

Terry Collings
and Kurt Wall



DVD INCLUDES FULL
FEDORA CORE 4



Red Hat[®] Linux[®] Networking and System Administration

Third Edition

Terry Collings and Kurt Wall



WILEY

Wiley Publishing, Inc.

**Red Hat[®] Linux[®] Networking
and System Administration
Third Edition**



Red Hat® Linux® Networking and System Administration

Third Edition

Terry Collings and Kurt Wall



WILEY

Wiley Publishing, Inc.

Red Hat® Linux® Networking and System Administration, Third Edition

Published by

Wiley Publishing, Inc.

10475 Crosspoint Boulevard

Indianapolis, IN 46256

www.wiley.com

Copyright © 2005 by Wiley Publishing, Inc., Indianapolis, Indiana

Published simultaneously in Canada

ISBN-13: 978-0-7645-9949-1

ISBN-10: 0-7645-9949-6

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923, (978) 750-8400, fax (978) 646-8600. Requests to the Publisher for permission should be addressed to the Legal Department, Wiley Publishing, Inc., 10475 Crosspoint Blvd., Indianapolis, IN 46256, (317) 572-3447, fax (317) 572-4355, or online at <http://www.wiley.com/go/permissions>.

Limit of Liability/Disclaimer of Warranty: The publisher and the author make no representations or warranties with respect to the accuracy or completeness of the contents of this work and specifically disclaim all warranties, including without limitation warranties of fitness for a particular purpose. No warranty may be created or extended by sales or promotional materials. The advice and strategies contained herein may not be suitable for every situation. This work is sold with the understanding that the publisher is not engaged in rendering legal, accounting, or other professional services. If professional assistance is required, the services of a competent professional person should be sought. Neither the publisher nor the author shall be liable for damages arising herefrom. The fact that an organization or Website is referred to in this work as a citation and/or a potential source of further information does not mean that the author or the publisher endorses the information the organization or Website may provide or recommendations it may make. Further, readers should be aware that Internet Websites listed in this work may have changed or disappeared between when this work was written and when it is read.

For general information on our other products and services or to obtain technical support, please contact our Customer Care Department within the U.S. at (800) 762-2974, outside the U.S. at (317) 572-3993 or fax (317) 572-4002.

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic books.

Trademarks: Wiley, the Wiley Publishing logo and related trade dress are trademarks or registered trademarks of John Wiley & Sons, Inc. and/or its affiliates in the United States and other countries, and may not be used without written permission. Red Hat is a registered trademark of Red Hat, Inc. Linux is a registered trademark of Linus Torvalds. All other trademarks are the property of their respective owners. Wiley Publishing, Inc., is not associated with any product or vendor mentioned in this book.



About the Authors

Terry Collings is the owner of TAC Technology, located in eastern Pennsylvania. He provides Linux consulting and training services to a variety of clients.

Terry has been an adjunct faculty member at several colleges in his area where he has taught A+ and Network+ certification courses. He also has taught courses on Unix, Linux, TCP/IP, and Novell Netware.

Terry is the author of *Red Hat Enterprise Linux 4 For Dummies* and has co-authored and contributed to several other Linux books. He has been a technical editor for the following books: *KDE Bible*, *The Samba Book*, *Unix Weekend Crash Course*, *Red Hat Linux 9 For Dummies*, *Solaris 9 For Dummies*, *Fedora Linux 2 For Dummies*, and *Linux Timesaving Techniques For Dummies*.

Kurt Wall first touched a computer in 1980 when he learned FORTRAN on an IBM mainframe of forgotten vintage; things have improved since then. A professional technical writer by trade, a historian by training, and an all-around Linux guy by avocation, Kurt's work history is diverse. These days, Kurt works in the Customer Engineering group at TimeSys Corporation in Pittsburgh, Pennsylvania. His primary responsibilities include building and maintaining TimeSys's Developer Exchange and working with portal customers and users. He also fixes broken servers, writes documentation, and builds TimeSys software.

Kurt, who dislikes writing about himself in the third person, receives entirely too much e-mail at kwall@kurtwerks.com.

Credits

Acquisitions Editor

Debra Williams Cauley

Development Editor

Sydney Jones

Technical Editor

William von Hagen

Production Editor

Angela Smith

Copy Editor

Foxxe Editorial Services

Editorial Manager

Mary Beth Wakefield

Production Manager

Tim Tate

Vice President & Executive Group**Publisher**

Richard Swadley

Vice President and Publisher

Joseph B. Wikert

Graphics and Production Specialists

Carrie Foster

Denny Hager

Jennifer Heleine

Stephanie D. Jumper

Ron Terry

Quality Control Technicians

Amanda Briggs

John Greenough

Susan Moritz

Joe Niesen

Proofreading and Indexing

TECHBOOKS Production Services

*This book is dedicated to my wife, Nancy, and daughter, Sabrina,
who bring joy and wonder every day.*

—Terry Collings

*To my new wife, Kelly, who is indeed flesh
of my flesh and bone of my bone.*

—Kurt Wall



Preface

Red Hat produces the most popular distribution of Linux currently in use. It is a robust, reliable operating system that can run on a variety of hardware, from personal computers to large mainframes. Linux in general, Fedora Core 4 and Red Hat Enterprise Linux in particular, are very powerful operating systems that can be used at the enterprise level as a full-fledged server. Linux functions equally well at the enterprise-workstation level for typical user applications, as well as on home PCs. For those of us dissatisfied with the reliability and security of other commercially available operating systems, Fedora Core 4 and Red Hat Enterprise Linux are a pleasant alternative.

How This Book Is Organized

This book is divided into five parts and one appendix, each covering a specific area of functionality in a typical Fedora Core 4 and Red Hat Enterprise Linux system. In this book, the third edition, we have added more chapters that cover areas we discussed in the first and second editions in more detail or that explore material not covered in the first or second editions. With this edition, the book now contains 35 chapters and a rather large appendix, a considerable increase in content since the first edition was released three years ago. We want to emphasize that this book is useful for users of Fedora Core, the open source community-based Linux project supported by Red Hat, as well as users of Red Hat Enterprise Linux.

Part I: System and Network Administration Defined

This part sets the stage and defines the role of a system administrator, beginning with an explanation of the duties of a system administrator and continuing through installing your system and finishing with descriptions of the file system and system configuration files. Chapter 1 explains some of the common tasks an administrator may perform, such as installing servers and application software, managing user accounts, and backing up and restoring files. Chapter 2 details the steps involved in planning and implementing a network, including security and disaster-recovery considerations. Chapter 3 covers all the steps required to install Fedora Core or Red Hat Enterprise Linux on a local system using the most typical installation method. Chapter 4 gives you instructions on using Kickstart to perform system installations on remote systems. Chapter 5 gives you an introduction to the GNOME and KDE graphical user environments and helps you find your way around the desktop. Chapter 6 teaches you about the startup and shutdown process of your system, including the GRUB boot loader and the init process. In Chapter 7, you explore the details of the file system hierarchy and learn about other supported file systems. Part I ends with Chapter 8, which lists the system and network configuration files and explains the purpose of each file.

Part II: Network Services

This part of the book is where you learn about the networking services available in Fedora Core and Red Hat Enterprise Linux. Beginning with Chapter 9, you learn about the X Window system used to provide a graphical working environment as well as font management. Chapter 10 tells you how to configure your printers to use the Common Unix Printing System (CUPS), the default printing system used by Fedora Core and Red Hat Enterprise Linux. In Chapter 11, you learn about the TCP/IP protocol suite and how to configure it on your system. Chapter 12 explains the configuration of the Network File System (NFS) used to share files with other Linux or UNIX computers on your network. Chapter 13 gives you the details about the Network Information System (NIS) and configuration instructions. If you have computers running Microsoft Windows NT, 2000, or XP, you will want to read Chapter 14 to learn how to share files with them using Samba. Chapter 14 also provides instructions on connecting a client to Novell networks so you can share files with these systems as well. Chapter 15 gives you the details of installing and configuring an Oracle database on your server. In Chapter 16 you learn about setting up a VNC server to provide remote access with a graphical interface. Chapter 17 is all about convenience and some of the convenience services you

can provide with your system. The last chapter in this part, Chapter 18, gives you some helpful tips for optimizing the services discussed in Part II.

Part III: Internet Services

Internet services are somewhat different from network services on an internal network, and Chapter 19 begins this part by explaining what we mean by Internet services. Included in this chapter is an explanation of the Xinetd and TCP wrappers configuration files. The ability to convert domain names to IP addresses is a fundamental part of providing Internet services. Chapter 20 explains how to configure BIND on your system to provide this service. The next three chapters provide installation and configuration instructions for three commonly used Internet services. Chapter 21 describes the process of sending e-mail and how to configure Sendmail, the most commonly used mail transfer agent, as well as Postfix, which is quickly gaining popularity. Chapter 22 explains setting up an FTP server on your system. Chapter 23 covers the most widely used Web server, Apache, and explains the configuration process. In Chapter 24 you learn about other common Web services that you can provide. The last chapter in Part III, Chapter 25, provides some optimization information for the services covered in this part of the book.

Part IV: System Administration

The goal of this part of the book is to provide enough information so you have a fundamental understanding of the tasks required to maintain your system and ensure that it runs well. Chapter 26 explains the `up2date` program that is included with Fedora Core and Enterprise Linux that you can use to keep your system updated. Also covered is the Red Hat Network, a subscription service available with Red Hat Enterprise Linux that you can use to keep your system current. You can register your systems with Red Hat and then receive automatic notifications of updated or new software that can be installed. Sometimes it is advantageous to upgrade or recompile your kernel for your specific needs. Chapter 27 discusses the pros and cons of making changes and provides instructions to recompile your kernel. If you would rather do your system configuration from a command prompt instead of using many of the available GUI tools, Chapter 28 is for you. This chapter provides command prompt configuration instructions, as well as instructions to create scripts to automate many routine administration tasks. Chapter 29 tells you all you need to know to effectively manage the users and groups on your system. In Chapter 30, you learn how to install and upgrade software packages on your system. And in the last chapter in this part, Chapter 31, you explore the process of backing up the files on your system and how to restore them.

Part V: System Security and Problem Solving

Most of the last part of the book deals with performance monitoring and tuning, and securing your system, with a final chapter on general system troubleshooting. Maintaining a secure system is a critical area of concern for system administrators. Chapter 32 explains the basic steps involved in monitoring your system's performance to keep it running as quickly as it should. Chapter 33 addresses a new topic in this edition, SELinux, the access-based security system developed by the National Security Agency. Continuing the discussion of security, Chapter 34 gives you an explanation of firewalls and Internet security and the risks involved with connections from outside your network. You also learn about LDAP and Kerberos and their role in network security. The last chapter in this part, Chapter 35, provides some general troubleshooting tips and techniques and lists some problems you may encounter during normal operation of your system and the steps to take to solve the problems discussed.

Appendix A

This appendix is new to this edition. We had a lot of information about shell scripting and couldn't find a good place for it in the parts, so we put it here. If you want to become a shell-scripting pro, read this section.

How to Use This Book

Our intention in this book is to cover the Fedora Core and Red Hat Enterprise Linux operating system in enough detail to provide the answers you need. The book is divided into the parts previously discussed to make it easy for you to go to the specific part for the topic you need to learn about. You can use the book as a reference for whatever you need to know about a particular topic.

Using This Book's Icons

Look for the following margin icons to help you get the most out of this book:

TIP Tips provide special information or advice.

CAUTION Caution icons warn you of a potential problem or error.

CROSS-REFERENCE Cross-references direct you to related information in another section or chapter.

NOTE Notes highlight areas of interest or special concern related to a topic.

Conventions

This book uses the following conventions for explanations of how to do things on your computer:

- *Italic type* introduces new technical terms. It also indicates replaceable arguments that you should substitute with actual values — the context makes clear the distinction between new terms and replaceable arguments.
- **Bold** shows a command you type.
- Monospaced text distinguishes commands, options, and arguments from surrounding explanatory content.
- Keys to press in combination are shown as in this example:
Ctrl+Alt+Delete means to press all three keys at the same time.
- The term click means to press the left mouse button once. Double-click means to press the left button twice in quick succession. Right-click means to press the right mouse button once. Drag means to hold down the left mouse button and move the mouse while holding down the button.



Acknowledgments

Terry Collings: My first thought when I was asked to write the third edition of this book was Wow! Now we are doing a third edition, so what can I say now? It appears that we did a good-enough job on the first and second editions that many people bought the book and found it useful. So to everyone who bought the first and second editions of the book and made it possible for us to do yet another edition, here's a big thank you!

Thanks to Kurt Wall, my co-author, for again doing a great job in our collaboration on this new edition. Kurt is very easy to work with, and I hope I'll have a chance to work with him again. Thanks, Kurt, and I wish you all the best in your recent marriage!

This book would not have been possible without the hard work of everyone at John Wiley & Sons, especially our acquisitions editor, Debra Williams Cauley, and our development editor, Sydney Jones. Debra and Sydney are both consummate professionals who were always there to answer our questions or concerns and make sure we kept on track. Thanks go to our copy editor, technical editor, and production staff at Wiley for ensuring that our book is technically accurate and grammatically correct.

Finally, I would like to thank my wife, Nancy, for all the hours she spent keeping our daughter Sabrina entertained so I could work undisturbed completing this new edition.

Kurt Wall: I agree with Terry: thanks to everyone who bought the previous editions of this book, which made it possible for us to write this one. Unlike Terry, though, I knew what I'd say because many of you contacted me to let me know what we'd missed. Thank you. It is a privilege and an honor to write for you; I hope this book is worthy of that trust.

I'm grateful to Terry for again allowing me to work on this book with him. Let's do this again, eh? As usual, the staff at Wiley has been terrific, despite the fact that they drove us to meet an insane deadline. Debra Williams Cauley, our Acquisitions Editor and Voice of Deadlines Missed, is just a doll; Sydney Jones worked hard to mash the text into something presentable. Thank you Debra and Sydney. Kudos to the unsung, unnamed others who converted the manuscript into printable copy and who never get enough credit.

Our technical editor and my friend and colleague from TimeSys, Bill von Hagen, helped us write a tighter, more relevant book. Bill also gets an award for keeping me sane during YAB (Yet Another Book) and for keeping me entertained during YADATO (Yet Another Day At The Office). Thanks, Bill! Thanks also to Christopher Faylor, another TimeSys colleague, for reviewing the chapter on RPM and to Tim Wunder, who suggested improvements in the Web services chapter. Any remaining mistakes are either figments of your imagination or my responsibility.

Tim Wunder and the other residents of the Linux Step-by-Step mailing list (<http://mail.linux-sxs.org/cgi-bin/mailman/listinfo/linux-users>) were quite forthcoming with ideas and input when I asked for it. They've been a great group of people with whom to converse these last 12 years and are a big part of the reason I keep getting to write books about Linux. Thanks guys. Hats off to Red Hat Software and the Fedora Core Project earn mention here for providing our subject matter.

My agent, Marta Justak, is happy to see me finally nail a deadline but wants to know what Debra does that she couldn't. Beats me! Thanks to Kevin Bartlett for miscellaneous corrections and to Krischan Jodies for his excellent `ipcalc` tool. Above all, if I have any talent as a writer, credit goes to God who gave me the talent, provided me the opportunities to develop and use it, and kept me sober long enough to do so. Thanks and Amen.



Contents

Preface	ix
Acknowledgments	xv
Part One System and Network Administration Defined	1
Chapter 1 Duties of the System Administrator	3
The Linux System Administrator	3
Installing and Configuring Servers	5
Installing and Configuring Application Software	6
Creating and Maintaining User Accounts	7
Backing Up and Restoring Files	7
Monitoring and Tuning Performance	9
Configuring a Secure System	10
Using Tools to Monitor Security	12
Summary	12
Chapter 2 Planning the Network	13
Deciding How Your Network Will Be Used	13
Understanding Topologies	15
Star Topology	15
Bus Topology	16
Ring Topology	16
Tree Topology	17
Client-Server or Peer-to-Peer?	18
What's in the Mix?	19
Determining System Requirements	20
Planning and Implementing Security	21
Addressing External and Internal Threats	21
Formulating a Security Policy	22

	An Effective Password Policy	22
	General Security Rules	22
	Security Updates	23
	An Appropriate Firewall System	23
	Planning for Recovery from Disasters	23
	Clustering Solutions	24
	Disaster Recovery	25
	Writing It Down: Good Records Can Save Your Job	26
	Summary	28
Chapter 3	Standard Installation	29
	Exploring Your PC's Components	30
	Processor	30
	Bus	30
	Memory	31
	Video Card and Monitor	31
	Hard Drive	32
	Floppy Disk Drive	32
	Keyboard and Mouse	33
	SCSI Controller	33
	CD/DVD-R/RW Drive	33
	Sound Card	33
	Network Card	34
	Checking for Supported Hardware	34
	Creating the Red Hat Boot Disk	35
	Starting the Installation	36
	Partitioning the Hard Disk	42
	Using Disk Druid to Partition Your Disks	45
	Naming Disks and Devices	45
	Mounting a File System	46
	Understanding the Swap Partition	47
	Preparing Disk Partitions	47
	Setting Up the Partitions	49
	Configuring the Installation	51
	Installing the Boot Loader	51
	Configuring the Network	54
	Configuring the Firewall	56
	Choosing Additional Languages	58
	Setting the Time Zone	59
	Setting the Root Password	61
	Selecting the Package Groups to Install	62
	Running Firstboot	65
	Summary	70
Chapter 4	Kickstart Installation	71
	Using the Kickstart Configurator	71
	Installing the Kickstart Configurator	72

Boot Loader Options Screen	77
Partition Information Screen	78
Network Configuration	83
Authentication	84
Firewall Configuration	86
Display Configuration	87
Package Selection	90
Pre-Installation Script	91
Post-Installation Script	92
Starting the Kickstart Installation	93
Creating a Bootable Floppy	93
Creating a Bootable CD-ROM	94
Starting a Kickstart Installation	95
Summary	96
Chapter 5 Exploring the Desktops	97
Examining the Graphical Login Screen	97
Logging In and Using the GNOME Desktop	99
Playing with the Panel	101
Managing Applets on the Panel	102
Choosing Items from the Applications Menu in Fedora Core	103
Choosing Items from the Places Menu in Fedora Core	105
Choosing Items from the Desktop Menu in Fedora Core	106
Choosing Items from the Applications Menu on Enterprise Linux	107
Choosing Actions from the Actions Menu in Enterprise Linux	109
Using the Nautilus File Manager	110
Displaying Your Home Folder	112
Displaying the Contents of a Folder	112
Opening Files	112
Accessing FTP Sites	113
Using Bookmarks	113
Adding a Bookmark	113
Editing Bookmarks	113
Deleting Bookmarks	114
Managing Your Files and Folders	114
Customizing the Nautilus File Manager	115
Editing File Manager Preferences	115
Changing the File Manager Background and Icon Emblems	117
Showing and Hiding Views	118
Configuring GNOME	118
Logging Out	119
Taking a Look at KDE	119
Managing Applets	121
Choosing Applications from the Applications Menu	122
Using the Konqueror File Manager	124
Logging Out of KDE	126
Summary	126

Chapter 6	System Startup and Shutdown	127
	Examining the Boot Process	128
	The Boot Loader	128
	Using GRUB during Boot	130
	The Kernel	132
	The /sbin/init Program	133
	Exploring Runlevels	136
	Changing the System Runlevel	136
	Starting Programs at System Boot	137
	Shutting Down the System	138
	GRUB Configuration File	139
	Summary	140
Chapter 7	The File System Explained	141
	Understanding the File System Structure	141
	The / Directory	143
	Working with Linux-Supported File Systems	144
	ext3	145
	ext2	146
	reiserfs	146
	SystemV	147
	ufs	147
	FAT	147
	NTFS	147
	IBM JFS	147
	SGI XFS	148
	Nonstandard Linux File Systems	148
	FREEVxFS	148
	GFS	148
	Memory and Virtual File Systems	149
	cramfs	149
	tmpfs	149
	ramfs	150
	romfs	150
	proc	150
	Proc Software Information	150
	Proc Hardware Information	152
	/dev/pts	154
	devfs	154
	sysfs	155
	Linux Disk Management	155
	Disk Partitioning on an x86 Machine	155
	Mounting Other OS Partitions/Slices	155
	Metadevices	156
	Logical Volumes	156
	RAID	160
	Summary	161

Chapter 8	Examining the System Configuration Files	163
	Examining the System Configuration Files	164
	Systemwide Shell Configuration Scripts	164
	Shell Config Scripts: bashrc, csh.cshrc, zshrc	165
	bash, tcsh, zsh, and Their Config File Read Orders	167
	System Environmental Settings	168
	/etc/motd	168
	issue	168
	issue.net	168
	aliases	169
	fstab	169
	grub.conf	170
	cron files	171
	syslog.conf	172
	ld.so.conf	174
	logrotate.conf	174
	Examining the /etc/sysconfig/ Directory	175
	/etc/sysconfig/apmd	176
	/etc/sysconfig/authconfig	177
	/etc/sysconfig/clock	177
	/etc/sysconfig/crond	178
	/etc/sysconfig/desktop	178
	/etc/sysconfig/firstboot	178
	/etc/sysconfig/grub	178
	/etc/sysconfig/harddisks	178
	/etc/sysconfig/hwconf	179
	/etc/sysconfig/i18n	179
	/etc/sysconfig/init	179
	/etc/sysconfig/iptables	180
	/etc/sysconfig/irda	181
	/etc/sysconfig/kernel	181
	/etc/sysconfig/keyboard	181
	/etc/sysconfig/kudzu	182
	/etc/sysconfig/mouse	182
	/etc/sysconfig/named	183
	/etc/sysconfig/netdump	183
	/etc/sysconfig/network	184
	/etc/sysconfig/ntpd	184
	/etc/sysconfig/pcmcia	184
	/etc/sysconfig/selinux	185
	/etc/sysconfig/system-config-users	185
	/etc/sysconfig/system-logviewer	185
	/etc/sysconfig/samba	186
	/etc/sysconfig/sendmail	186
	/etc/sysconfig/vncservers	186
	/etc/sysconfig/xinetd	187

Directories in the /etc/sysconfig/ Directory	187
apm-scripts	187
daemons	187
networking	187
network-scripts	188
rhn	188
Examining the Network Configuration Files	188
Files to Change When Setting Up a System or Moving the System	188
Setting Up the IP Address	189
Setting Up the Hostname	190
Setting Up the DNS Name Resolution	190
Making a Local File of Hostname to IP Address Mappings	191
Setting Up Name Service Resolution Order	192
Starting Up Network Services from xinetd	193
Starting Up Network Services from the rc Scripts	194
Other Important Network Configuration Files	
in the /etc/sysconfig Directory	195
static-routes	195
Iptables	195
Network Configuration Files in /etc/sysconfig/network-scripts	196
ifcfg-networkinterfacename	196
ifup and ifdown	196
Managing the init Scripts	196
Managing rc Scripts by Hand	198
Managing rc Scripts Using chkconfig	200
Summary	202
Part Two Network Services	203
Chapter 9 Managing the X Window System	205
Configuring the X Server with the X Configuration Tool	205
Changing the Display Resolution	206
Changing the Display Color Depth	207
Changing Monitor Type Settings	207
Changing Your Video Card Type	208
Configuring Dual Monitors	209
Manually Configuring Your X Server	210
The X Server Configuration File	210
Summary	215
Chapter 10 Configuring Printers	217
Configuring Printers with the Printer Configuration Tool	217
Configuring the Print Queue	219
Selecting the Print Driver	224

Editing the Printer Configuration	225
Deleting a Printer	227
Setting the Default Printer	227
Managing Print Jobs	227
Summary	228
Chapter 11 TCP/IP Networking	229
TCP/IP Explained	229
Understanding Network Classes	231
Setting Up a Network Interface Card (NIC)	233
Configuring the Network Card	234
Configuring an Internal Network	235
Understanding Subnetting	238
Interpreting IP Numbers	240
Before You Subnet Your Network	241
Classless InterDomain Routing	244
Working with Gateways and Routers	246
Configuring Dynamic Host Configuration Protocol	247
Setting Up the Server	248
Configuring the DHCP Client	250
Configuring the Network Using the Network	
Configuration Tool	250
Adding an Ethernet Device	251
Adding a Wireless NIC	254
Adding a Modem Connection	256
Editing Your Network Configuration	259
Removing a NIC	259
Changing the NIC Configuration	260
Managing DNS Settings	261
Managing Hosts	261
Working with Profiles	262
Configuring IP Masquerading	263
Summary	263
Chapter 12 The Network File System	265
NFS Overview	265
Understanding NFS	266
What's New with NFSv4?	268
NFS Advantages and Disadvantages	269
Planning an NFS Installation	271
Configuring an NFS Server	273
NFS Server Configuration and Status Files	274
NFS Server Daemons	283
NFS Server Scripts and Commands	285
Using Secure NFS	290
Example NFS Server	290
Using the NFS Server Configuration Tool	292

Configuring an NFS Client	296
Configuring an NFSv4 Client	299
Example NFS Client	300
Using Automount Services	301
Examining NFS Security	305
General NFS Security Issues	305
Server Security Considerations	306
Client Security Considerations	307
Summary	308
Chapter 13 The Network Information System	309
Understanding NIS	309
Planning an NIS Installation	311
Configuring an NIS Server	315
Key Files and Commands	315
Starting the NIS Password Daemon	321
Starting the Server Transfer Daemon	321
Starting the NIS Servers at Boot Time	322
Configuring an Example NIS Server	324
Configuring an NIS Client	326
Setting the NIS Domain Name	326
Configuring and Starting the Client Daemon	326
Configuring the Client Startup Files	331
NIS Client Commands	331
Configuring an Example NIS Client	333
Using NIS and NFS Together	334
Summary	337
Chapter 14 Connecting to Microsoft and Novell Networks	339
Installing Samba	340
Configuring the Samba Server	341
[global]	342
[homes]	343
[printers]	344
Creating Samba Users	344
Starting the Samba Server	345
Connecting to a Samba Client	345
Connecting from a Windows PC	
to the Samba Server	347
Connecting to Novell Networks	348
Summary	350
Chapter 15 Configuring a Database Server	351
Linux Database Servers	351
Using MySQL	353
Securing the MySQL Installation	355
Using the MySQL Client Programs	359

Using PostgreSQL	362
Verifying the PostgreSQL Installation	365
Finalizing the PostgreSQL Installation	366
Initializing the Installation	366
Modifying Access Privileges	368
Creating a Test Database	372
Testing Connectivity to the Test Database	374
Using the PostgreSQL Client Programs	375
Summary	379
Chapter 16 Creating a VNC Server	381
What Is VNC?	381
Setting Up a VNC Server	383
Configuring Your Firewall for VNC	384
Customizing the VNC Server	386
Testing the VNC	388
Summary	392
Chapter 17 Providing Additional Network Services	393
Configuring a Time Server	394
Selecting a Time Server Solution	395
Configuring the Time Server	396
Selecting Reference Clocks	397
Configuring an NTP Client	401
Playing Nicely and Wisely with NTP	405
Providing a Caching Proxy Server	406
Verifying the Kernel Configuration	408
Configuring Squid	409
Modifying Netfilter	411
Starting Squid	412
Testing the Configuration	412
Summary	414
Chapter 18 Optimizing Network Services	415
Optimizing the X Window System	416
Optimizing NFS	418
Optimizing NIS	423
Optimizing Samba Networking	423
Getting More from a Database Server	424
Summary	425
Part Three Internet Services	427
Chapter 19 What Are Internet Services?	429
Learning about Secure Services	430
SSH	430
scp	431
sftp	433

Less Secure Services	434
Telnet	434
FTP	434
rsync	435
rsh	435
rlogin	435
finger	435
talk and ntalk	435
Using Your Linux Machine as a Server	436
HTTP	436
sshd	436
ftpd	436
DNS	437
Configuring the xinetd Server	437
Comparing xinetd and Standalone	439
xinetd-Started Services	439
Standalone Services	440
Configuring Linux Firewall Packages	441
Summary	441
Chapter 20 Configuring BIND: The Domain Name System	443
Understanding DNS	443
Installing the Software	446
Understanding Types of Domain Servers	447
Examining Server Configuration Files	449
The named.conf file	450
Options	451
Include	454
Acl	455
Logging	455
server	457
zones	457
Zone Files	458
SOA — Start of Authority	459
The Reverse Zone File	460
Configuring a Caching DNS Server	461
Configuring a Secondary Master DNS Server	462
Configuring a Primary Master Server	462
Checking Your Configuration	464
The Host Program	464
The dig Program	465
Summary	466
Chapter 21 Configuring Mail Services	467
Email Explained	467
Tracing the Email Delivery Process	468
Mail User Agent (MUA)	468

Mail Transfer Agent (MTA)	469
Mail Delivery Agent (MDA)	469
Introducing SMTP	470
Understanding POP3	471
Understanding IMAP4	471
Configuring Sendmail	472
Configuring Sendmail	474
The m4 Macro Processor	475
Understanding and Managing the Mail Queue	476
Setting Up Aliases to Make Life Easier	476
Using Other Sendmail Files and Commands	478
Using the Postfix Mail Server	479
Switching to Postfix	479
Configuring Postfix	480
Running Postfix behind a Firewall or Gateway	482
Running Postfix on a Mail Host	483
Serving Email with POP3 and IMAP	484
Setting up an IMAP Server	485
Configuring Dovecot	485
Testing Cyrus	486
Maintaining Email Security	486
Protecting against Eavesdropping	487
Using Encryption	487
Using a Firewall	487
Don't Get Bombed, Spammed, or Spoofed	488
Be Careful with SMTP	488
Summary	489
Chapter 22 Configuring FTP Services	491
Introducing vsftpd	492
Configuring vsftpd	493
Configuring User Level FTP Access	496
Configuring vsftpd Features	497
Disabling Anonymous FTP	501
Advanced FTP Server Configuration	502
Running vsftpd from xinetd	502
Enabling Anonymous Uploads	503
Enabling Guest User FTP Accounts	504
Running vsftpd over SSL	507
Using SFTP	509
Summary	510
Chapter 23 Configuring a Web Server	511
Introducing Apache	511
Apache Features	512
Changes in Apache 2	516
How Web Servers Work	517

Configuring Apache	519
Apache's Startup Process	520
Configuring Global Behavior	521
Configuring the Default Server	524
Configuring Virtual Servers	537
Starting and Stopping Apache	539
Implementing SSI	540
Enabling CGI	543
Enabling PHP	545
Creating a Secure Server with SSL	546
Understanding SSL and Server Certificates	547
Creating a Self-Signed Certificate	549
Obtaining a Certificate from a Certification Authority	554
Summary	554
Chapter 24 Providing Web Services	555
Creating Mailing Lists	555
Completing the Initial Mailman Configuration	556
Creating a Mailing List	559
Modifying a Mailing List's Configuration	560
Performing Common Mailman Administrative Tasks	561
Adding Multiple List Members	562
Hiding a Mailing List	562
Restricting Archives Access	563
Setting Up Web-Based Email	563
Connecting to SquirrelMail	563
Reconfiguring SquirrelMail	565
Configuring an RSS Feed	567
Selecting Content for an RSS Feed	570
Creating the Feed File	570
Turning on an RSS Feed	572
Adding Search Functionality	574
Getting Started with ht://Dig	574
Summary	579
Chapter 25 Optimizing Internet Services	581
Optimizing LDAP Services	582
Optimizing DNS Services	583
Improving the Performance of DNS Clients	583
Tweaking DNS Servers	585
Logging	586
Optimizing Mail Services	587
Getting More from Sendmail	587
Getting More from Postfix	588
Optimizing FTP Services	590
Optimizing Web Services	590
Summary	593

Part Four	System Administration	595
Chapter 26	Keeping Your System Updated with up2date and the Red Hat Network	597
	Using the Red Hat up2date Agent	598
	Configuring the up2date Agent	599
	Updating Your System	602
	Registering Your System	605
	Accessing the Red Hat Network with a Web Browser	608
	Summary	614
Chapter 27	Upgrading and Customizing the Kernel	615
	Determining Whether to Upgrade to a New Kernel	616
	Upgrading versus Customizing	618
	Preparing to Upgrade	618
	Installing a Kernel RPM	619
	Getting the Kernel Source	620
	Using the Kernel Source RPM	621
	Using Pristine Kernel Source	623
	Verifying and Unpacking the Archive	626
	Patching the Kernel	627
	Configuring the Kernel	629
	Selecting a Kernel Configuration File	630
	Configuring the Kernel with xconfig	633
	Configuring the Kernel with menuconfig	634
	Reviewing the Configuration Options	637
	Code Maturity Level Options	637
	General Setup	637
	Loadable Module Support	640
	Processor Type and Features	640
	Power Management Options	643
	Bus Options	643
	Executable File Formats	644
	Device Drivers	645
	Generic Driver Options	645
	Memory Technology Devices	645
	Parallel Port Support	645
	Plug and Play Support	646
	Block Devices	646
	ATA / ATAPI / MFM / RLL Support	647
	SCSI Device Support	648
	Old CD-ROM Drivers	648
	Multidevice Support	649
	Fusion MPT Device Support	649
	IEEE 1394 / FireWire Support	649
	I2O Device Support	649
	Networking Support	649
	ISDN and Telephony	653

Input Device Support	653
Character Devices	654
I2C Support	654
Multimedia Devices	655
Graphics Support	655
Sound	656
USB Support	656
MMC/SD Card Support	660
InfiniBand Support	660
File Systems	660
CD-ROM/DVD File Systems	661
DOS/FAT/NT File Systems	661
Pseudo-File-Systems	662
Miscellaneous File Systems	662
Network File Systems	662
Partition Types	662
Native Language Support	663
Profiling Support	663
Kernel Hacking	664
Security Options	664
Cryptography Options	664
Library Routines	665
Saving the Kernel Configuration	665
Compiling the Kernel	666
Installing the Kernel	669
Updating GRUB	670
Summary	671
Chapter 28 Configuring the System at the Command Line	673
Administrating Your System from the Command Line	673
Managing Processes	675
Obtaining Process Information	676
Signaling Processes	680
Modifying Process Priorities	682
Maintaining the File System	683
Working with File Systems	683
Creating and Manipulating Partitions	683
Creating and Manipulating File Systems	685
Working with Files and Directories	691
Managing Disk Space Usage	695
Timekeeping	696
Single-Use Commands	697
Using the Network Time Protocol	702
Automating Scripts	702
Running One-Shot Jobs with at	702
Running Regularly Scheduled Jobs with cron	704
Summary	705

Chapter 29 Administering Users and Groups	707
Administering User Accounts	707
Working with User Accounts	708
The User Database Files	708
Modifying Multiple Accounts Simultaneously	715
Viewing Login and Process Information	717
Working with Group Accounts	718
Creating Groups	719
Modifying and Deleting Groups	720
Using a Shadowed Group File	722
Using User Private Groups	723
Administering Users and Groups with User Manager	725
Creating User Accounts	726
Modifying and Deleting User Accounts	727
Creating Group Accounts	728
Modifying and Deleting Group Accounts	729
Understanding the Root Account	730
Implementing Sudo	731
Deciphering Sudo's Configuration File	733
Sudo Configuration and Usage Tips	737
Using File System Quotas	737
Enabling Quotas	738
Creating the Quota Files	739
Turning on Quotas	740
Setting and Modifying Quotas	740
Viewing Quota Utilization	742
Summary	744
Chapter 30 Installing and Upgrading Software Packages	745
Using the Red Hat Package Manager	745
General Options	746
Query Mode	748
Querying Package Dependencies	750
What's in That RPM?	751
Formatting Query Output	754
Package Installation and Removal	755
Installing RPMs	756
Upgrading RPMs	757
Removing RPMs	758
Verifying RPMs	758
Building Packages Using Source RPMs	761
Checking Software Versions	764
Obtaining Newer Software	767
Using Third-Party Sites to Find RPMs	768
Using Ibiblio.org	770

Installing Software from Source	771
Configuring the Build Environment	772
Unpacking the Source Code	772
Configuring the Source Code	773
Building the Software Package	775
Testing the Build	776
Installing the Software	777
Summary	778
Chapter 31 Backing Up and Restoring the File System	779
Creating a Backup Plan	779
Choosing Media for Backups	781
Understanding Backup Methods	781
Tape Rotation	783
Using Backup Tools	784
Command Line Tools	784
Using mt-st	784
Using the cdrecord Package	787
Using dump	789
Using restore	790
Using tar	793
Advanced Tools	795
Using AMANDA	795
Summary	804
Chapter 32 Performance Monitoring	805
System-Performance-Monitoring Tools	805
Measuring Memory Usage	806
Memory Usage as Seen by Users and Processes	806
Examining Kernel Memory Usage	810
Viewing Running Tasks	812
Getting Started with ps	813
Using top	817
Monitoring I/O Activity	822
Using sar	826
Monitoring Memory with sar	827
Monitoring CPU Usage with sar	829
Summary	831
Part Five System Security and Problem Solving	833
Chapter 33 Exploring SELinux Security	835
Understanding SELinux	835
Mandatory and Role-Based Access Control	836
SELinux Policies	838
Using SELinux	838
Enabling SELinux Manually	842
Modifying the Targeted Policy	843

Finding More Information about SELinux	845
Summary	846
Chapter 34 Implementing Network Security	847
Creating a Firewall	847
Installing, Configuring, and Using LDAP	851
Overview of LDAP Directory Organization	852
OpenLDAP Packages for Linux	855
Core OpenLDAP Server Files, Daemons, and Utilities	856
Configuring and Starting an OpenLDAP Server	857
Using OpenLDAP for System Authentication	860
Adding User, Password, and Group	
Entries to an LDAP Server	860
Updating Client Systems to Use LDAP Authentication	861
Installing, Configuring, and Using Kerberos	864
Kerberos Terminology, Machine Roles, and Reliability	865
Kerberos Packages for Linux	865
Core Kerberos Utilities	866
Installing and Configuring a Kerberos Server	867
Enabling Kerberos Clients and Applications	870
Using Kerberos for Login Authentication	871
Summary	874
Chapter 35 Troubleshooting and Problem Solving	875
Troubleshooting Techniques	876
Step 1: Identify the Problem	876
Step 2: Reproduce the Problem	876
Step 3: Look for Changes	877
Step 4: Determine the Most Likely Cause	877
Step 5: Implement a Solution	878
Step 6: Keep Documentation	878
Troubleshooting Resources	878
The Internet	878
System Log Files	879
README Files	882
Solving Common Problems	883
Unable to Log In	883
Resetting a User's Password	883
Creating a User Account	883
Lost or Forgotten Root Password	884
CD-ROM Drive Not Detected during Installation	884
CD-ROM Drive Does Not Mount after Installation	885
Sound Does Not Work after Installation	885
Unable to Unmount a Drive	887
Shell Commands Don't Work	888
Solving File System Problems	888
Cannot Delete a File	889
Commands with Multiword Arguments	889

Accessing Windows File Systems	890
Working with Floppy Disks	890
Cannot Mount a Partition	891
Avoiding File System Checks at Each System Reboot	891
Solving Networking Problems	891
Getting Online with a Modem	893
The Boot Process Hangs	895
Using Two Ethernet Cards	896
Solving NFS Problems	896
Exploring Miscellaneous Problems	898
Solving Boot Problems	899
ht://Dig Won't Run	900
Starting cyrus-imapd	900
Solving Laptop Video Problems	901
The Signal 7 and Signal 11 Problems	902
Using Screensavers and Power Management	903
Starting the X Window System	903
Making an Emergency Boot Disk	904
Summary	904
Appendix A Bash Shell Scripting	905
Using Wildcards and Special Characters	906
Using Variables	909
Using Bash Operators	913
Comparison Operators	913
Arithmetic Operators	916
File Test Operators	917
Understanding Flow Control	919
Conditional Execution Using if Statements	920
Determinate Loops Using the for Statement	922
Indeterminate Loops Using while and until Statements	923
Selection Structures Using case and select Statements	924
The case Statement	925
The select Statement	926
Using Shell Functions	928
Processing Input and Output	929
Redirecting I/O	929
String I/O	932
Working with Command Line Arguments	934
Using Processes and Job Control	936
Summary	941
Index	943

PART

One

System and Network Administration Defined

- Chapter 1:** Duties of the System Administrator
- Chapter 2:** Planning the Network
- Chapter 3:** Standard Installation
- Chapter 4:** Kickstart Installation
- Chapter 5:** Exploring the Desktops
- Chapter 6:** System Startup and Shutdown
- Chapter 7:** The File System Explained
- Chapter 8:** Examining the System Configuration Files

Duties of the System Administrator

IN THIS CHAPTER

- The Linux System Administrator
- Installing and Configuring Servers
- Installing and Configuring Application Software
- Creating and Maintaining User Accounts
- Backing Up and Restoring Files
- Monitoring and Tuning Performance
- Configuring a Secure System
- Using Tools to Monitor Security

Linux is a multiuser, multitasking operating system from the ground up. In this regard the system administrator has flexibility — and responsibility — far beyond those of other operating systems. Red Hat has employed innovations that extend these duties even for the experienced Linux user. This chapter briefly looks at those responsibilities, which are covered in more detail in later chapters.

The Linux System Administrator

Using Linux involves much more than merely sitting down and turning on the machine. Often you hear talk of a “steep learning curve” but that discouraging phrase can be misleading. Linux is quite different from the most popular commercial operating systems in a number of ways. While it is no more difficult to learn than other operating systems are, it is likely to seem very strange even to the experienced administrator of other systems. In addition, the sophistication of a number of parts of the Red Hat distribution has increased by an order of

magnitude, so even an experienced Linux administrator is likely to find much that is new and unfamiliar. Fortunately, there are new tools designed to make system administration easier than ever before.

Make no mistake: Every computer in the world has a system administrator. It may be — and probably is — true that the majority of system administrators are those who decided what software and peripherals were bundled with the machine when it was shipped. That status quo remains because the majority of users who acquire computers for use as appliances probably do little to change the default values. But the minute a user decides on a different wallpaper image or adds an application that was acquired apart from the machine itself, he or she has taken on the role of system administration.

The highfalutin' title of system administrator brings with it some responsibilities. No one whose computer is connected to the Internet, for instance, has been immune to the effects of poorly administered systems, as demonstrated by the distributed denial of service (DDoS) and email macro virus attacks that have shaken the online world in recent years. The scope of these acts of computer vandalism (in some cases, computer larceny) would have been greatly reduced if system administrators had a better understanding of their duties.

Linux system administrators are likely to understand the necessity of active system administration more than those who run whatever came on the computer, assuming that things came properly configured from the factory. The user or enterprise that decides on Linux has decided, also, to assume the control that Linux offers, and the responsibilities that this entails.

By its very nature as a modern, multiuser operating system, Linux requires a degree of administration greater than that of less robust, home-market systems. This means that even if you use just a single machine connected to the Internet by a dial-up modem — or not even connected at all — you have the benefits of the same system employed by some of the largest businesses in the world, and will do many of the same things that IT professionals employed by those companies are paid to do. Administering your system does involve a degree of learning, but it also means that in setting up and configuring your own system you gain skills and understanding that raise you above mere “computer user” status. The Linux system administrator does not achieve that mantle by purchasing a computer but by taking full control of what the computer does and how it does it.

You may end up configuring a small home or small office network of two or more machines, perhaps including ones that are not running Linux. You may be responsible for a business network of dozens of machines. The nature of system administration in Linux is surprisingly constant, no matter how large or small your installation. It merely involves enabling and configuring features you already have available.

By definition, the Linux system administrator is the person who has “root” access, which is to say the one who is the system’s “superuser” (or root user). A standard Linux user is limited to whatever he or she can do with the underlying

engine of the system. But the root user has unfettered access to everything — all user accounts, their home directories, and the files therein; all system configurations; and all files on the system. A certain body of thought says that no one should ever log in as “root,” because system administration tasks can be performed more easily and safely through other, more specific means, which we discuss in due course. Because the system administrator has full system privileges, your first duty is to know what you’re doing, lest you break something.

NOTE By definition, the Linux system administrator can be anyone who has “root” access — anyone who has root access is the system’s “superuser.”

The word *duty* implies a degree of drudgery; in fact, it’s a manifestation of the tremendous flexibility of the system measured against the responsibility to run a tight organization. These duties do not so much constrain you, the system administrator, as free you to match the job to the task. Let’s take a brief look at them.

Installing and Configuring Servers

When you hear the word *server* to describe a computer, you probably think of a computer that offers some type of service to clients. The server may provide file or printer sharing, File Transfer Protocol (FTP) or Web access, or email-processing tasks. Don’t think of a server as a standalone workstation; think of it as a computer that specifically performs these services for many users.

In the Linux world, the word *server* has a broader meaning than what you might be used to. For instance, the standard Red Hat graphical user interface (GUI) requires a graphical layer called XFree86. This is a server. It runs even on a standalone machine with one user account. It must be configured. (Fortunately, Red Hat has made this a simple and painless part of installation on all but the most obscure combinations of video card and monitor; gone are the days of anguish as you configure a graphical desktop.)

Likewise, printing in Linux takes place only after you configure a print server. Again, this has become so easy as to be nearly trivial.

In certain areas the client-server nomenclature can be confusing, though. While you cannot have a graphical desktop without an X server, you can have remote Web access without running a local Web server, remote FTP access without running a local FTP server, and email capabilities without ever starting a local mail server. You may well want to use these servers, all of which are included in Red Hat; then again, maybe not. Whenever a server is connected to other machines outside your physical control, there are security implications to consider. You want your users to have easy access to the things they need, but you don’t want to open up the system you’re administering to the whole wide world.

NOTE Whenever a server is connected to machines outside your physical control, security issues arise. You want users to have easy access to the things they need but you don't want to open up the system you're administering to the whole wide world.

Linux distributions used to ship with all imaginable servers turned on by default. Just installing the operating system on the computer would install and configure — with default parameters — all the services available with the distribution. This was a reflection of an earlier, more innocent era in computing when people did not consider vandalizing other people's machines to be good sportsmanship. Unfortunately, the realities of this modern, more dangerous world dictate that all but the most essential servers remain turned off unless specifically enabled and configured. This duty falls to the system administrator. You need to know exactly which servers you need and how to employ them, and to be aware that it is bad practice and a potential security nightmare to enable services that the system isn't using and doesn't need. Fortunately, the following pages show you how to carry out this aspect of system administration easily and efficiently.

Installing and Configuring Application Software

Although it is possible for individual users to install some applications in their home directories — drive space set aside for their own files and customizations — these applications may not be available to other users without the intervention of the user who installed the program or the system administrator. Besides, if an application is to be used by more than one user, it probably needs to be installed higher up in the Linux file hierarchy, which is a job that only the system administrator can perform. (The administrator can even decide which users may use which applications by creating a “group” for that application and enrolling individual users in that group.)

New software packages might be installed in `/opt` if they are likely to be upgraded separately from the Red Hat distribution itself. Doing this makes it simple to retain the old version until you are certain that the new version works and meets your expectations. Some packages may need to go in `/usr/src` or even `/usr` if they are upgrades of packages installed as part of Red Hat. (For instance, there are sometimes security upgrades of existing packages.) The location of the installation usually matters only if you compile the application from source code; if you use a Red Hat Package Manager (RPM) application package, it automatically goes where it should.

Configuration and customization of applications is to some extent at the user's discretion, but not entirely. “Skeleton” configurations — administrator-determined default configurations — set the baseline for user employment of

applications. If there are particular forms, for example, that are used throughout an enterprise, the system administrator would set them up or at least make them available by adding them to the skeleton configuration. The same applies to configuring user desktops and in even deciding what applications should appear on user desktop menus. For instance, your company may not want to grant users access to the games that ship with modern Linux desktops. You may also want to add menu items for newly installed or custom applications. The system administrator brings all this to pass.

Creating and Maintaining User Accounts

Not just anyone can show up and log on to a Linux machine. An account must be created for each user and — you guessed it — no one but the system administrator can do this. That's simple enough.

But there's more. It involves decisions that either you or your company must make. You might want to let users select their own passwords, which would no doubt make them easier to remember but which probably would be easier for a malefactor to crack. You might want to assign passwords, which is more secure in theory but increases the likelihood that users will write them down on a conveniently located scrap of paper — a risk if many people have access to the area where the machine(s) is located. You might decide that users must change their passwords periodically — something you can configure Red Hat Enterprise Linux to prompt users about.

What happens to old accounts? Suppose that someone leaves the company. You probably don't want that person to gain access to the company's network, but you also don't want to delete the account wholesale, only to discover later that essential data resided nowhere else.

To what may specific users have access? It might be that there are aspects of your business that make Web access desirable, but you don't want everyone spending their working hours surfing the Web. If your system is at home, you may wish to limit your children's access to certain Web sites.

These and other issues are part of the system administrator's duties in managing user accounts. Whether the administrator or his or her employer establishes policies governing accounts, these policies should be delineated — preferably in writing for a company — for the protection of all concerned.

Backing Up and Restoring Files

Until computer equipment becomes infallible, until people lose the desire to harm others' property, and — truth be told — until system administrators become perfect, there is considerable need to back up important files so that

the system can be up and running again with minimal disruption in the event of hardware, security, or administration failure. Only the system administrator may do this. (Because of its built-in security features, Linux doesn't allow even users to back up their own files to removable disks.)

It's not enough to know that performing backups is your job. You need to formulate a strategy for making sure your system is not vulnerable to catastrophic disruption. This is not always obvious. If you have a high-capacity tape drive and several good sets of restore disks, you might make a full system backup every few days. If you are managing a system with scores of users, you might find it more sensible to back up user accounts and system configuration files, figuring that reinstallation from the distribution CDs would be quicker and easier than getting the basics off a tape archive. (Don't forget about applications you install separately from your Red Hat distribution, especially those involving heavy customization.)

Once you decide *what* to back up, you need to decide *how frequently* to perform backups, whether to maintain a series of incremental backups — adding only files that have changed since the last backup — or multiple full backups, and *when* these backups should be performed. Do you trust an automated, unattended process? If you help determine which equipment to use, do you go with a redundant array of independent disks (RAID), which is to say multiple hard drives all containing the same data as insurance against the failure of any one of them, in addition to other backup systems? (A RAID is not enough because hard drive failure is not the only means by which a system can be brought to a halt.)

You don't want to become complacent or foster a lackadaisical attitude among users. Part of your strategy should be to maintain perfect backups without ever needing to resort to them. This means encouraging users to keep multiple copies of their important files in their home directories so that you won't be asked to mount a backup to restore a file that a user corrupted. (If your system is a standalone one then, as your own system administrator, you should make a habit of backing up your configuration and other important files.)

Restoring files from your backup media is no less important than backing them up in the first place. Be certain you can restore your files if the need arises by testing your restore process at least once during a noncritical time. Periodically testing your backup media is also a good idea.

Chances are good that even if you work for a company, you'll be the one making these decisions. Your boss just wants a system that runs perfectly, all the time. Backing up is only part of the story, however. You need to formulate a plan for bringing the system back up after a failure. A system failure could be caused by any number of problems, either related to hardware or software (application, system configuration) trouble, and could range from a minor inconvenience to complete shutdown.

Hardware failures caused by improper configuration can be corrected by properly configuring the device. Sometimes hardware failures are caused by the device itself, which typically requires replacing the device. Software failures caused by improperly configured system files are usually corrected by properly configuring those files. An application can cause the system to fail for many reasons, and finding the root of the problem may require a lot of research on the part of the administrator.

If you are the administrator of servers and workstations for a business, you should have a disaster recovery plan in place. Such a plan takes into account the type of data and services provided and how much *fault tolerance* your systems require — that is, how long your systems could be down and what effect that would have on your company's ability to conduct business. If you require 100 percent fault tolerance, meaning your systems must be online 24/7, disaster recovery may be unnecessary in some circumstances as your systems never go down and there is no disaster from which to recover. Most organizations, though, cannot afford such a high level of fault tolerance; they are willing to accept less stringent standards. Based on the level of fault tolerance you require, your disaster recovery plan should list as many possible failures as you can anticipate and detail the steps required to restore your systems. In Chapter 2 we describe fault tolerance and disaster recovery in more detail.

TIP Backing up is only part of the story. You need to formulate a disaster recovery plan to bring your system back up in the event of a failure.

Monitoring and Tuning Performance

The default installation of Red Hat Enterprise Linux goes a long way toward capitalizing on existing system resources. There is no “one size fits all” configuration, however. Linux is infinitely configurable, or close to it.

On a modern standalone system, Linux runs pretty quickly. If it doesn't, there's something wrong — something the system administrator can fix. Still, you might want to squeeze one last little bit of performance out of your hardware — or a number of people might be using the same file server, mail server, or other shared machine, in which case seemingly small improvements in system performance add up.

System tuning is an ongoing process aided by a variety of diagnostic and monitoring tools. Some performance decisions are made at installation time, while others are added or tweaked later. A good example is the use of the `hdparm` utility, which can increase throughput in IDE drives considerably, but for some high-speed modes a check of system logs shows that faulty or inexpensive cables can, in combination with `hdparm`, produce an enormity of non-destructive but system-slowing errors.

Proper monitoring allows you to detect a misbehaving application that consumes more resources than it should or fails to exit completely upon closing. Through the use of system performance tools, you can determine when hardware — such as memory, added storage, or even something as elaborate as a hardware RAID — should be upgraded for more cost-effective use of a machine in the enterprise or for complicated computational tasks such as three-dimensional rendering.

Possibly most important, careful system monitoring and diagnostic practices give you a heads-up when a system component is showing early signs of failure, so that you can minimize any potential downtime. Combined with the resources for determining which components are best supported by Fedora Core and Red Hat Enterprise Linux, performance monitoring can result in replacement components that are far more robust and efficient in some cases.

In any case, careful system monitoring plus wise use of the built-in configurability of Linux allows you to squeeze the best possible performance from your existing equipment, from customizing video drivers to applying special kernel patches or simply turning off unneeded services to free memory and processor cycles.

TIP To squeeze the best performance from your equipment, monitor your system carefully and use Linux's built-in configurability wisely.

Configuring a Secure System

If there is a common thread in Linux system administration, it is the security of the computer and data integrity.

What does this mean? Just about everything. The system administrator's task, first and foremost, is to make certain that no data on the machine or network is likely to become corrupted, whether by hardware or power failure, misconfiguration or user error (to the extent that the latter can be avoided), or malicious or inadvertent intrusion from elsewhere. This means doing all the tasks described throughout this chapter, and doing them well, with a full understanding of their implications.

No one involved in computing has failed to hear of the succession of increasingly serious attacks on machines connected to the Internet. For the most part, these attacks have not targeted Linux systems. That doesn't mean Linux systems have been entirely immune, either to direct attack or to the effects of attacks on machines running other operating systems. In one distributed denial of service (DDoS) attack aimed at several major online companies,

for instance, many “zombie” machines — those that had been exploited so that the vandals could employ thousands of machines instead of just a few — were running Linux that had not been patched to guard against a well-known security flaw. In the various Code Red attacks during the summer of 2001, Linux machines themselves were invulnerable, but the huge amount of traffic generated by this worm infection nevertheless prevented many Linux machines from accomplishing much Web-based work for several weeks, so fierce was the storm raging across the Internet. And few email users have been immune from receiving at least some SirCam messages — nonsensical messages from strangers with randomly selected files attached from their machines. While this infection did not corrupt Linux machines per se, as it did those running MS Windows, anyone on a dial-up Internet connection who had to endure downloading several megabytes of infected mail each day would scarcely describe himself or herself as unaffected by the attack.

Depending on how a Linux machine is connected, and to what; the sensitivity of the data it contains; and the uses to which it is put, security can be as simple as turning off unneeded services, monitoring the Red Hat security mailing list to make sure that all security advisories are followed, regularly using system utilities to keep the system up to date, and otherwise engaging in good computing practices to make sure that the system runs robustly. It’s almost a full-time job, involving levels of security permissions within the system and systems to which it is connected; elaborate firewalls to protect not just Linux machines but machines that, through their use of non-Linux software, are far more vulnerable; and physical security — making sure that no one steals the machine itself!

For any machine connected to another machine, security means hardening against attacks and making certain that no one else uses your machine as a platform for launching attacks against others. If you run Web, FTP, or mail servers, it means giving access to only those who are entitled to it, while locking out everyone else. It means making sure that passwords are not easily guessed and not made available to unauthorized persons. It means that disgruntled former employees no longer have access to the system and that no unauthorized person may copy files from your machines.

Security is an ongoing process. The only really secure computer is one that contains no data, is unplugged from networks and power supplies, has no keyboard attached, and resides in a locked vault. While this is theoretically true, it implies that security diminishes the usefulness of the machine.

In the chapters that follow, you learn about the many tools that Red Hat provides to help you guard against intrusion, even to help you prevent intrusion into non-Linux machines that may reside on your network. Linux is designed from the beginning with security in mind. In all your tasks you should maintain that same security awareness.

TIP Your job as system administrator is to strike the right balance between maximum utility and maximum safety, all the while bearing in mind that confidence in a secure machine today means nothing about the machine's security tomorrow.

Using Tools to Monitor Security

People who, for purposes of larceny or to amuse themselves, like to break into computers — they're called *crackers* — are a clever bunch. If there is a vulnerability in a system, they will find it. Fortunately, the Linux development community is quick to find potential exploits and to create ways of slamming the door shut before crackers can enter. Fortunately, too, Red Hat is diligent in making available new, patched versions of packages in which potential exploits have been found. Your first and best security tool, therefore, is making sure that whenever a security advisory is issued, you download and install the repaired package. This line of defense can be annoying but it is nothing compared to rebuilding a compromised system.

As good as the bug trackers are, sometimes their job is reactive. Preventing the use of your machine for nefarious purposes and guarding against intrusion are, in the end, your responsibility alone. Red Hat equips you with tools to detect and deal with unauthorized access of many kinds. As this book unfolds, you'll learn how to install and configure these tools and how to make sense of the warnings they provide. Pay careful attention to those sections and do what they say. If your machine is connected to the Internet, you will be amazed at the number of attempts made to break into your machine. You'll be struck by how critical the issue of security is.

Summary

As you, the system administrator, read this book, bear in mind that your tasks are ongoing and that there is never a machine that is completely tuned, entirely up to date, and utterly secure for very long. The pace of Linux development is quite rapid, so it's important to keep current in the latest breakthroughs. This book gives you the very best information as to the Red Hat distribution you're using and tells you all you need to know about getting the most from it. Even more than that, you should read it with an eye toward developing a Linux system administrator's point of view, an understanding of how the system works as opposed to the mere performance of tasks. As the best system administrators will tell you, system administration is a state of mind.

Planning the Network

IN THIS CHAPTER

- Deciding How Your Network Will Be Used
- Planning and Implementing Security
- Planning for Recovery from Disasters
- Writing It Down: Good Records Can Save Your Job

While you can set up a Fedora Core or Red Hat Enterprise Linux network on the fly, your time will be spent most efficiently if you plan your network. Preparation reduces confusion not just now but also in the future, makes provision for expansion later on, and ensures that you make the best use of your resources, both budgetary and system-related. Although setting up a huge network of hundreds of nodes requires planning beyond the scope of this chapter, we explore here the fundamentals of planning and preparing for your new network installation.

Deciding How Your Network Will Be Used

By definition and right out of the box, Linux is a network operating system. It is also nearly infinitely configurable: You can tailor a network to meet your precise needs. That is a tremendous strength but it can also be daunting when compared with systems that are more limited. As the philosopher James Burnham said, “Where there is no alternative, there is no problem.”

Before you install Fedora Core or Red Hat Enterprise Linux on anything other than a standalone box just to take a look at it, you would be well advised to consider what kind of network you want to install, what it will be used for, what kinds of connections to the outside world it will have, and whether it is something you’re likely to expand later.

Questions to ask include the following:

- What services do you wish to provide within your local network?
- Will your network be made up entirely of Linux machines or will boxes running other operating systems be connected to it?
- What devices (printers, scanners, and DSL, cable modem, or T-1 connections) do you plan to share?
- Do you intend to host a Web site or an FTP site?
- What security implications do you foresee?
- How many machines will make up your network?

It makes sense now to take notes and answer these questions. You can find details about setting up your network elsewhere in this book. But careful planning now lets you chart a clear path to developing a quick and efficient network and, perhaps even more important, helps you make sure that your network is secure from both internal and external mischief.

CROSS-REFERENCE To learn more about setting up your network, see Chapter 11.

For example, many people who now enjoy DSL or cable Internet service wish to set up small networks purely to allow the sharing of that broadband connection. Having a permanent Internet connection demands that you pay more attention to security, which means making sure that you don't accidentally run any easily exploited services. If the network includes easily exploited operating systems, security becomes even more of a concern. Perhaps you will decide to set up a firewall on your Linux machine (or even set up a Linux box solely for firewall purposes). Or you might decide to employ one of the firewall-gateway-router network appliances that are gaining popularity and simply attach a hub to the appliance and attach each machine on the "network" to that hub. Such a network may not be big, but it may be all you need or want.

TIP A good rule of thumb is to provide the services for your network – and *only* those it needs.

Chances are good that you want to do more. Even if your needs are modest at first, adding services is simple in Red Hat Linux. Some features, such as printer sharing, you'll probably set up at the beginning.

Before you do anything else, decide the physical layout, or *topology*, of your network — how machines are connected — and whether you want a peer-to-peer or client-server network. These details matter because on the one hand you can overbuild your network so that your equipment isn't used efficiently;

on the other hand you can underestimate the demands on the network and end up slowing down one or more machines to near uselessness.

Understanding Topologies

Your network will probably be one of the first two of the following four commonly used topologies (at least for starters): star, bus, ring, and tree.

Star Topology

Think of this system as resembling a power strip with various devices plugged into it. In this case, instead of a power strip you have a network hub, and instead of devices requiring electricity you have devices needing and providing data. These devices might include computers, network-equipped printers, cable or DSL modems, a local network backbone, or even other hubs. Star topology networks are connected by twisted pair cabling, which looks like the cabling used in modular phone systems. Twisted pair cables and other devices are rated according to category (typically just called cat): Cat 3 uses two pairs of twisted wires as the standard for regular telephone service. Star topology networks usually use cat 5 twisted pair cabling, which has four twisted pairs and terminates in connectors called RJ-45s. (Your phone is connected with RJ-11s.) You may have up to 1024 nodes — distinct machines or devices — on a star topology network, at speeds of up to 100 MB per second. The newest networking technology provides even faster speeds. Figure 2-1 shows an example of a star topology network.

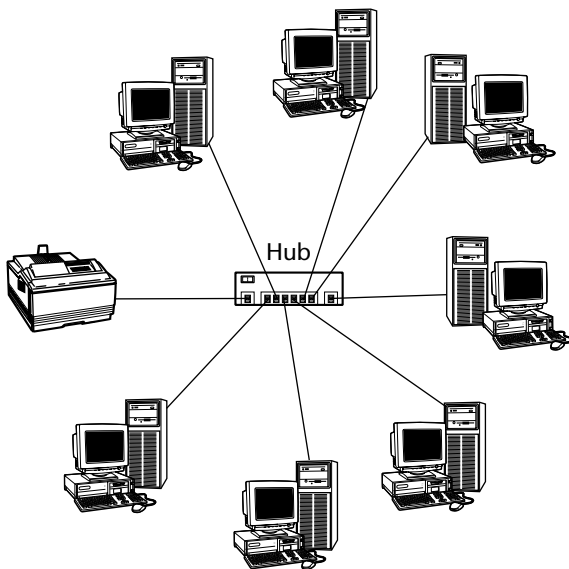


Figure 2-1 A typical star topology network.

Bus Topology

If star topology resembles a power strip with many devices plugged into it, bus topology physically resembles strings of old-fashioned Christmas tree lights, hooked together one after another. Of course, on your network there's a lot more going on than what happens on a string of lights. On a bus topology network one machine is plugged to the next, which is plugged to the next, and so on. Two types of coaxial cable hold bus topology networks together: RG-8, often called Thicknet because of the thickness of the cable, and RG-58, often called Thinnet because it is thinner than RG-8. RG-8 is familiar at least in general form to anyone involved in amateur radio or anyone who has ever hooked up cable television. With this kind of topology, each end of the cable is specifically terminated by use of a "terminating resistor."

Bus topology networks are limited to 30 machines. They were a very common style in early networks. While considered highly reliable, bus topology networks are not very fault tolerant because the failure of any device on the cable causes the entire network to fail. Also, their potential bandwidth (data-handling capacity) is limited to 10 MB per second. Nearly all modern networks use some type of star topology with cat 5 or better cabling. Figure 2-2 shows a typical bus topology network.

Ring Topology

Imagine those Christmas tree lights again. This time the end of the string plugs into its beginning, creating a loop. Popularized by IBM's Token Ring system, ring networks are relatively difficult to set up but do offer high-bandwidth capabilities. Figure 2-3 shows a typical ring topology network.

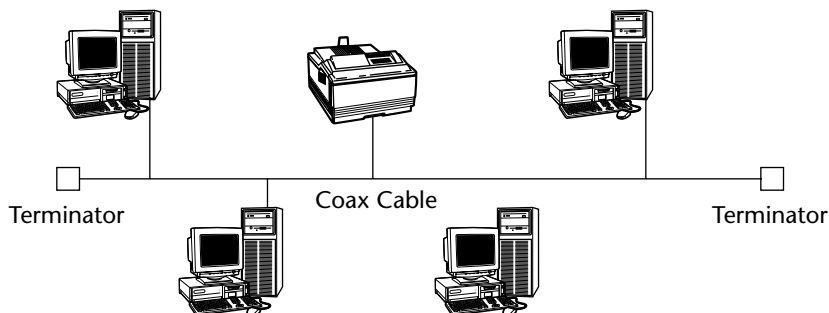


Figure 2-2 A typical bus topology network.

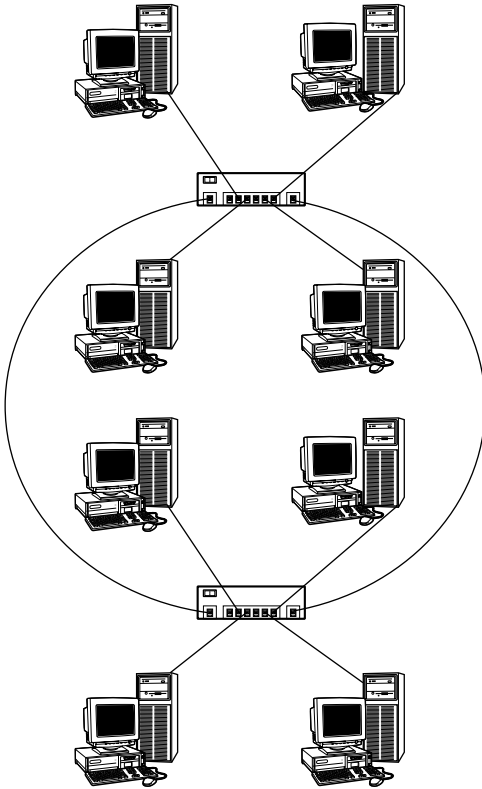


Figure 2-3 A typical ring topology network.

Tree Topology

Although you almost certainly won't undertake this system at the outset, you should know about it anyway. A tree network involves a high-speed "backbone" that is connected in the fashion of bus topology. However, instead of connecting individual machines, it connects groups of star topology subnetworks. Many network backbones use fiber-optic cabling to achieve high throughput. Figure 2-4 shows a typical tree topology.

Ultimately, your choice of network is determined by the equipment you already own and the amount of money you have to spend. If you are setting up a new network, speed, ease of configuration, and relatively low cost all argue in favor of establishing a star topology network.

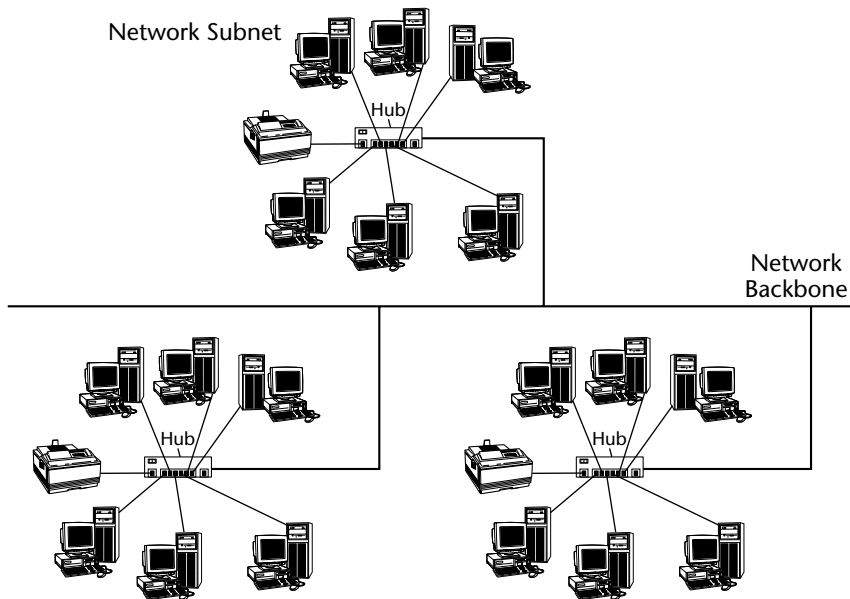


Figure 2-4 A typical tree topology network.

Client-Server or Peer-to-Peer?

In a *client-server* network, machines are dedicated to performing a variety of functions, in some ways like the old mainframe/dumb terminal days. You might, for instance, have a print server that handles print jobs for every computer on the network — a highly useful arrangement if, for example, yours is an enterprise that prepares many invoices, contracts, or other documents. Or you might have a file server, whose sole purpose is to serve up “boilerplate” documents or the contents of a huge database, or to serve as the repository of a big project on which many people collaborate. If your enterprise has an online presence, you may wish to dedicate one machine (or more) as a Web server, and perhaps one or more machines as a File Transfer Protocol (FTP) server, so that people can download (and upload) files. You’ll probably need some kind of mail server to handle both external email and messages sent within the network. Clients are machines connected to such a network. They are not servers; instead, they rely on network services provided by the server machines. Clients are usually full, freestanding workstations, although it is possible to connect dumb terminals — monitor, keyboard, pointing device — to such a network in some circumstances. To use the services provided by the server(s), clients need to have accounts on the desired server(s) and must log in to those accounts.

A *peer-to-peer* network resembles a client-server network in that the machines are wired to each other and some services are shared. But in a peer network, those shared items — a CD reader, perhaps, or a printer — reside on machines that are also used for other purposes. If you have a very small, low-traffic network, a peer-to-peer system might be right for you because it requires no dedicated server machine(s). Peer networking can prove impractical for high-volume operations because, for instance, multiple big print jobs will keep the poor soul who shares his printer from getting much else done.

What's in the Mix?

If you are only a little bit familiar with Red Hat Enterprise Linux, your exposure to it has probably relied on industry press reports extolling its suitability as a server operating system. There is no doubt that it is indeed superb for this purpose. Don't make the mistake, though, of thinking that this is *all* it is good for. Red Hat Enterprise Linux comes with a full range of powerful and secure server applications. (Secure by industry standards, although security is a process, not a state of being.) It also comes with powerful, attractive, and easy-to-use graphical desktops and a wide range of productivity applications, communications tools, and yes, even amusements, which make it an ideal client or peer operating system as well.

Your network may be a mixed marriage of machines of different architectures and operating systems. For instance, your graphics design department would probably rather paint with their fingers on a cave wall than use anything other than a Macintosh. Meanwhile your legacy applications, boilerplate documents, and relations with paying customers dictate that you maintain one or more Windows machines. In these cases, choose a client-server arrangement with a secure Red Hat box serving as a firewall between your network and the outside world, a mail server (it is now easy with Linux to filter out email attachments of the sort that have caused so much disruption of Windows networks in recent years), a Web server if you have a presence in that milieu, and even perhaps a print server.

Although many peer functions can be performed on a mixed network, your job as system administrator is much easier if you undertake the more straightforward client-server approach with a mixed installation. Additionally, if your network includes machines running Windows and is connected to the Internet, you would be irresponsible not to set up a firewall and let Linux handle Web, FTP, and mail services. History has shown that Linux is more secure in its fundamental architecture. Beyond that, however, there are thousands of eyes constantly searching for and fixing potential security exploits. Red Hat is often first to make these fixes available, usually before the exploits are known to the cracker crowd.

TIP If your network includes machines running Windows and is connected to the Internet, set up a firewall and let Linux handle your Web, FTP, and mail services.

A client-server network acts very much like a small Internet: just about any machine can connect to it and make use of its services, irrespective of its architecture or operating system.

Determining System Requirements

Depending on the kind of network you choose, you need, of course, computers and any devices you intend to connect to the hub (if you're using a star topology network). Don't forget an appropriate and supported Ethernet card for each machine — two for your firewall machine because it has one line in from the outside world and one line out to the rest of your network. You also need the appropriate cabling and, if you go the recommended star topology route, one or more hubs to support the network.

Fedora Core and Red Hat Enterprise Linux have excellent support for a broad range of Ethernet cards. Still, there are some factors to take into consideration. If you have old, 8-bit Ethernet adapters, now is the time to replace them. They are slow and often difficult to configure. Now that 100-Mbps cards are quite inexpensive, it's probably not a good idea to invest in a slower card that's slightly cheaper. Be sure to check the Red Hat hardware database at www.redhat.com/support/hardware before buying new cards; in fact, it's a good idea to go here to make sure that the ones you do have are fully supported if you're upgrading to Red Hat Linux. (You don't need to do this for Windows machines connected to an existing network because as long as they're properly configured for use with Windows, and as long as you continue to use Windows with them, they will work on your network, even though it's served by Red Hat machines. Of course, if you use 8-bit or older, slow peer network cards and you're going to star architecture, you need to replace them too.)

TIP If you have old, non-PCI Ethernet adapters, now is the time to replace them.

At this stage, you need to decide which headaches you're willing to accept and which ones might more sensibly be placed elsewhere. An example of this is Web and FTP hosting. For a few dollars per month, you can arrange for your Web site (and FTP site, if you have one) to be hosted by a large and secure commercial enterprise. Although it's fun to host your own site, it may be much more cost-effective to outsource those duties. The best ones have strict security

rules — assigned passwords and administrator access by SSH or other highly secure methods only — and offer very high bandwidth and professional administration. With such an arrangement, you can still have your own domain and still use your own local mail server, with mail downloaded from your hosting company. (Your own SMTP mail server for outgoing mail remains on a local machine.) For many smaller companies, the cost of outsourcing is more than covered by the ability to use a low-cost cable or DSL service whose terms of use prohibit Web and FTP servers, meaning that you gain an extra level of professional service at no cost — quite a bargain. Of course, if your enterprise is of sufficient size that you have a T-1 line and a huge server farm, there's little to gain by not hosting your site yourself.

Planning and Implementing Security

We cannot overstate the importance of computer security. Practically every day there are new stories of large and small systems' being cracked by vandals and other criminals. Enterprises have been held hostage as criminals threatened to release the credit card numbers of thousands or hundreds of thousands of customers. Not long ago, the entire Internet was slowed drastically because hundreds of thousands of machines, many of whose owners weren't even aware they were running Web server software, were hijacked by a series of increasingly vicious and efficient "worm" programs that tried to corrupt other machines. The attack grew to the point where there were so many machines at work scanning the Internet for potential victims that much of the Internet's bandwidth — practically all of it in some places — was consumed. System administrators who allow machines under their control to be vulnerable to this kind of attack, when there is an alternative, are certainly not doing their jobs as completely as they should be.

Addressing External and Internal Threats

The threats from the outside world are very real and very frightening. The extent to which data on your network is safe from prying eyes or random acts of destruction is entirely up to you. But your concerns cannot stop there. While cracker attacks get all the press, there are security concerns just as real that you can find within your own network, whether it's a 500-node affair in an enterprise or a 3-node setup at home.

Imagine the damage that a disgruntled employee could do or, for that matter, a child disappointed at being grounded. Imagine what harm a dishonest employee could bring about, upon gaining access to company accounts or company secrets, or a troubled teenager who finds your credit card number, which you should *never* put on your computer.

There is also the security issue of a user who simply makes a mistake that could have destroyed crucial data or brought down the entire system had certain security safeguards not been in place.

Take advantage of all the multiple lines of defense available, no matter the size of your network, even if you have a single standalone system that is physically accessible to others or that is connected to the Internet. Otherwise, assume that anything on your network (or the machines on it) is accessible to anyone else in the world who is a little bit determined to get it. Part V deals with security issues in great detail, but much of your security policy needs to be established before you install the network.

CROSS-REFERENCE To learn more about security, see Chapters 33 and 34.

Formulating a Security Policy

What should your security policy consist of? It should encompass an effective password policy, general security rules, security updates, and an appropriate firewall system.

An Effective Password Policy

Although the method brings grumbles from users, assigned, random passwords made up of a combination of numbers and uppercase and lowercase letters, all with no discernable meaning, are safest. (This procedure includes, most especially, the root account password.)

Who has access to what? Red Hat Enterprise Linux enables you to create special groups and assign users to them. This means that some users might have access to devices such as CD burners and modems, while others would not. You may have sensitive situations in which you do not want users to send files or even carry them from the building on a floppy disk. You can provide increased security by using groups. Although you don't necessarily have to set this up first thing, it's important to plan for and good to keep in mind.

General Security Rules

A server that isn't running cannot be used to crack your system. If you're not using a server application, don't run it. Change all passwords periodically. Be prompt in removing network access of any employee who is discharged. Employ intrusion detection software and check your logs regularly for anything unusual.

Security Updates

Do you subscribe to the Red Hat Linux security mailing list? (Find it at www.redhat.com/mailling-lists/linux-security/index.html.)

Have you established a procedure that makes sure that every security update is downloaded and installed? Luckily for you, Red Hat has created the Red Hat Network, a program intended to help keep your system updated with the latest security updates.

CROSS-REFERENCE Learn how to use the Red Hat Network in Chapter 26.

An Appropriate Firewall System

If yours is a standalone box on a broadband connection, the bare minimum is an Internet firewall-gateway-router appliance plus iptables. If you run a Web site, set up a separate firewall machine. (The more experienced administrator could go so far as to create a custom firewall on a bootable CD and then boot the firewall machine from that CD. It is impossible to install a root kit on a CD, and the machine can have a hard drive for logging purposes.)

Security is a process — a matter of constantly outwitting people who wish your network ill. Red Hat Enterprise Linux goes a long way toward helping you beat crackers to the punch. It's up to you to make sure that your machine is not just buttoned up tightly today but continues to be secure tomorrow and the day after.

Planning for Recovery from Disasters

Rare is the professional system administrator who hasn't been awakened in the middle of the night or who had to work an entire weekend to recover from some tremendous mishap. Likewise, rare is the owner of a standalone machine who hasn't spent frantic hours trying with varying degrees of success to recover from a catastrophic failure of hardware, software, or execution.

When you plan your systems, it is critical to keep in mind two important terms: fault tolerance and disaster recovery. Fault tolerance is the system's ability to respond automatically to various conditions that can affect system operation and resolve the condition. By responding automatically to a problem, fault tolerance reduces the effect of the problem on the system. Properly implemented fault tolerance is transparent to users of the system. They will continue to work, totally unaware that any problem has occurred.

Disaster recovery is the ability to restore functionality to the system after a failure has occurred. The system administrator must implement this process; it

does not occur automatically. The goal of disaster recovery is to restore full functionality as quickly as possible. Depending on the degree of fault tolerance your systems have, disaster recovery may not be necessary at all.

Planning for fault tolerance and disaster recovery requires that you assess the needs of your systems. The following two questions are the most important ones you should ask:

- How critical are the systems to daily operation?
- Could the systems be down and not affect operations?

Obviously, if your systems are used for business, they could not be down for long, if at all. You must determine how vital a given system is to your operation. Vital systems require greater fault tolerance than nonvital systems do. Be sure to keep in mind that greater fault tolerance costs more than less fault tolerance does. Another important consideration is the amount of money available for building fault tolerance into your system. Balance your need for fault tolerance with the amount of money you can spend on it.

Clustering Solutions

If your systems must be up 24/7, you have to rely on a clustering solution for your fault tolerance. You basically have two choices for clustering: failover clustering and true clustering. Both of these solutions are costly to implement and require a great deal of configuration. Configuring these types of systems is well beyond the scope of this book, but a brief description of the two types of clustering is useful.

Failover clustering typically requires two systems. The first system is the active system that responds to service requests. The second, failover system is an exact copy of the first system that is connected to the first system by a dedicated link. The second system uses the dedicated link to listen for a signal — called a heartbeat — from the first system at a specified interval. The second system does nothing but listen to the heartbeat signal from the first system. If the second system does not receive the signal, it assumes that the first system has gone offline and immediately begins accepting requests for services. When the first system comes back online, the second system returns to monitoring the heartbeat signal.

True clustering uses multiple systems, usually more than two, often in different locations, that act as a single system. Network services run on each system and requests for services are distributed between the systems. Each system is connected to every other system by a dedicated link. Unlike the second system in failover clustering that only listens for the heartbeat, the systems in true clustering handle requests for services. If a system does go down, the requests for service are just sent to the other systems, which take up the slack. Neither clustering solution employs disaster recovery. Because they

must be up 100 percent of the time, there is no disaster from which to recover, except perhaps the disaster to your budget because the cost of implementing such a system is quite high.

Disaster Recovery

For systems that do not require 100 percent uptime, disaster recovery is the method used. A typical solution is to configure an identical system and keep it ready for service. Placing the other system into service requires intervention by the administrator and no services will be possible during this time. Service can usually be restored fairly quickly using this method, and the cost is less than the cost of a clustering solution.

The least costly method (in hardware) of dealing with system problems is to fix them after they have occurred. Here, you shut down the system until it is fixed; no services are available during the repair. For example, if the hard drive in your system crashes, you simply replace the hard drive.

System administrators who plan their network well may not be able to prevent disasters entirely, but they greatly reduce the likelihood of such events taking place and make complete or near-complete recovery a quick and orderly process.

Planning for recovery ideally involves considering everything bad that can possibly happen and figuring out a way around it. However, that which is ideal often does not square with what's practical, especially when it involves spending money to guard against an infinitesimal likelihood. Fortunately, the things that save you from likely disasters save you from the most unlikely ones, too.

Just as security planning requires attention to threats from outside and inside the network, there are two parts to disaster planning. The first is doing everything you can to prevent a catastrophe from taking place.

Only you, or other administrators at your organization, know how important your system is and how much money is budgeted to keep it running. Chances are good that an uninterruptible power supply (UPS) that keeps the network up long enough to save and close files and shut down the system in an orderly fashion fits within the available budget. A good UPS system is especially useful if your enterprise has a generator backup that kicks on in the event of power failure because generators do not always start instantly and, when they do, the electricity provided is not always clean enough for computer use. A battery backup can protect you from both of these potential problems. If your enterprise is important enough to have an emergency generator, it's probably important enough to keep the network running.

Renegade electricity is one of the worst enemies of system reliability. Small power strips with surge suppression are better than nothing, but more robust power conditioning is needed if really important equipment and data are to be protected. In fact, be sure to protect all lines from the outside world that attach

to your computer or its peripherals, be they phone lines or cable or DSL connections. Likewise, put the peripherals themselves on protected circuits.

Second, formulate a regular (daily or better) backup scheme, with one set of backups stored in a safe place off-site as protection against loss of data in the event of fire, flood, tornado, or other physical disaster. One way of making this process relatively painless, albeit an expensive one, is to rent storage from a commercial operation whose business is storing other people's data. The best firms are very responsive and secure.

Redundancy is also important. Make sure that your plans don't put critical data on only one machine. That way, in the event of a machine failure, a replacement machine with a copy of your critical data can be put online very quickly. This is some, but not all, of the theory behind redundant array of independent disks (RAID) systems, in which multiple hard drives in the same machine contain the same data. RAID is good protection in case any one drive fails. (The best RAIDs allow the hot-swapping of drives so that a replacement can be added without bringing the system down.) But RAID also allows for much faster data access, making it especially useful for file server machines. Don't be lulled into complacency by a RAID, though; there are computer failure modes that can render an entire system useless. In keeping with Murphy's Law, the worst failures and calamities occur at the worst possible time — just before the rollout of a new product, just as the monthly billing is scheduled to go out, in the middle of the worst blizzard in 10 years, or when most of the computer staff is on vacation or out sick. You need to establish an emergency response policy that takes these examples, and there are many others, into account. This involves convincing your employer of the necessity of having sufficient staff to guard against such horrors, or even the employment of an outside firm to augment your own staff in the event of an especially ill-timed disaster. If your company follows the latter route, it's well worth the investment of time and money to make sure that the outside firm's representatives tour and learn your network on a day when everything is working smoothly.

Some of this planning is far more elaborate than anything you're likely to undertake if you have only a small household network or a very small office; on the other hand, if you're in a very large enterprise, data security and system integrity involve issues and procedures far beyond the scope of this book. Everything mentioned in this section, however, can be scaled to fit any network.

Writing It Down: Good Records Can Save Your Job

A very important part of network planning is to put it all down on paper and to save that piece of paper. Working out your network's design is best done by actually diagramming the network, making multiple diagrams to explore different strategies. Once you settle on a design, draw a more formal diagram.

Sometimes it's a good idea to save your discarded designs as well, with a note on each version explaining why it wasn't chosen. Formalizing the network design and saving the discarded ideas is useful for several reasons. It bolsters your decisions in case you're second-guessed, it demonstrates that you considered all the possibilities, and the formal diagram is a valuable tool should someone need to administer the system in your absence.

A written security policy is essential in the enterprise and not a bad idea even for a home network. An additional security file you should always keep is a full security log. Such a record might begin by detailing what security measures you have designed into the system. It should include copies of any security notices you have received, as well as an initialed notation of when the recommended security patch was applied. If log files show an attempted crack of your network, hard copies of the relevant portions should be kept there, too.

When users or management complain about how you have the system so tight that it seems inconvenient even for them to log in, there's nothing like proving that the system is regularly under attack — and it will be, by port scanners and others — to demonstrate the wisdom of tight security. One very big company has made huge amounts of money by putting user convenience over security, and many companies have paid a high price for adopting their products. Your Red Hat system costs a very small amount in user inconvenience in exchange for greatly enhanced system security. It's useful to be able to prove that the threat is real.

A security log is also the place to keep copies of any security-related email messages from within the company, from log listings of employees who have decided to “go exploring” (which is sometimes but not always a sign of bad intent) to exchanges with management over the implementation of new security features. This file is not something for general consumption, but it's very important. Keep a copy locked away at work, and it won't hurt to keep a copy safely off-site, too.

CROSS-REFERENCE To learn more about writing a security policy, see Chapter 34.

While your security log should detail actions you have taken to prevent disaster and actions you have recommended in that regard, your plan of action in the event of a catastrophe should also be committed to paper and should be well known and easily available. If you are the sole administrator, it is far better to work out your plan of action calmly and ahead of time, which of course you will have done. But under the stress of an actual emergency, it is easy to forget important details. Having a specific plan on paper right in front of you is a big help and a great stress reliever. Your action plan should be sufficiently detailed so that if the disaster takes place while you are away, any competent

system administrator can use it to bring the system back up. If you are part of a larger department, include the assignments of others in restoring the system. In either case, someone who is completely trusted and who is never on vacation at the same time you are should know the root's password. Alternately, the password can be placed in a sealed envelope inside the company safe — the one time it is allowable to put a password on paper.

TIP Keep a hard copy of your security log in a safe place!

We're all happy with the idea of the paperless office, but until computers become perfectly reliable, paper — as a roadmap, indicating where you are and how you arrived there — will remain necessary.

Summary

In this chapter you learned the importance of planning your network before you begin to construct it, discovered some of the options available to you, and found out some of the reasons why you might choose one over another. You learned that network security is a never-ending task made easier by careful planning and that threats can come both from outside the network and from among its users. Working to prevent catastrophic failures and having a plan to recover from them is something you've learned to do. You now know the importance of putting it all on paper as you proceed, too.

Standard Installation

IN THIS CHAPTER

- Exploring Your PC's Components
- Checking for Supported Hardware
- Creating the Red Hat Boot Disk
- Starting the Installation
- Partitioning the Hard Disk
- Using Disk Druid to Partition Your Disks
- Configuring the Installation
- Selecting the Package Groups to Install
- Running Firstboot

This chapter explains the steps necessary to install Red Hat Enterprise Linux and Fedora Core on a single system. You begin by making a list of your PC's hardware. You use this hardware inventory later when you begin the installation.

NOTE When you purchase Red Hat Enterprise Linux, you are eligible for installation support from Red Hat. Also, an online installation manual is available on the Red Hat Web site at www.redhat.com/docs. There is no official support for Fedora Core from Red Hat.

NOTE The installation processes for Red Hat Enterprise Linux and Fedora Core are nearly identical. Throughout the remainder of this chapter, we will refer to both Red Hat Enterprise Linux and Fedora Core as Red Hat Linux except in the instances where it is necessary to make a distinction between them. The figures in the chapter show the Fedora installation screens, but with the exception of the name on the screen (Fedora or Enterprise Linux), the content of the installation screens is identical.

Exploring Your PC's Components

Before installing Red Hat Linux, you should compile a list of the hardware components in your computer. Linux supports different types of hardware through software components called *device drivers*, similarly to other operating systems. A driver is required for each type of peripheral device; depending on the age of your hardware, a driver may not be available. If your hardware is current, meaning less than two years old, the drivers you need are probably available and included with the distribution. If you need a driver that is not included with the distribution, searching the Internet usually provides you with a solution.

You can install and run Red Hat Linux even if no Linux drivers are available for certain devices. Of course, those devices won't function, but this may not be a problem for you, depending on the device. To be able to install Red Hat Linux, you must have a compatible processor, bus type, floppy disk, hard disk, video card, monitor, keyboard, mouse, and CD-ROM drive. If you are planning to use a graphical user interface (GUI), such as GNOME or KDE, you must ensure that XFree86 (the X Window System for Linux) supports the mouse, video card, and monitor. Nearly all devices made within the past two years are supported.

The following sections briefly describe the supported PC hardware. Your hardware list should contain information about the hardware described here before you begin to install Red Hat Linux on your PC.

Processor

The *central processing unit (CPU)* — or just *the processor* — is an integrated circuit chip that performs nearly all control and processing functions in the PC. Both Red Hat Enterprise Linux and Fedora Core run on an Intel 80386 processor or newer, as well as compatibles made by AMD or Cyrix. However, you probably don't want to use any processor older than a Pentium-class processor. Red Hat Linux also supports motherboards with multiple processors that use the *symmetric multiprocessing (SMP)* Linux kernel.

Bus

The *bus* provides the electrical connection between the processor and its peripherals. Several types of PC buses exist on the motherboard with slots to accept peripheral components. Each of the slots is colored to help in its identification. The most recent is the *Peripheral Component Interconnect (PCI)* bus, and it is found on all current production motherboards. The PCI slot is white and is available in 32- and 64-bit form as well as 33 and 64 MHz. The new PCI-X standard will

support speeds up to 533 MHz. Another type of slot is also based on the PCI bus specifications, but offers significant advantages over the PCI bus. The *Accelerated Graphics Port (AGP)* is a special slot on the motherboard designed to accept an AGP graphics card. The AGP slot is brown. Another is the *Industry Standard Architecture (ISA)* bus, formerly called the AT bus because IBM introduced it in the IBM PC-AT computer in 1984. The ISA bus is black. Other, less frequently encountered, buses because of their aging status include *Extended Industry Standard Architecture (EISA)*; *VESA local (VL-bus)*; and *Micro Channel Architecture (MCA)*. Red Hat Enterprise Linux supports all of these buses.

Memory

Referred to as *random access memory*, or *RAM*, is not a consideration in determining compatibility. This means that Linux does not care what kind of memory it is or how fast it is, it just uses whatever is there. For good performance though, you need at least 64 MB of RAM for a text install and 192 MB for a graphical install. If you are planning to run the X Window system to use a graphical user interface (GUI) on the PC, you need even more memory because the X Window System manages the graphical interface through an X server, which is a large program that needs a lot of memory to run efficiently. Red Hat recommends a minimum of 256 MB RAM to run a graphical system.

TIP If you are buying a new PC, it probably comes with 128 MB or more RAM.

If you can afford it, buy as much RAM as you can. The more RAM a system has, the more efficiently it runs multiple programs (because the programs can all fit in memory). Red Hat Linux can use a part of the hard disk as virtual memory. Such disk-based memory, called *swap space*, is much slower than physical memory.

Video Card and Monitor

If you are not planning to use the X Window system, any video card works. Red Hat Linux supports all video cards in text mode. If you are planning to use the X Window system, be sure to find a video card that is supported by *XFree86*, which is the version of the X Window System used in Red Hat Linux. It is pretty unlikely that you would find a video card that doesn't work with X, but you can save yourself a lot of aggravation if your video card is supported by *XFree86*.

Your choice of monitors depends on your use of the X Window system. For text mode displays, typically used on servers, any monitor will do. If you are setting up a workstation, or using the X Window system on your server, choose

a monitor that supports the display resolution you use. Resolution is expressed in terms of the number of picture elements, or *pixels*, horizontally and vertically (such as 1024 × 768).

XFree86's support for a video card depends on the *video chipset* — the integrated circuit that controls the monitor and causes the monitor to display output. You can find out the name of the video chipset used in a video card from the card's documentation.

Your video card's name may not be in the list at the Red Hat site. The important thing to note is the name of the video chipset. Many popular video cards made by different manufacturers use the same video chipsets. Look for the name of the video chipsets listed at the Red Hat site. In nearly all cases, the Red Hat installation program automatically detects the video chipset as it sets up the X Window System.

Hard Drive

Red Hat Linux supports any IDE hard drive that your PC's *basic input/output system* (BIOS) supports, as long as the system BIOS supports the hard drive without any additional drivers. This would include EIDE- and ATA-compatible drives as well.

For hard drives connected to your PC through a Small Computer System Interface (SCSI) controller card, Red Hat Linux must have a driver that enables the SCSI controller to access and use the hard drive. If you have a recent SCSI controller card, there is most likely a driver for it already included with the distribution.

Also supported are Serial Advanced Technology Attachment (SATA) drives, which use serial technology instead of the parallel ATA technology currently used by IDE drives. SATA provides a significant speed increase over IDE.

As for the size (storage capacity) of the drive, most new systems seem to have drives 20 GB or larger. You should buy the highest capacity drive you can afford.

Floppy Disk Drive

Linux drivers use the PC BIOS to access the floppy disk drive, so any floppy disk drive is compatible with Red Hat Linux. The Red Hat installation program can be started from the CD-ROM if your PC has one and is able to boot from it. If not, you have to boot Red Hat Linux from a floppy disk drive during the installation, so you need a high-density 3.5-inch (1.44-MB capacity) floppy disk drive. You can also avoid booting from a floppy if you can boot your PC under MS-DOS (not an MS-DOS window under Windows 95/98/2000), and you can access the CD-ROM from the DOS command prompt.

Keyboard and Mouse

Red Hat Linux supports any keyboard that already works with your PC. The mouse, however, needs explicit support in Red Hat Linux. You need a mouse if you want to configure and run XFree86, the X Window System for Linux. Red Hat Linux supports most popular mice, including the commonly found PS/2 and USB mouse. Red Hat Linux also supports touch pad devices, such as ALPS GlidePoint, as long as they are compatible with one of the supported mice.

SCSI Controller

The Small Computer System Interface, commonly called *SCSI* (and pronounced “skuzzy”), is a standard way of connecting many types of peripheral devices to a computer. SCSI is used in many kinds of computers, from servers to high-end UNIX workstations to PCs. Typically, you connect hard drives and CD-ROM drives through a SCSI controller. To use a SCSI device on your PC, you need a SCSI controller card that plugs into one of the bus connector slots on your PC’s bus.

If you want to access and use a SCSI device under Linux, you have to make sure that Red Hat Linux supports your SCSI controller card.

CD/DVD-R/RW Drive

CD-R (compact disc read-only) drives are popular because each CD-ROM can hold up to 650 MB of data, a relatively large amount of storage compared with a floppy disk. CD-ROMs are reliable and inexpensive to manufacture. Vendors can use a CD-ROM to distribute a large amount of information at a reasonable cost.

DVD-ROM drives are found already installed on many new systems. DVD-ROM discs are capable of storing up to 4.7 GB and are most frequently used to record digital video, but can be used to hold any data.

CD-RW and DVD-R/RW and DVD+R/RW drives are used to create CDs and DVDs, respectively. Either of these types of drives can be used in your Red Hat system. Any IDE/ATAPI-compatible drive, as well as SCSI drives, will work with Red Hat Enterprise Linux.

Sound Card

If you are configuring a server, you probably aren’t too interested in playing sounds. But with Red Hat Linux you can play sound on a sound card to enjoy multimedia programs and games. If you have a sound card, you can also play audio CDs. Nearly all sound cards available today, whether built into the motherboard or a separate card that plugs into a bus socket, are supported.

Network Card

A *network interface card (NIC)* is necessary if you connect your Red Hat Linux PC to a *local area network (LAN)*, which is usually an Ethernet network. If you are configuring a server, you certainly want to configure at least one network card. Red Hat Enterprise Linux supports a variety of Ethernet network cards. ARCnet and IBM's Token Ring network are also supported. Check the hardware list on the Red Hat site to see if your NIC is supported. Nearly all NICs currently in use are supported.

For any Red Hat Linux PC connected to a network, you need the following information:

- Hostname of the PC
- Domain name of the network
- Internet Protocol (IP) address of the PC
- Address of the gateway
- IP address of name servers

NOTE If you plan to use DHCP to obtain your IP information, you do not need to specify the IP information in the above list.

Checking for Supported Hardware

To check if Red Hat Linux supports the hardware in your PC, follow these steps:

1. Make a list of the make, model, and other technical details of all hardware installed in your PC. Most of this information is in the manuals that came with your hardware. If you don't have the manuals, and you already have an operating system (MS Windows for example) on the PC, you may be able to obtain this information from that operating system. Refer to that operating system's instructions for obtaining hardware information.
2. Go to the Red Hat Web site at redhat.com/hardware. Compare your hardware list to the list of hardware that the latest version of Red Hat Linux supports. If the components listed earlier are supported, you can prepare to install Red Hat.

NOTE You do not need a boot disk if you can start your PC from your CD-ROM drive. The first installation disk is a bootable CD-ROM and can be used to start the installation process. If you can boot from your CD-ROM, skip to the “Starting the Red Hat Linux Installation” section. If you are unable to boot from your CD-ROM drive, continue to the next section, “Creating the Red Hat Boot Disk,” and then go to the installation section.

Creating the Red Hat Boot Disk

To boot Red Hat Linux for the first time and start the Red Hat Linux installation program, you need a Red Hat boot disk. For this step, you should turn on your PC without any disk in the A drive and run Windows as usual.

NOTE You do not need a boot disk if you can start your PC under MS-DOS (not an MS-DOS window in Windows) and access the CD-ROM from the DOS command prompt. If you run Windows, restart the PC in MS-DOS mode. However, you may not be able to access the CD-ROM in MS-DOS mode because the startup files (`AUTOEXEC.BAT` and `CONFIG.SYS`) may not be configured correctly. To access the CD-ROM from DOS, you typically must add a CD-ROM driver in `CONFIG.SYS` and add a line in `AUTOEXEC.BAT` that runs the `MSCDEX` program. Try restarting your PC in MS-DOS mode and see whether the CD-ROM can be accessed.

The Red Hat boot disk starts your PC and the Red Hat Linux installation program. After you install Red Hat Linux, you no longer need the Red Hat boot disk (except when you want to reinstall Red Hat Linux from the CD-ROMs).

The Red Hat boot disk contains an initial version of the Red Hat Linux installation program that you use to start Red Hat Enterprise Linux, prepare the hard disk, and load the rest of the installation program. Creating the Red Hat boot disk involves using a utility program called `RAWRITE.EXE` to copy a special file called the Red Hat Linux boot image to a disk.

To create the Red Hat boot disk under Windows, follow these steps:

1. In Windows 95/98/ME open an MS-DOS window (select Start ⇨ Programs ⇨ MS-DOS Prompt). In Windows 2000 or XP, select Start ⇨ Run and enter **cmd** in the dialog box.
2. In the MS-DOS window, enter the following commands at the MS-DOS prompt. (Our comments are in parentheses and your input is in boldface.)

- a. **d:** (use the drive letter for the CD-ROM drive)
 - b. **cd \dosutils**
 - c. **rawrite**
 - d. Enter disk image source filename: **\images\boot.img**
 - e. Enter target disk drive: **a**
 - f. Insert a formatted disk into drive A and press **ENTER**.
3. As instructed, you should put a formatted disk into your PC's A drive and press Enter. **RAWRITE.EXE** copies the boot-image file to the disk.

When the DOS prompt returns, remove the Red Hat boot disk from the floppy drive and label it as a Red Hat boot disk.

Starting the Installation

To start the Red Hat Linux installation, power up the PC and put the Red Hat Installation CD-ROM 1 (and the boot disk if you created one) into your PC's CD-ROM drive (and floppy drive if applicable). The PC boots Red Hat Linux and begins running the Red Hat installation program. The Red Hat installation program controls the installation of the operating system.

NOTE If you are using a boot disk that you created, be sure to place the first installation CD-ROM in the CD-ROM drive after you start the PC. The installation program looks for the Red Hat Linux CD-ROMs to start the installation in graphical mode. If the installation program can't find the CD-ROM, the installation program starts in text mode and prompts for it.

A few moments after you start the boot process, an initial screen appears. The screen displays a welcome message and ends with a **boot :** prompt. The welcome message tells you that more information is available by pressing one of the function keys F1 through F5.

If you want to read the help screens, press the function key corresponding to the help you want. If you don't press any keys after a short delay, the boot process proceeds with the loading of the Linux kernel into the PC's memory. To start booting Red Hat Linux immediately, press Enter.

NOTE On CDs that you made from downloaded ISO files (Fedora Core) you are prompted to check the CD media for errors. The disk-checking process takes a few minutes but is time well spent to be sure there are no errors on your installation CDs. To begin disk checking, press Enter. You will be prompted to change the disks as required. You can also choose to skip disk checking by using the Tab key to highlight Skip and then pressing Enter. If you purchased Red Hat Enterprise Linux, you are not prompted to check the disks.

After the Linux kernel loads, it automatically starts the Red Hat Linux installation program. This, in turn, starts the X Window System, which provides a graphical user interface for the installation.

You should compile all the configuration information explained earlier in this chapter before you begin. If the installation program detects your hardware, installing Red Hat Linux from the CD-ROM on a 200-MHz or better Pentium PC should take 30 to 40 minutes.

NOTE During installation, the Red Hat installation program tries to determine the hardware in your PC and alters the installation steps as required. For example, if the installation program detects a network card, the program displays the appropriate network configuration screens. If a network card is not detected, the network configuration screens are not displayed. So, depending on your specific hardware, the screens you see during installation may differ from those shown in this section.

CROSS-REFERENCE If you run into any problems during the installation, refer to Chapter 35 to learn how to troubleshoot common installation problems.

You go through the following steps before moving on to disk setup and installation:

1. The installation program starts the X Window System and displays a welcome message that provides some explanatory information on the left side of the screen. Take a look at this information and be sure to click the Release Notes button. When you are finished reading, click the Next button to proceed.

2. After clicking Next, a list of languages to use during the installation is displayed, as shown in Figure 3-1. Use your mouse to select the language you want to use for installation, and then click the Next button to proceed to the next step.

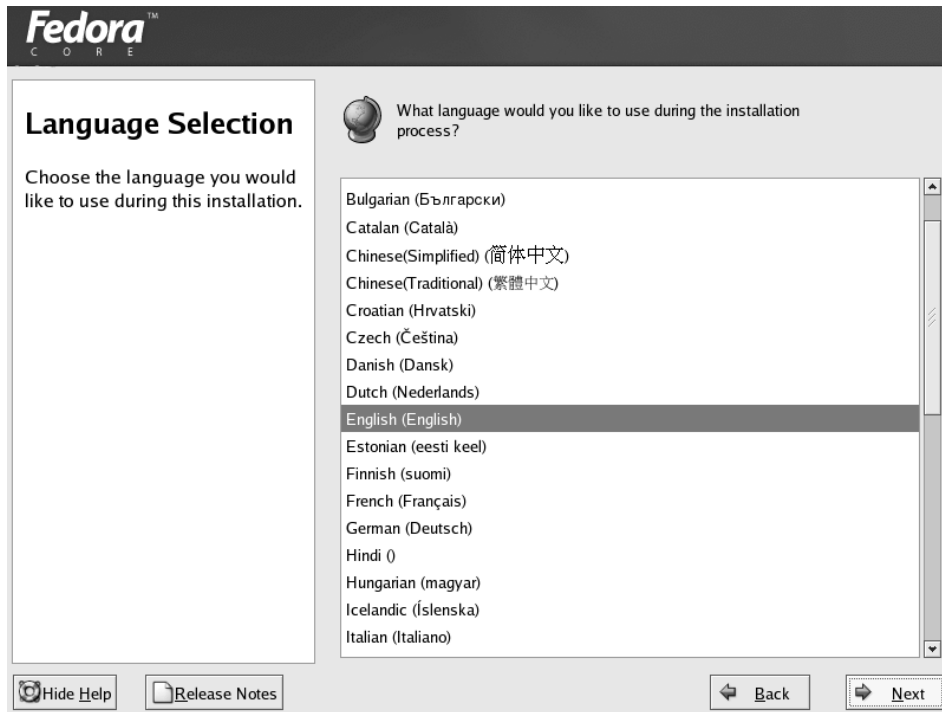


Figure 3-1 Choosing the installation language.

NOTE In the graphical installation, each screen has online help available on the left side of the screen. You can read the help message to learn more about what you are supposed to select in a specific screen.

3. The installation program displays a list of keyboard language layouts, as shown in Figure 3-2. Select a keyboard language layout appropriate to the language you desire.

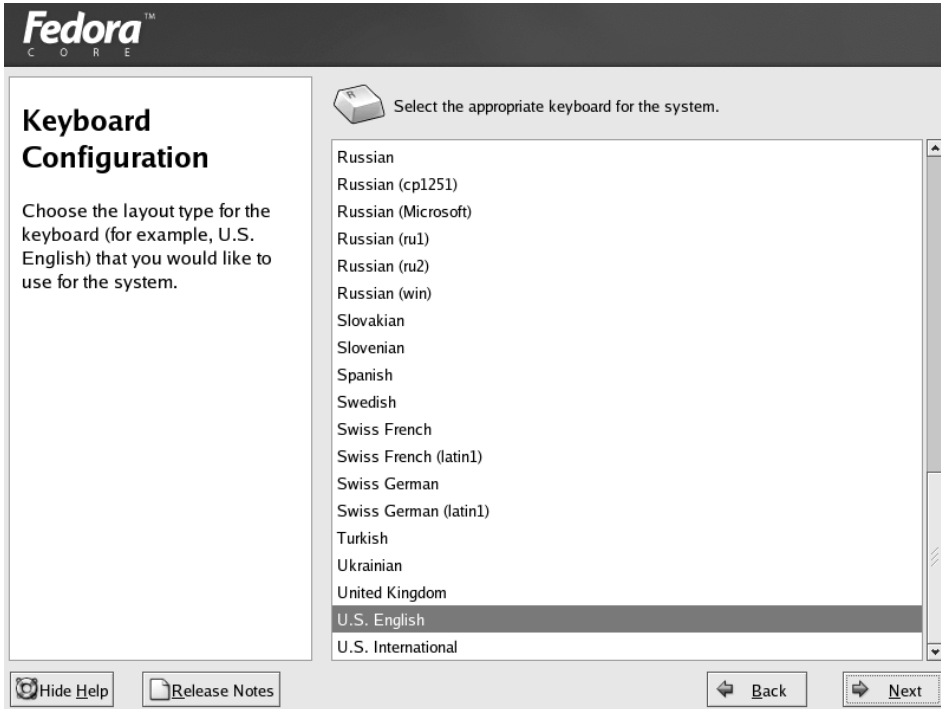


Figure 3-2 Selecting a keyboard configuration during Red Hat Linux installation.

4. The installation program searches for previous installations of Enterprise Linux or Fedora Core and displays a screen, shown in Figure 3-3, asking if you want to install a new system or upgrade an older Red Hat installation if one is detected. Select the choice that is appropriate for your system, and click Next to continue.

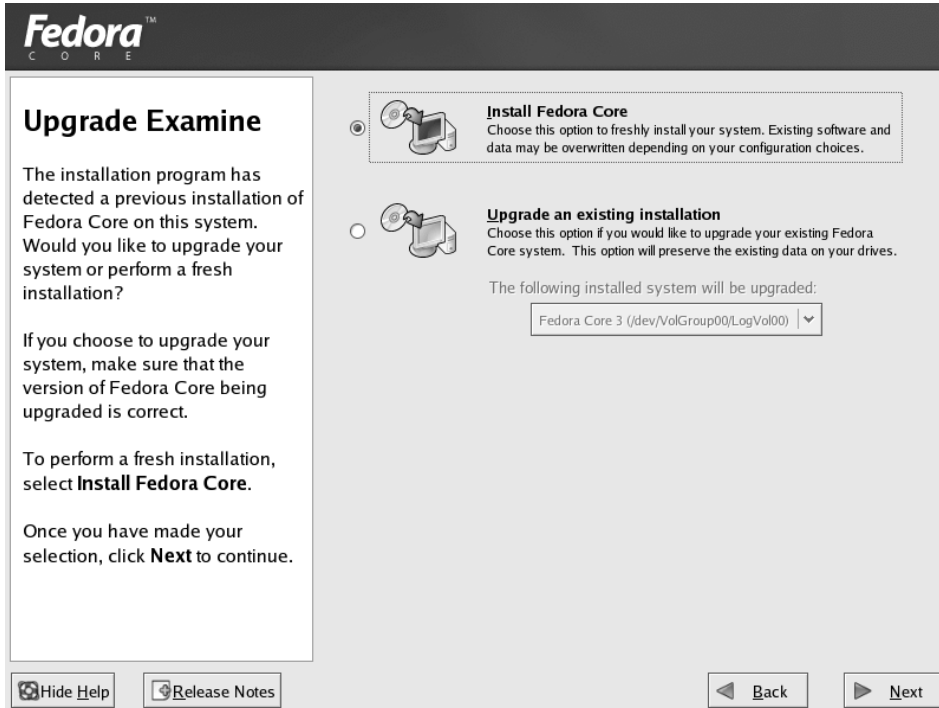


Figure 3-3 Choosing to install a new system or upgrade an existing one.

5. If you select new installation, the installation program displays a screen, shown in Figure 3-4, that requires you to select the installation type: **Personal Desktop**, **Workstation**, **Server**, or **Custom**.

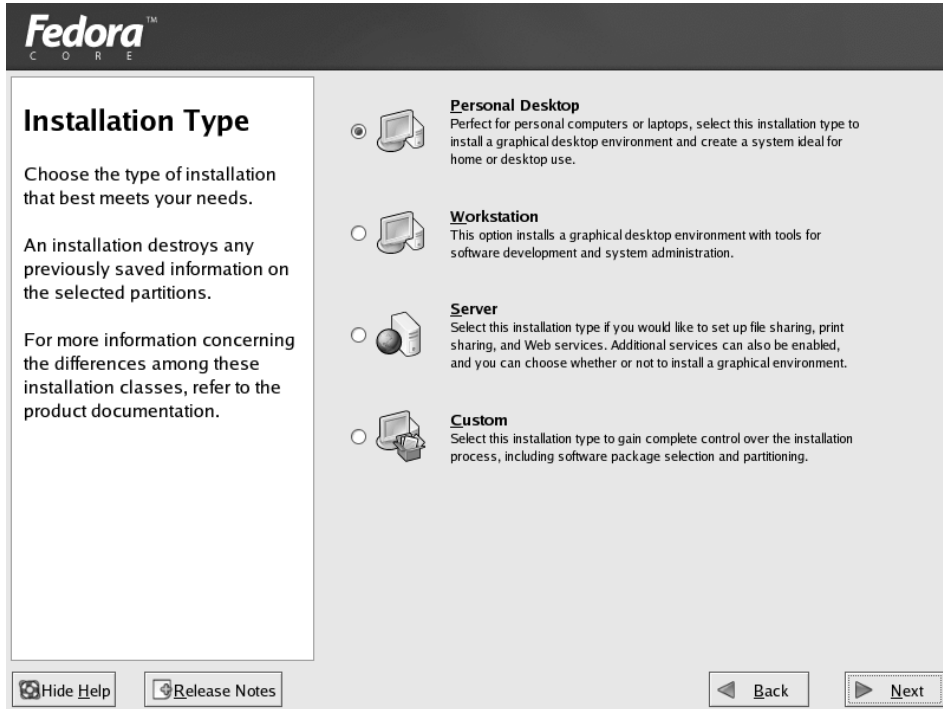


Figure 3-4 Choosing the installation type for your system.

6. There is a brief description of each installation type, and you should choose the appropriate type for your system, depending on how you plan to use the system. After making your selection, click Next to go on to the Disk Partitioning Setup screen shown in Figure 3-5.

The next major phase of installation involves partitioning the hard disk.

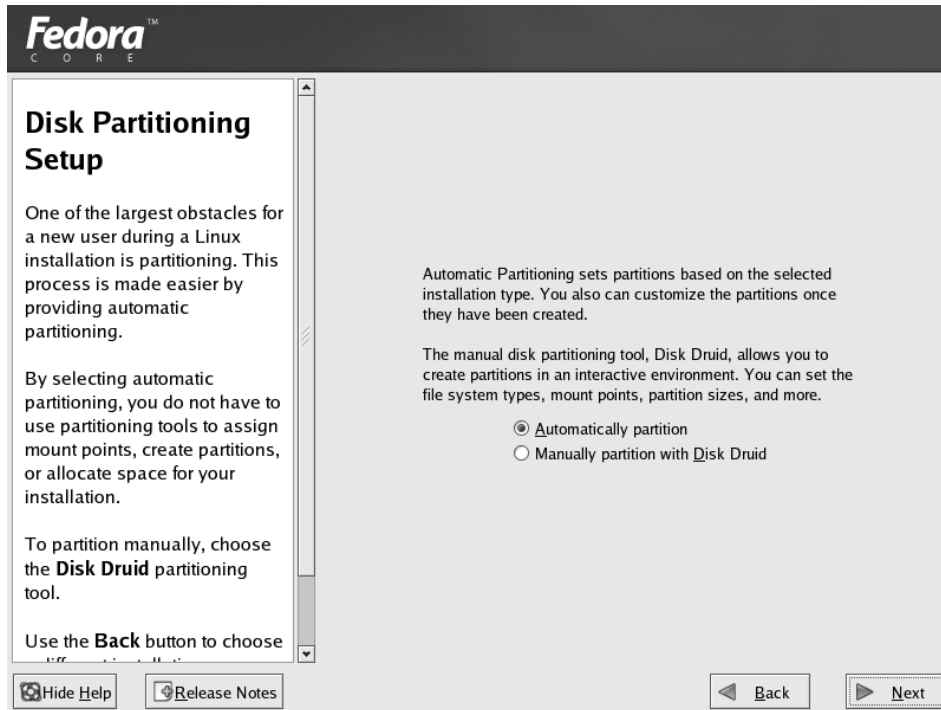


Figure 3-5 Choosing to use automatic partitioning or manual partitioning.

Partitioning the Hard Disk

Red Hat Linux requires you to partition and prepare a hard disk before you can begin installation. With a new PC that you purchase from a vendor, you usually do not perform this step because the vendor normally takes care of preparing the hard disk and installing the operating system and other applications on the hard disk. Because you are installing Red Hat Linux from scratch, however, you have to perform this crucial step yourself. As you see in the following sections, this task is just a matter of following instructions.

The Red Hat Linux installation program offers you two choices for partitioning your hard drive. You can choose to have the installation program automatically partition your disk or you can choose to use Disk Druid to manually partition your drives. If you select automatic partitioning, the installation program does all the hard work for you and creates the partitions and allocates space for your file system. If you want to manually partition your disks, go to the section “Using Disk Druid to Partition Your Disks.”

1. To use automatic partitioning be sure the radio button in front of this choice is checked, and click Next on the Disk Partitioning Setup screen.
2. The Automatic Partitioning screen, shown in Figure 3-6, appears. Here you decide how automatic partitioning should handle existing partitions. You can choose to remove Linux partitions, remove all partitions, or keep all partitions and use free space. If you are installing a new system, you should choose to remove all partitions. A warning screen will appear asking if you are sure you want to remove all partitions. Click Yes to continue.

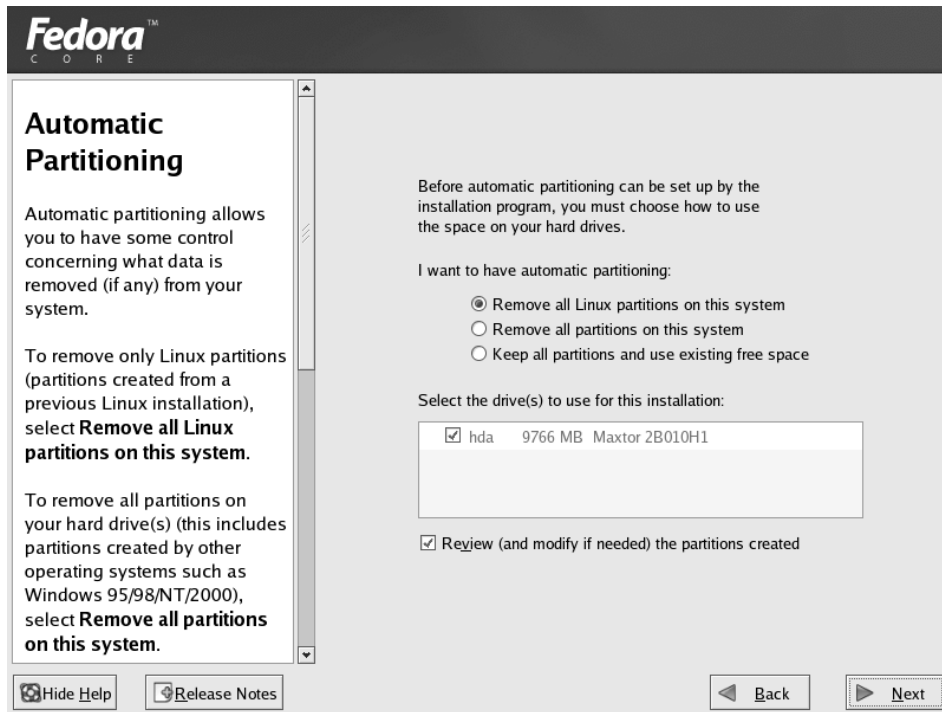


Figure 3-6 Choosing how to use automatic partitioning on your disk partitions.

3. The Disk Setup screen shown in Figure 3-7 appears. The Disk Setup screen shows the partition setup that automatic partitioning has selected for your system. Click Next to accept the settings and go the section titled “Configuring the Installation.”

TIP If you want to change any of the settings here you still can, but then you will be doing manual partitioning. Continue to the next section to learn how to manually partition your disks.

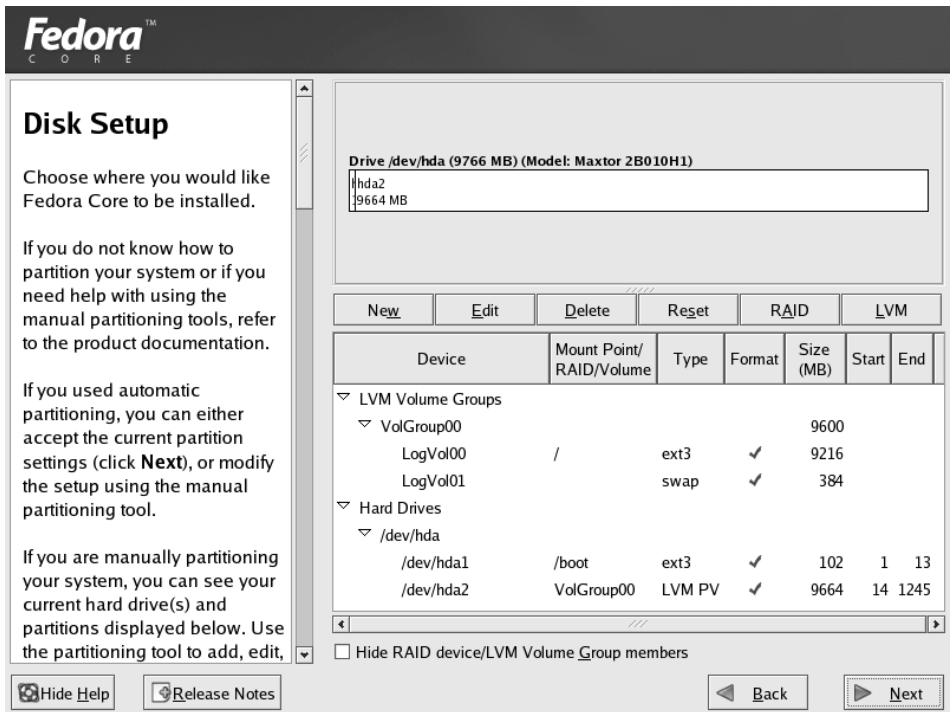


Figure 3-7 Reviewing the disk partitioning settings selected by automatic partitioning.

Using Disk Druid to Partition Your Disks

Before you begin to use Disk Druid to partition your disk, you need to know how to refer to the disk drives and partitions in Linux. Also, you should understand the terms *mount points* and *swap partition*. In the next three sections, you learn these terms and concepts and then proceed to use Disk Druid.

Naming Disks and Devices

If you are experienced with UNIX or Linux, this section and the two following are quite basic to you. If you are already familiar with UNIX and Linux file systems and naming conventions, skip to the section titled “Preparing Disk Partition.” The first step is to understand how Red Hat Linux refers to the various disks. Linux treats all devices as files and has actual files that represent each device. In Red Hat Linux, these *device files* are located in the `/dev` directory. If you are new to UNIX, you may not yet know about UNIX filenames. But you’ll learn more as you continue to use Red Hat Linux. If you know how MS-DOS filenames work, you find that Linux filenames are similar. However, they have two exceptions: they do not use drive letters (such as A and C), and they substitute the slash (/) for the MS-DOS backslash (\) as the separator between directory names.

Because Linux treats a device as a file in the `/dev` directory, the hard disk names start with `/dev`. Table 3-1 lists the hard disk and floppy drive names that you may have to use.

Table 3-1 Hard Disk and Floppy Drive Names

NAME	DESCRIPTION
<code>/dev/hda</code>	First Integrated Drive Electronics (IDE) hard drive (the C drive in DOS and Windows) connected to the first IDE controller as the master drive
<code>/dev/hdb</code>	Second (IDE) hard drive connected to the first IDE controller as the slave drive
<code>/dev/hdc</code>	First (IDE) hard drive connected to the second IDE controller as the master drive
<code>/dev/hdd</code>	Second (IDE) hard drive connected to the second IDE controller as the slave drive
<code>/dev/sda</code>	First Small Computer System Interface (SCSI) drive or first SATA drive
<code>/dev/sdb</code>	Second SCSI drive or second SATA drive
<code>/dev/fd0</code>	First floppy drive (the A drive in DOS)
<code>/dev/fd1</code>	Second floppy drive (the B drive in DOS)

TIP When Disk Druid displays the list of partitions, the partition names take the form `hda1`, `hda2`, and so on. Linux constructs each partition name by appending the partition number (1 through 4 for the four primary partitions on a hard disk) to the disk's name. Therefore, if your PC's single IDE hard drive has two partitions, notice that the installation program uses `hda1` and `hda2` as the names of these partitions.

Mounting a File System

In Red Hat Linux, you use a physical disk partition by associating it with a specific part of the file system. This arrangement is a hierarchical directory — a *directory tree*. If you have more than one disk partition (you may have other disks with Linux partitions), you can use all of them in Red Hat Linux under a single directory tree. All you have to do is decide which part of the Linux directory tree should be located on each partition — a process known in Linux as *mounting a file system on a device*. (The disk partition is a device.)

NOTE The term *mount point* refers to the directory you associate with a disk partition or any other device.

Suppose you have two disks on your PC and you have created Linux partitions on both disks. Figure 3-8 illustrates how you can mount partitions on different parts of the Linux directory tree (the *file system*).

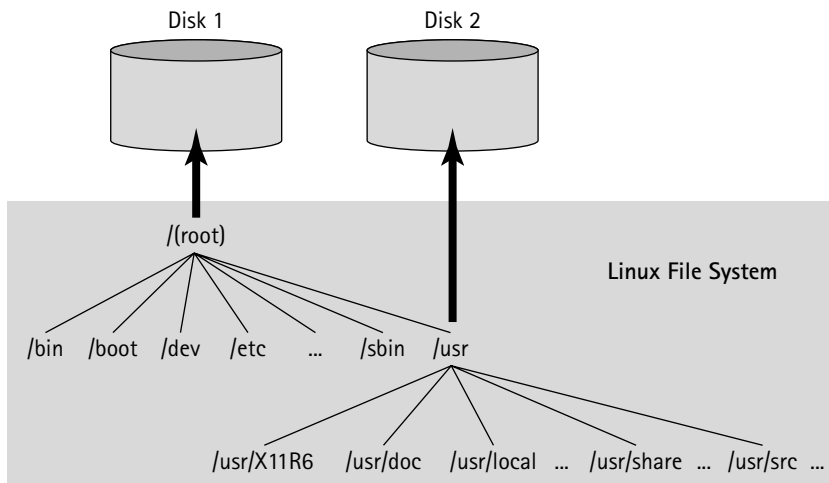


Figure 3-8 Mounting the Red Hat Linux file system on two disk partitions.

Understanding the Swap Partition

Most advanced operating systems support the concept of *virtual memory*, in which part of your system's hard disk functions as an extension of the physical memory (RAM). When the operating system runs out of physical memory, it can move (or swap out) the contents of currently unneeded parts of RAM to make room for a program that needs more memory. When the operating system needs to access anything in the swapped-out data, it has to find something else to swap out and then it swaps in the required data from disk. This process of swapping data back and forth between the RAM and the disk is also known as *paging*.

Because the disk is much slower than RAM, the system's performance is slower when the operating system has to perform a lot of paging. However, virtual memory enables you to run programs that you otherwise couldn't run by using the swap partition.

Red Hat Enterprise Linux supports virtual memory and can make use of a swap partition. When you create the Linux partitions, you also create a swap partition. With the Disk Druid utility program, described in the next section, creating a swap partition is easy. Simply mark a partition type as a swap device, choose the size, and let Disk Druid perform the necessary tasks.

Preparing Disk Partitions

After you select Disk Druid to manually partition your disks, the Disk Setup screen, shown in Figure 3-7, reappears.

Before beginning to partition the drive, consider exactly how you want to create the partitions. You must create one partition on the drive to be used as the root (/) partition. This works well in most cases, but it can cause some problems. If the root partition should become full, the system could crash. Many times the partition fills up because of system logging, email, and print queue files. These files are all written to the `/var` directory by default, so it may be a good idea to create a separate partition for `/var` to prevent the root partition from filling up with system logs, email, and print files. You might also want to create a separate partition for your user's home directories (`/home`) if you have a large number of users.

You also need to create a swap partition. A swap partition is used for virtual memory to hold data that is too large to fit into system RAM. Your swap partition should be at least 32 MB or two times your system's RAM, whichever is larger.

Disk Druid gathers information about the hard drives on your system and displays a list of disk drives in the lower part of the screen and the current partition information for one of the drives in the Partitions area in the upper part. For each partition, Disk Druid shows seven fields:

- Device refers to the partition's device name. For example, hda1 is the first partition on the first IDE drive.
- Mount Point/RAID/Volume indicates the directory where the partition will be mounted. For example, if you have only one partition for the entire Linux file system, the mount point is the root directory (/). For the swap partition, this field shows <Swap>. If this field appears as <not set>, you have to specify a mount point. To do so, select the partition and click the Edit button.
- The Type field shows the partition's file type, such as ext3, swap, or DOS.
- The Format field shows a check mark if the partition will be formatted and is blank if the partition will not be formatted.
- The Size field shows the amount of disk space the partition is using.
- Start and End are the beginning and ending cylinders on the hard drive used by the partition.

If there are no partitions defined, the table in the Partitions list is empty. You have to add new partitions by clicking the Add button.

You perform specific disk setup tasks in Disk Druid through the six buttons that run across the middle of the screen. The buttons perform the following actions:

- New enables you to create a new partition, assuming that there is enough free disk space available. When you click this button, another dialog box appears in which you can fill in information necessary to create a partition.
- Edit enables you to alter the attributes of the partition currently highlighted in the Partitions list. You make changes to the current attribute in another dialog box that appears when you click the Edit button.
- Delete removes the partition currently highlighted in the Partitions list.
- Reset causes Disk Druid to ignore any changes that you may have made.
- RAID sets up a redundant array of independent disks (RAID) device — a technique that combines multiple disks to improve reliability and data transfer rates. There are several types of RAID configurations. This button is active only if your system has the hardware necessary to support a RAID device.

NOTE The reference to RAID in this section is for a software RAID configuration.

- LVM sets up a logical volume. Before you can use this feature, you need to set up your partition's type as a physical volume. Then choosing this option lets you create a logical volume to make managing the physical disks easier.

Exactly what you do in Disk Druid depends on the hard drives in your PC and the partitions they already have. For this discussion, we assume that you are using the entire hard drive space for the Red Hat installation.

Setting Up the Partitions

To prepare a new hard drive to install Red Hat Linux, you have to perform the following steps in Disk Druid.

Create at least two partitions, one for the Linux file system and a swap partition. To do this, click the New button on the Disk Druid screen. You will see a dialog box (see Figure 3-9) containing the following fields, which you need to fill in:

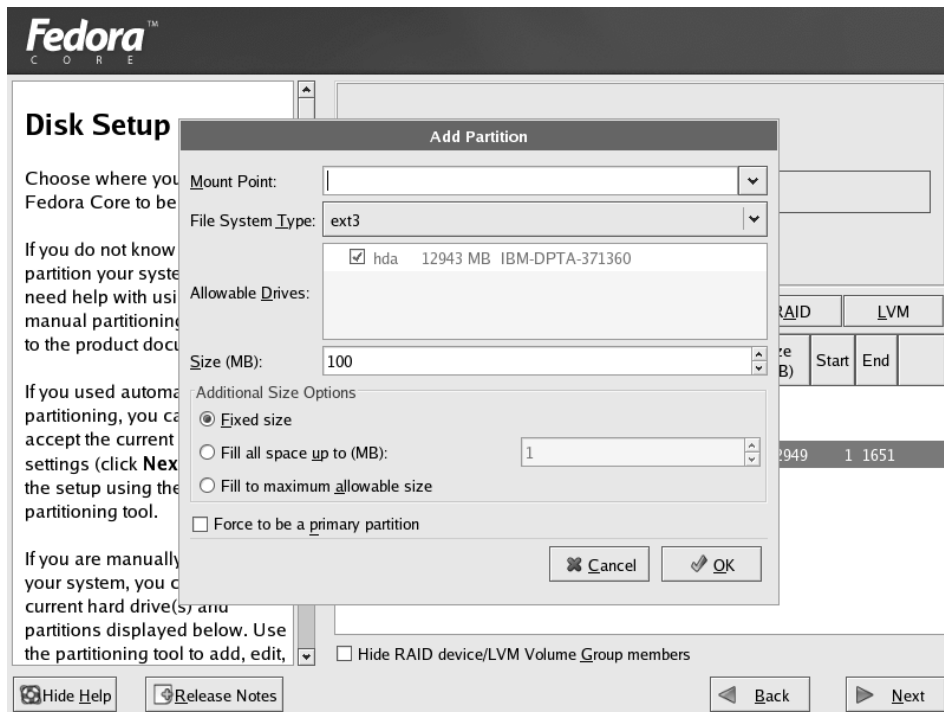


Figure 3-9 The Add Partition dialog box where you set the attributes for the new partition.

- Mount Point shows the place in the file system where the partition will be mounted. You can enter a mount point or click the down arrow to open the drop-down menu and choose a mount point from the list.
- File System Type is the type of file system you want the partition to be. Click the down arrow to open the drop-down menu and choose the file system type. The most common choices here are ext2, ext3, and swap. If you decide to use logical volumes you can choose Physical Volume (LVM) here. Then you will need to further configure the logical volumes by clicking the LVM button.
- Allowable Drives shows the installed hard drives. A check mark in the box means that the drive is available for creating partitions. Click to highlight the drive you want to use for the partitions.
- Size (MB) is the field where you can set the size of the partition. The default size is 100 MB, but you can set the size to whatever you desire.
- Additional Size Options lets you set other size restrictions by clicking the radio button. Fixed Size is the default. If you choose Fill All Space Up To, you need to enter a number in the field provided. Fill to Maximum Allowable Size will use all remaining space on the drive.
- Force to Be a Primary Partition lets you set the drive as either a primary or logical partition. Primary partitions are the first four partitions on the hard drive.
- Check for Bad Blocks, if checked, will cause the installation program to do a physical scan of the drive to find and mark any blocks that may be bad. This prevents the system from attempting to write data to such areas of the disk. Keep in mind that checking this box will slow down the installation process, depending on the size of your hard drive.

After you have filled in the field with your choices, click OK to return to the Disk Setup screen.

Create another new partition and set it as a Linux swap space. To do this, click the New button in the Disk Setup screen (refer to Figure 3-7). In the dialog box shown in Figure 3-9, enter the size of the partition. Click the list of partition types and select Linux Swap as the type. When you do so, <Swap Partition> appears in the Mount Point field. Enter the size of the swap partition and any other parameters you desire. Next, click OK to define the new partition and return to the Disk Druid screen.

After you finish specifying the partitions in Disk Druid, the Disk Setup screen will display the details: partition number, file type, size, and starting and ending cylinders. A check mark in the Format column indicates that the partition will be formatted. If you wish to change the mount point of the partition, highlight it and click the Edit button. If you want to make any other

changes to the partition, you will have to delete it and create a new one. If you are satisfied with your selections, click Next.

NOTE If you have multiple disk partitions mounted on different directories of the Linux file system and you are upgrading an existing installation, you do not have to format any partitions in which you want to preserve existing data. For example, if you have all user directories on a separate disk partition mounted on the `/home` directory, you do not have to format that partition.

You have now completed the disk preparation phase of the installation. The installation program performs the actual formatting of the partitions after it asks for some more configuration information, including the packages you want to install.

Configuring the Installation

After you prepare the disk partitions with Disk Druid and specify which partitions to format, the installation program moves on to some configuration steps. The typical configuration steps are as follows:

- Install GRUB
- Configure the network
- Set the time zone
- Set the root password and add user accounts
- Configure password authentication

The following sections guide you through each of these configuration steps.

Installing the Boot Loader

The Red Hat installation program displays the Boot Loader Configuration screen (see Figure 3-10), which asks you where you want to install the boot loader. A boot loader is a program that resides on your disk and starts Red Hat Enterprise Linux or Fedora Core from the hard disk. Red Hat provides *GRUB* (*Grand Unified Bootloader*) as the default boot loader.

The GRUB boot loader is selected as the default choice on this screen. Clicking the Change Boot Loader button lets you choose not to install any boot loader.

CAUTION If you choose not to install the boot loader, you will need to create a boot disk. Otherwise you can't start Red Hat Linux.

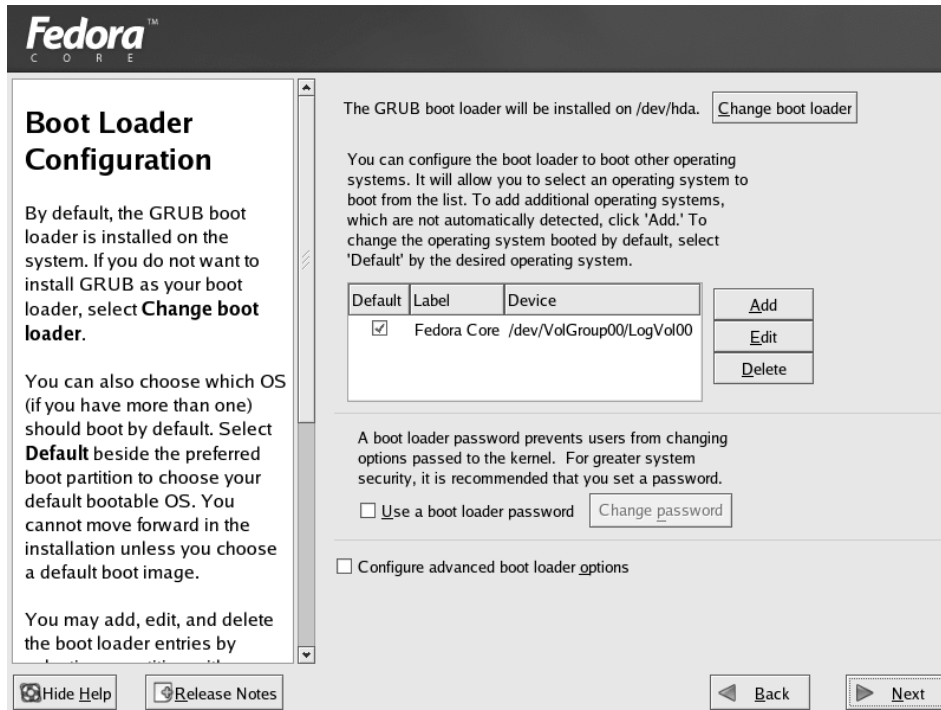


Figure 3-10 The Boot Loader Configuration screen enables you to specify which operating systems to boot.

In the center of the Boot Loader screen is a box that shows any operating systems, if there are any, that were detected on your system. The Red Hat Linux system will be shown as the default system. If you want to boot another system as the default, click the box in front of the other operating system's name. There are three options available:

- **Add** — Use this if an operating system was not detected and you wish to add it manually.
- **Edit** — Click this option if you want to change the name of the operating system.
- **Delete** — Use this option to remove an operating system from the list that you've highlighted.

The next item on this screen gives you the option to use a boot loader password. Using a boot loader password prevents someone from entering kernel commands, which could affect system security or operation. If your servers are not accessible to anyone except you or other trusted administrators, you may not need a boot loader password, but if you want good security for your system, using a boot loader password is a good idea. To enable this feature,

click the Use a Boot Loader Password check box, click the Change Password button, and enter the password.

The Configure advanced bootloader options on this page give you the opportunity of choosing the location of the boot loader and passing additional parameters to the kernel. If you want to do either of these, click the option and then click Next. The Advanced Boot Loader Configuration screen, shown in Figure 3-11, will appear.

The choices for the boot loader location are as follows:

- Master Boot Record (MBR), which is located in the first sector of your PC's hard disk
- First sector of the boot partition where you loaded Red Hat Linux

You should install the boot loader in the MBR unless you are using another operating system loader such as System Commander or OS/2 Boot Manager. You can also change the drive order by clicking the Change Drive Order button. This is only necessary if you want your drives to appear in a different order or if they are not listed in the correct order. As shown on the screen following the option Force LBA32, this option is usually not required and is necessary only if you put the boot loader beyond the first 1024 cylinders on your hard drive.

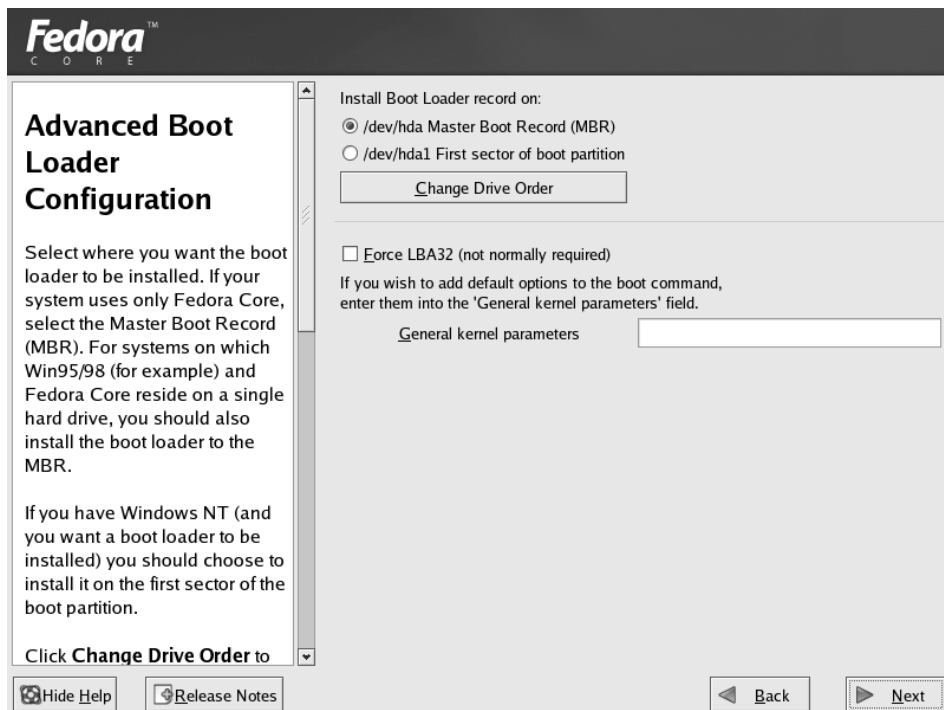


Figure 3-11 The Advanced Boot Loader Configuration screen specifies where you can change the location of the boot loader and enter kernel parameters.

The screen includes a General Kernel Parameters text field that enables you to enter any special options that Red Hat Linux may need as it boots. Your need for special options depends on what hardware you have.

The remainder of the Boot Loader Configuration screen gives you the option to select the disk partition from which you want to boot the PC. A table then lists the Linux partition and any other partitions that may contain another operating system. If your system has a Linux partition and a DOS partition (that actually has Windows 95/98 installed on it), the table shows both of these entries. Each entry in that table is an operating system that the boot loader can boot.

After you install the boot loader, whenever your PC boots from the hard disk, the boot loader runs and displays a screen showing the operating systems that you can boot. The default selection will be highlighted and will boot automatically after a few seconds. You may move the highlight bar to the name of another operating system to boot. (The Boot label column in the table in the center section of Figure 3-10 shows the names you may enter at the boot loader prompt.)

When booting the PC, if you enter nothing at the boot loader screen, the boot loader waits for a few seconds and boots the default operating system. The default operating system is the one with a check mark in the Default column in Figure 3-10. In this case, Red Hat Linux is the default operating system.

All of the instructions in this section are for your information if you choose to change any of the default settings. You can essentially accept the default selections on this screen and click the Next button to proceed to the next configuration step.

Configuring the Network

If the Linux kernel detects a network card, the Red Hat installation program displays the Network Configuration screen (see Figure 3-12), which enables you to configure the LAN parameters for your Linux system.

This step is not for configuring dial-up networking. You need to perform this step if your Linux system is connected to a TCP/IP LAN through an Ethernet card.

TIP If the Red Hat installation program does not detect your network card and you have a network card installed on the PC, you should restart the installation and type **expert** at the boot prompt. Then you can manually select your network card.

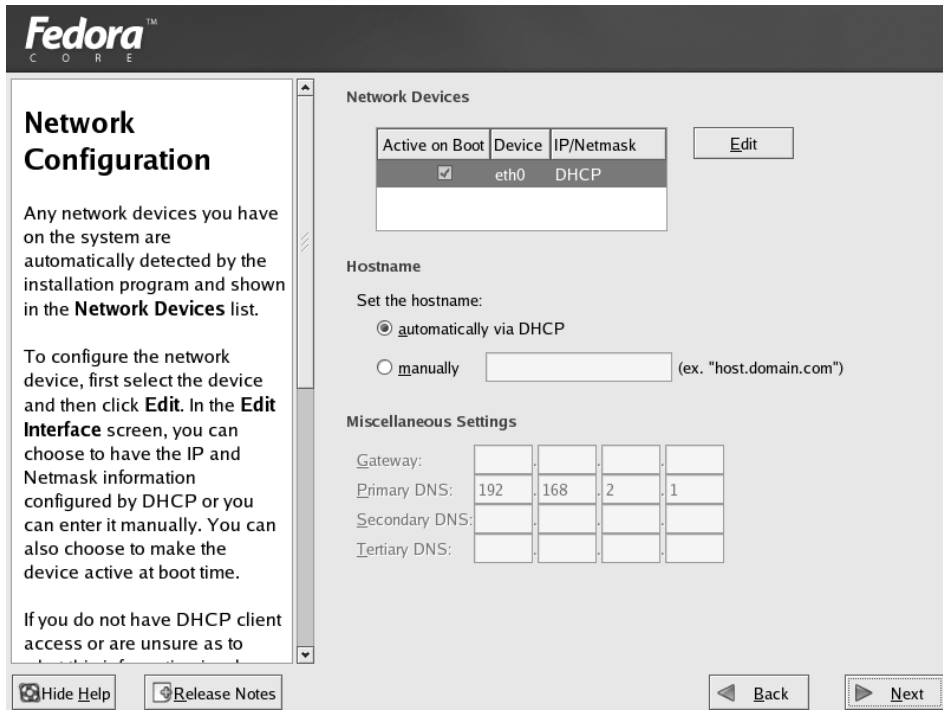


Figure 3-12 The Network Configuration screen enables you to configure the local area network.

The Network Configuration screen (see Figure 3-12) displays a list of the network card(s) installed on your system and detected by the Linux kernel. The network cards are labeled `eth0`, `eth1`, and so on. If your system has only one Ethernet card, you see only `eth0`. Figure 3-12 shows that only one network card has been detected. The default selections for the network card are Active on Boot and Configure Using DHCP. If you want to enter an IP address manually for your network card or disable the card on boot, click the Edit button to open the Edit Interface dialog box.

To disable DHCP remove the check mark from the box and enter an IP address and net mask into the appropriate boxes. To enable DHCP, click the option to place the check mark there.

To disable the card on boot, remove the check mark from the box. To enable the card on boot, click the option to place the check mark there. Normally, you would want your primary Ethernet interface to be configured on system boot.

The Hostname section of the Network Configuration screen shown in Figure 3-12 lets you choose how your system hostname will be set. The choices are:

- **Automatically via DHCP** — This is the default setting. Your PC will obtain its IP address and other network information from a DHCP server.
- **Manually** — If you choose this option, you must provide a hostname.

Select DHCP only if a DHCP server is running on your local area network. If you choose DHCP, your network configuration is set automatically and you can skip the rest of this section. You should leave the Activate on Boot button selected so that the network is configured whenever you boot the system.

If you have disabled DHCP, you will need to enter the IP address and net mask manually for the network card by editing the card. In addition, you have to enter certain parameters for the TCP/IP configuration in the text input fields for hostname and the “Miscellaneous Settings” section shown in Figure 3-12.

The Network Configuration screen asks for the following key parameters:

- The hostname of your Linux system (for a private LAN, you can assign your own hostname without worrying about conflicting with any other existing systems on the Internet)
- IP address of the gateway (the system through which you might go to any outside network)
- IP address of the primary name server
- IP address of a secondary name server (if available)
- IP address of a tertiary name server (if available)

CROSS-REFERENCE If you have a private LAN (one that is not directly connected to the Internet), you may use an IP address from a range designated for private use. Common IP addresses for private LANs are the addresses in the range 192.168.1.1 through 192.168.1.254. Chapter 11 provides more in-depth information about TCP/IP networking and IP addresses.

After you enter the requested parameters, click Next to proceed to the next configuration step.

Configuring the Firewall

In this part of the installation process, you can choose the firewall settings for your system security. Look at Figure 3-13 as you go through this section’s configuration steps.

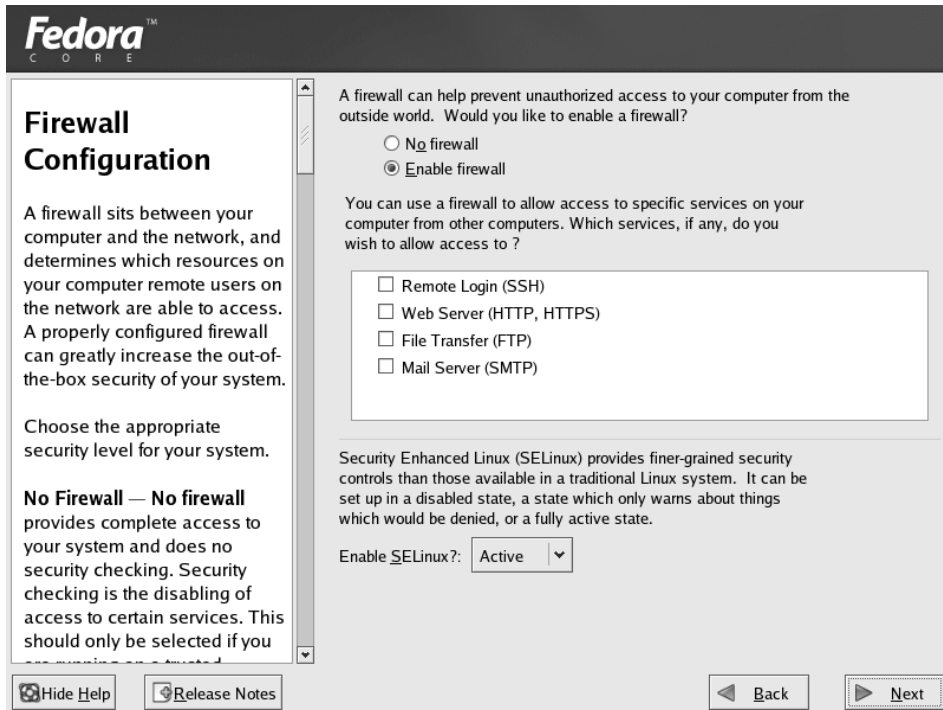


Figure 3-13 The Firewall Configuration screen lets you choose your security level.

CROSS-REFERENCE See Chapter 34 for more information about configuring a firewall.

The first choice that you make from this dialog box is whether you want to enable the firewall or to choose no firewall. By default, the installation program selects to enable the firewall for you. If you choose to enable the firewall, only connections that are in response to outbound requests are accepted. You can also select individual services that are allowed through the firewall. You can allow the following services:

- **Remote Login (SSH)** — If you allow remote access to your server through the SSH protocol, you should enable this option.
- **Web Server (HTTP, HTTPS)** — If you plan to run a Web server, you should choose this option. You do not need to choose this option to use a browser to view Web sites.
- **File Transfer (FTP)** — If you plan to run an FTP server, you should enable this option. You do not need to choose this option to retrieve files from FTP sites.

- **Mail Server (SMTP)** — If you are going to run an email server, you should enable this option. You do not need to enable this option to retrieve mail from an ISP.

If you choose the No Firewall option, all connections are allowed and no security checking is done on your system. Select No Firewall only if you have absolute faith in the security of your network.

TIP Choosing to enable the firewall is always safest, especially if you will be connecting directly to the Internet.

The final configuration step on the Firewall Configuration dialog box concerns Security-Enhanced Linux (SELinux). SELinux was developed by the National Security Agency (NSA) to provide enhanced security based on access control specified by a security policy. You can choose one of three states for SELinux:

- **Disable** — If you select this option, SELinux is not enabled on your system and there is no enforcement of a security policy.
- **Warn** — Choosing this option puts a security policy in place, but the policy is not enforced. Only warnings about possible security violations are noted. If you plan to use SELinux, this option provides a good basis for determining how the security policy would affect the operation of your system.
- **Active** — This state applies full enforcement of the SELinux security policy. You should choose this option only if you are sure that the policy will not affect your system operation.

CROSS-REFERENCE See Chapter 33 for more information about SELinux.

TIP You can read more about SELinux by visiting the NSA Web site at www.nsa.gov/selinux.

After you make your configuration choices, click Next to continue.

Choosing Additional Languages

The Additional Language Support screen, shown in Figure 3-14 is where you select the default language to be used on your system.

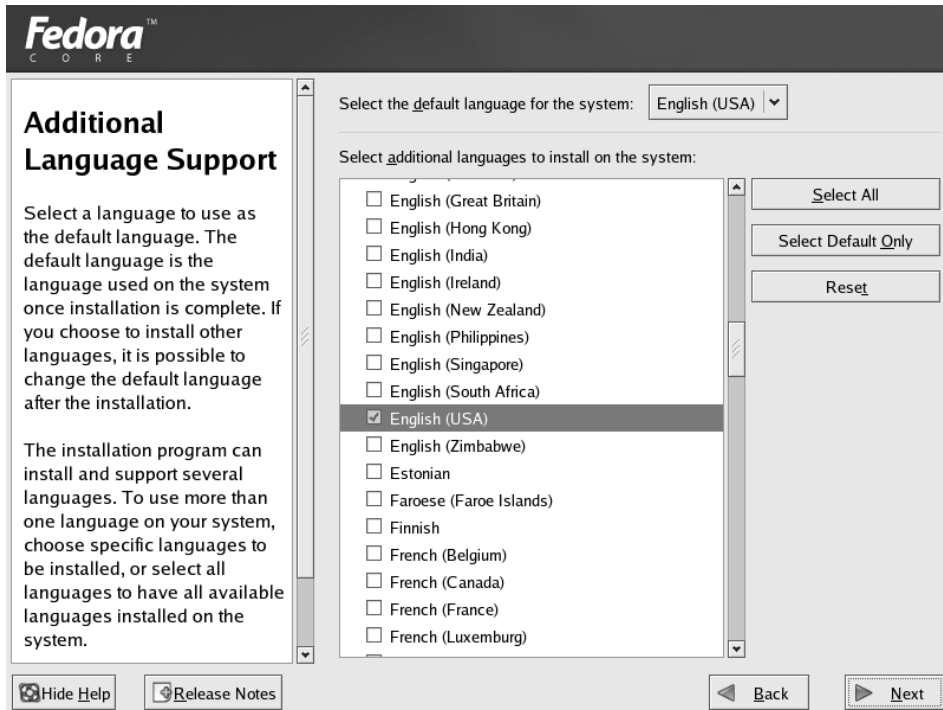


Figure 3-14 On the Additional Language Support screen you set the default language for your system as well as additional languages you may use.

The language you chose to use for system installation earlier in the installation process will be shown as the default language. If you desire to use other languages as well, you can select them from the list. Select as many other languages as you desire. Note that installing additional languages consumes storage space on your disk, so install only the languages you plan to use. After you make your selections, click Next to continue.

Setting the Time Zone

After completing the default and additional language selection, you have to select the *time zone* — the difference between your local time and *Greenwich Mean Time* (GMT) or UTC (the current time in Greenwich, England), which was selected by the International Telecommunication Union (ITU) as a standard abbreviation for Coordinated Universal Time. If you had systems in many different time zones, you might want to choose UTC for all your locations to keep your time synchronized on all your systems. The installation program shows you the Time Zone Selection screen (see Figure 3-15) from which you can select the time zone, either in terms of a geographic location or as an offset from UTC. Figure 3-16 shows the selection of a time zone.

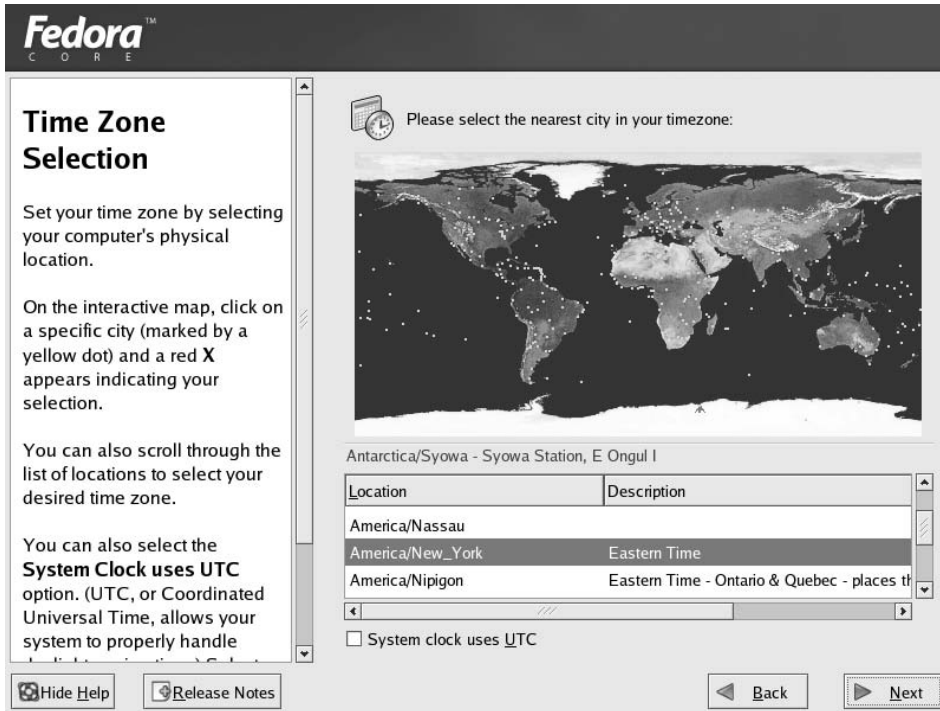


Figure 3-15 Select your time zone using the Time Zone Selection screen.

Notice that there are two tabs on the Time Zone Selection screen: Location and UTC Offset. Initially, the screen shows the Location tab, which enables you to pick a time zone by simply clicking your geographic location. As you move the mouse over the map, the currently selected location's name appears in a text field. If you want, you can also select your location from a long list of countries and regions. If you live on the east coast of the United States, for example, select USA/Eastern. Of course, the easiest way is to simply click the eastern United States on the map.

If the world view of the map is too large for you to select your location, click the View button on top of the map. A drop-down list of views appears with several choices. Click the appropriate view for your location.

The other way to set a time zone is to specify the time difference between your local time and UTC. Click the UTC Offset tab to select the time zone this way.

For example, if you live in the eastern part of the United States, select UTC-05:00 as the time zone. The -05:00 indicates that the eastern part of the U.S. is five hours behind UTC time. This tab also lets you activate Daylight Savings Time. After you select your time zone, click the Next button to proceed to the next configuration step.

Setting the Root Password

After selecting the time zone, the installation program displays the Set Root Password screen (see Figure 3-16) in which you set the root password.

The root user is the *superuser* in Linux. Because the superuser can do anything in the system, you should assign a password that only you can remember, and that others cannot guess easily. Typically, make the password at least eight characters long, include a mix of letters and numbers, and (for good measure) throw in some special characters such as + or *. Remember that the password is case-sensitive.

Type the password on the first line, and then reenter the password on the next line. Each character in the password appears as an asterisk (*) on the screen for security reasons. Both entries must match before the installation program accepts the password. The installation program displays a message when it accepts the root password.

NOTE You must enter the root password before you can proceed with the rest of the installation. After you do so, click Next to continue with the installation.

FedoraTM
C O R E

Set Root Password

Use the root account *only* for administration. Once the installation has been completed, create a non-root account for your general use and `su -` to gain root access when you need to fix something quickly. These basic rules minimize the chances of a typo or incorrect command doing damage to your system.

The root account is used for administering the system.
Enter a password for the root user.

Root Password:

Confirm:

Hide Help Release Notes Back Next

Figure 3-16 Setting the root password.

Selecting the Package Groups to Install

After you complete the key configuration steps, the installation program displays a screen from which you can select the Red Hat Linux package groups that you want to install. After you select the package groups and click Next, take a coffee break while the Red Hat installation program formats the disk partitions and copies all selected files to those partitions.

NOTE If you selected custom installation as your install type, you will see the screen shown in Figure 3-18. If you chose any other installation type, you will see a screen listing the most commonly installed packages for the installation type you chose. You can accept the default on that page or you can select Customize software packages to be installed option to pick your own packages, which will then show you the screen in Figure 3-17.

CROSS-REFERENCE Red Hat uses special files called *packages* to bundle files that make up specific software. For example, all configuration files, documentation, and binary files for the Perl programming language come in a Red Hat package. You use a special program called *Red Hat Package Manager (RPM)* to install, uninstall, and get information about packages. Chapter 30 shows you how to use RPM. For now, just remember that a package group is made up of several Red Hat packages.

Figure 3-17 shows the Package Group Selection screen with the list of package groups that you can elect to install. An icon, a descriptive label, and a radio button for enabling or disabling identify each package group.

Some of the components are already selected, as indicated by the checked boxes. This is the minimal set of packages that Red Hat recommends for installation for the class of installation (Personal Desktop, Workstation, Server, or Custom) you have chosen. You can, however, install any or all of the components. Scroll up and down the list and click the mouse on an entry to select or deselect that package group.

TIP In an actual Red Hat Linux installation, you install exactly those package groups that you need. Each package group requires specific packages to run. The Red Hat installation program automatically checks for any package dependencies and shows you a list of packages that are required but that you have not selected. In this case, you should install the required packages. Install only those packages that you think you will need immediately after starting the system. Installing too many packages could expose your system to security risks. You can always add packages later.

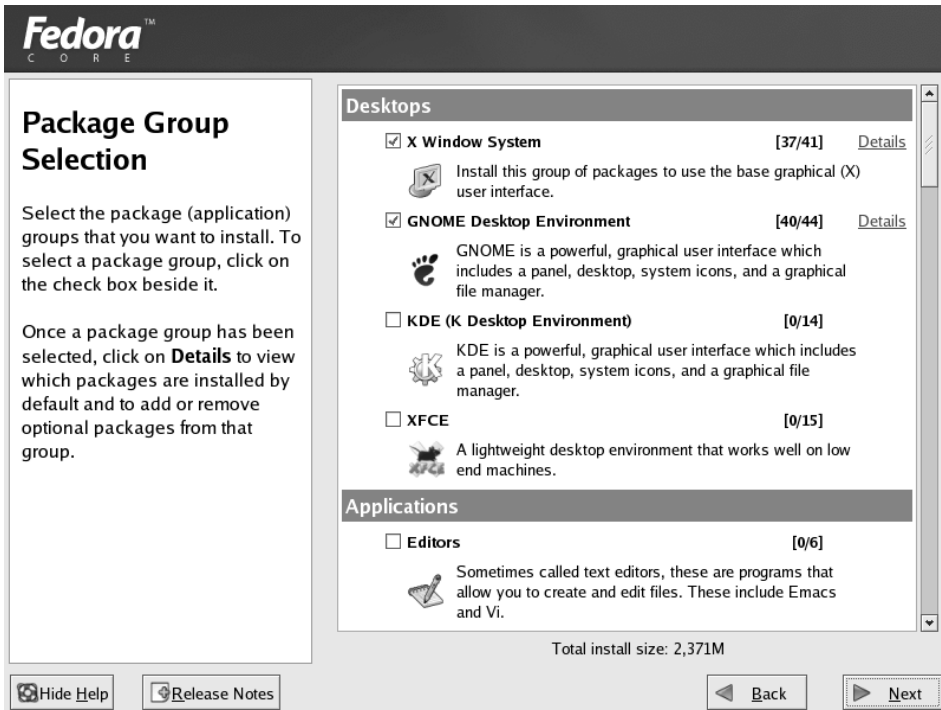


Figure 3-17 GUI screen from which you select the components to install.

Because each package group is a collection of many different Red Hat packages, the installation program also gives you the option of selecting individual packages. If you click the Customize software packages option, which appears on the Personal Desktop, Workstation, and Server package screens and then click Next, the installation program takes you to other screens where you can select individual packages. If you are installing Red Hat Enterprise Linux for the first time, you really do not need to go down to this level of detail to install specific packages.

Notice to the right of each group name there are two numbers separated by a slash. For instance, next to the X Window System is 37 / 41. This means that 37 of the 41 packages in this group have been selected for installation. To the right of the numbers is a link labeled Details. Clicking this link opens a new screen that lists the packages that are in the selected group. You can select or deselect packages as desired.

After you select the groups you want, click Next to continue with the rest of the installation. The installation program now presents the About to Install screen, as shown in Figure 3-18. This screen tells you which disks are required for the installation.



Figure 3-18 The About to Install screen gives you one last chance to cancel the installation process.

If you are absolutely sure that everything is correct and you are ready to proceed, click Continue to begin the installation. The time required for installation depends on the number of packages you chose to install. This would be a good time to take a break, but remember to check the installation's progress occasionally as you will need to change CDs. A screen showing the installation progress is displayed to show you how the installation is proceeding.

CROSS-REFERENCE You can always install additional packages later with the RPM utility program, described in Chapter 30.

Running Firstboot

After the installation process completes, you are prompted to remove all disks and to reboot your system. A program called Firstboot runs the first time the system boots after the installation, as shown in Figure 3-19.



Figure 3-19 The Firstboot program runs to let you do additional system configuration.

Shown on the left side of the Firstboot Welcome screen are the steps you must complete to boot into your newly installed system. Proceed as follows:

1. Click Next to continue to the License Agreement screen, as shown in Figure 3-20.

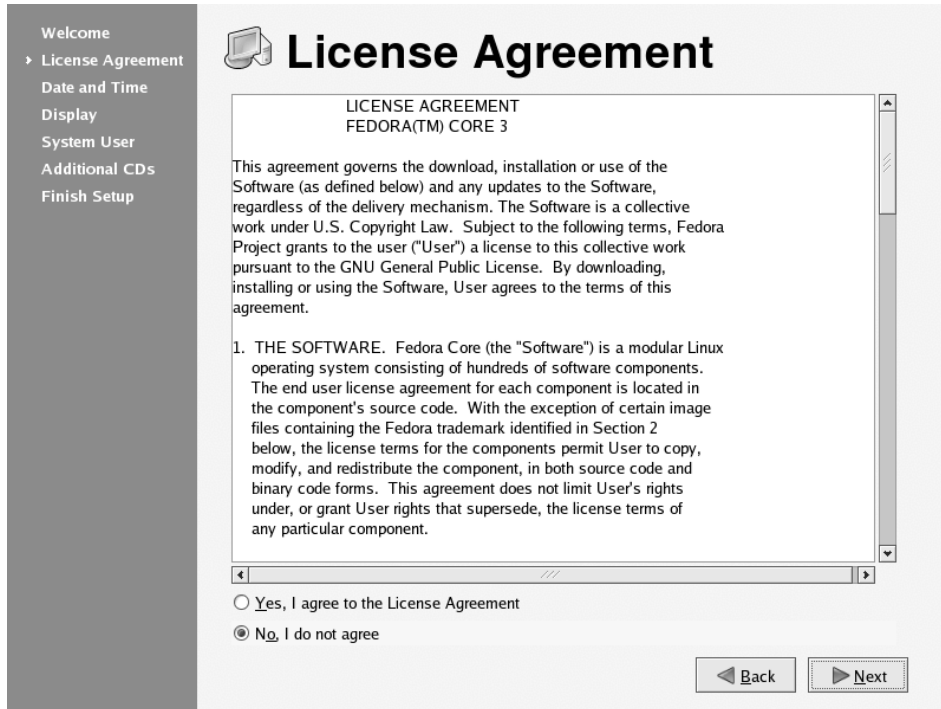


Figure 3-20 The License Agreement screen.

2. Click the radio button in front of Yes to accept the License Agreement.
3. Click Next to continue to the Date and Time screen, as shown in Figure 3-21.

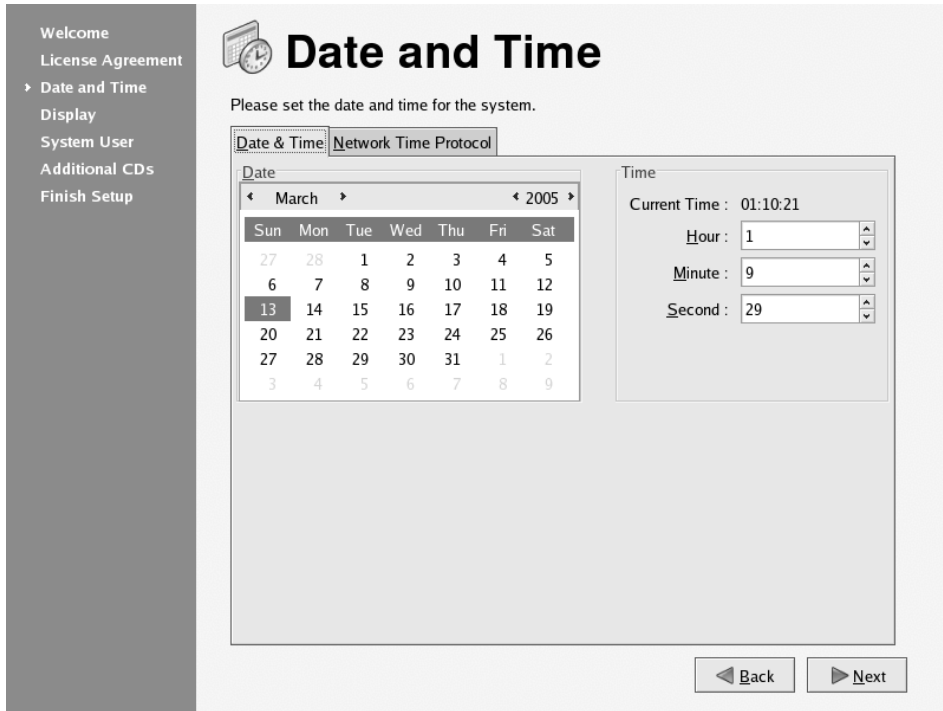


Figure 3-21 You can verify or change the system time on this screen.

4. Click Next to continue to the Display screen, as shown in Figure 3-22.

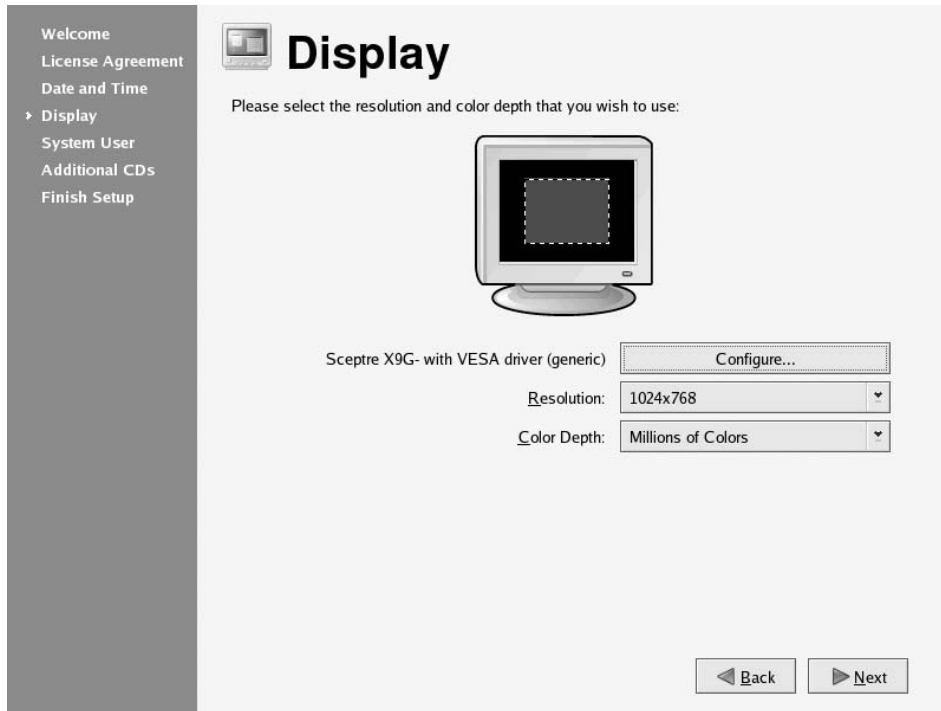


Figure 3-22 The Display screen is where you can configure your screen resolution and color depth.

5. Click the button next to the parameter you wish to change and select the new parameter from the drop-down list. When you are satisfied with the settings, click Next to continue to the System User screen, as shown in Figure 3-23.

On this screen you need to create a system user. The system user is a regular, nonroot user with ordinary, not superuser access rights. After you have filled in the appropriate information into the text boxes, click Next to continue to the Additional CDs screen, as shown in Figure 3-24.

6. If you have any additional CDs or documentation disks from Red Hat you can install them by clicking Install and selecting what you want to install. Be sure to put the appropriate CD into your CD drive. When you are finished, click Next.
7. When the Finish Setup screen appears click Finish, and your newly installed system will boot to the GNOME desktop.

Welcome
License Agreement
Date and Time
Display
➤ System User
Additional CDs
Finish Setup

System User

It is recommended that you create a system 'username' for regular (non-administrative) use of your system. To create a system 'username,' please provide the information requested below.

Username:

Full Name:

Password:

Confirm Password:

If you need to use network authentication such as Kerberos or NIS, please click the Use Network Login button.

Figure 3-23 You add a system user on the System User screen.



Figure 3-24 You can install additional programs from Red Hat provided CDs here.

Summary

In this chapter you learned how to install Red Hat Enterprise Linux and Fedora Core. You began by examining the hardware on your system and making a list of the components. You checked for hardware compatibility by referring to the Red Hat Web site. You learned how to partition your hard drive, and you chose the type of system you wanted to install. You chose the packages you wanted to install, and you began the process of installing them on your system. Finally, you rebooted your system and ran the Firstboot program to finish the installation.

Kickstart Installation

IN THIS CHAPTER

- Using the Kickstart Configurator
- Starting the Kickstart Installation

As a system administrator, one of your jobs is installing and configuring Red Hat on other computers. This could be time-consuming if you have many servers and clients to administer. To make your job easier, a program is available that automates the Red Hat installation process. This program is called Kickstart. With Kickstart you can create and store configuration files for your server or client installations and then use these files to perform your installations and upgrades. Installations and upgrades can be done from a local CD-ROM or using NFS, FTP, Hypertext Transfer Protocol (HTTP), or a hard drive. If you are installing across the network, you need to have a Dynamic Host Configuration Protocol (DHCP) server for each network segment.

Using the Kickstart Configurator

Fedora Core has a graphical tool, called Kickstart Configurator, that you can use to configure the Kickstart configuration file. The Kickstart configuration file is used to give the installation program the answers to the configuration steps that usually require user intervention. So by using Kickstart, you can automate the installation process. But before you can use the Kickstart Configurator, you need to install it, since it isn't installed by default.

Installing the Kickstart Configurator

To install the Kickstart Configurator program in GNOME follow these steps.

1. From the desktop, choose Desktop ⇨ System Settings ⇨ Add/Remove Applications to open the Package Management tool dialog box, as shown in Figure 4-1.
2. Scroll down to the System section, and click Details to the right of Administration Tools to open the Administration Tools Package Details dialog box, as shown in Figure 4-2.
3. Click the check box in front of system-config-kickstart.
4. Click Close and then Update on the Package Management tool.
5. When the Completed System Preparation screen appears, click Continue. When prompted, put installation disk one into the CD drive and Click OK.
6. When the Update Complete dialog box appears click OK, then click Quit to close the Package Management tool.

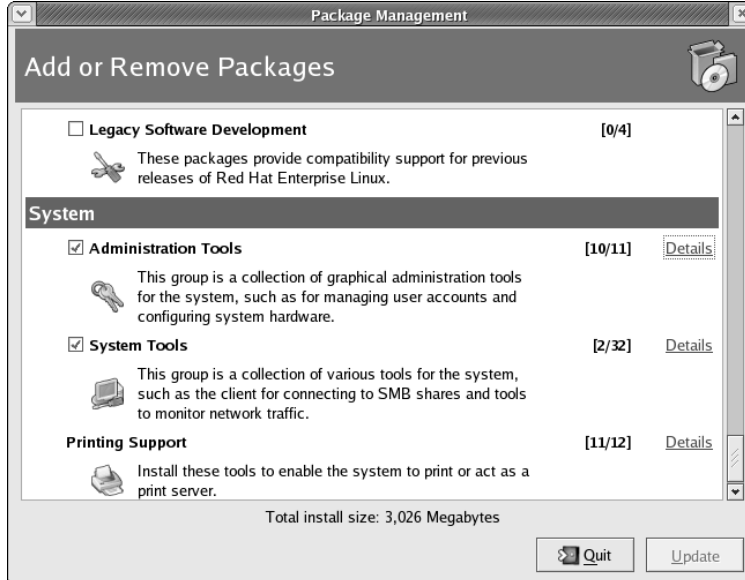


Figure 4-1 The Package Management tool is used to install packages from the installation CDs.

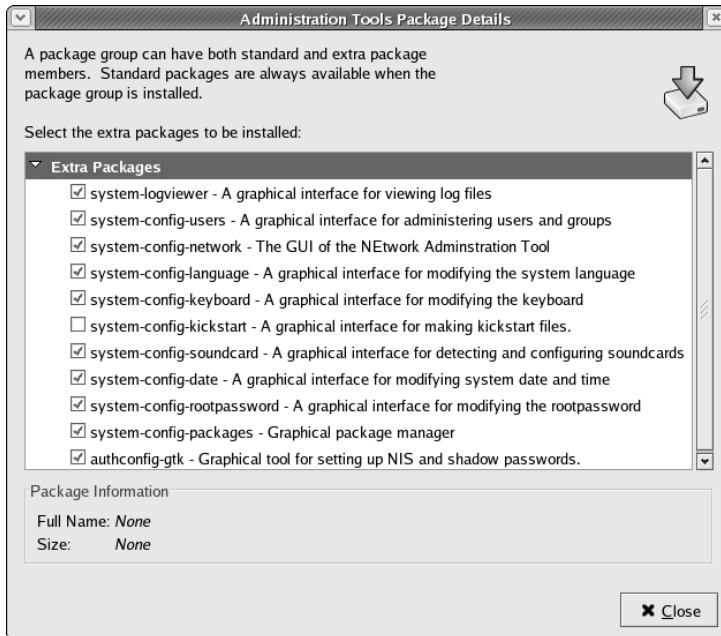


Figure 4-2 The Administration Tools Package details dialog box.

You are now ready to start the Kickstart Configurator. Follow these steps:

1. Choose Applications ⇄ System Tools ⇄ Kickstart. The Kickstart Configurator window shown in Figure 4-3 appears.

When the Kickstart Configurator window opens, the Basic Configuration dialog box is shown on the right side. You will notice that many of the text fields are already filled in. This is because the program has loaded the `anaconda-ks.cfg` file that was created during your system installation.

NOTE The `anaconda-ks.cfg` file you see will be based on the type of installation you did. You should keep in mind that there may be significant differences between a server configuration file and a workstation or desktop configuration file, although having a file to refer to is always a good place to start.

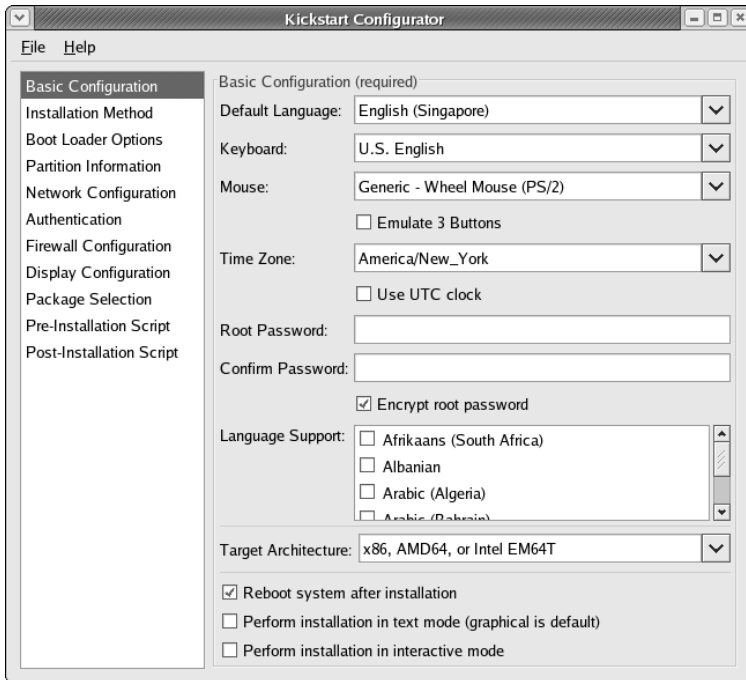


Figure 4-3 The Kickstart Configurator program opens to the Basic Configuration dialog box.

On the left side of the window is a list showing the other configuration dialog boxes. Clicking an item in the list will open its respective dialog box on the right side of the Kickstart Configurator window. Beginning with the basic configuration, the fields are:

- **Language** — Click the down arrow on the right of the field, and click the language you want to use for the installation.
- **Keyboard** — Click the down arrow on the right of the field, and click the language appropriate for your keyboard.
- **Mouse** — Click the down arrow on the right of the field, and click the appropriate mouse for your system. If you have a two-button mouse and want to emulate a three-button mouse, check the box to emulate three buttons.
- **Time zone** — Click the down arrow on the right of the field, and click the appropriate time zone for your location. Click the Use UTC Coordinated Universal Time clock check box if you want your time to be set to UTC (Coordinated Universal Time). UTC was previously known as GMT, or Greenwich Mean Time.

- **Root password** — Enter the password for the root user. Notice that it will be encrypted by default unless you remove the check from the box.
 - **Language support** — Choose additional languages you want to install on your system by checking the box in front of each language's name.
 - **Target architecture** — This field lets you choose the type of system on which you are installing. For example, choose x86 for a typical Intel Pentium system.
 - **Reboot system after installation** — By default, the system reboots unless you remove the check from the box.
 - **Perform installation in text mode** — By default, the installation is performed in graphical mode unless you remove the check from the box.
 - **Perform installation in interactive mode** — Place a check in this box if you want to use interactive mode during the installation. This method still uses the Kickstart configuration file but lets you see the options selected for installation one screen at a time. You need to click Next to continue at each screen.
2. After you have verified or entered the appropriate information into the Basic Configuration dialog box, click Installation Method to open the Installation Method dialog box shown in Figure 4-4.

In the Installation Method dialog box you select whether you want to do a new installation or an upgrade of an existing system.

On the Installation Methods screen, you can pick the type of installation you will be performing. You can choose to do a new installation or an upgrade by clicking the radio button in front of your choice. If you choose to upgrade an existing system you won't have to define partitions or packages for the installation program to use because the existing partitions and packages will be used.

NOTE You will require a separate disk for each type of installation. You cannot use a disk meant for a new installation to do an upgrade, or use a disk meant to upgrade a system to do a new installation.

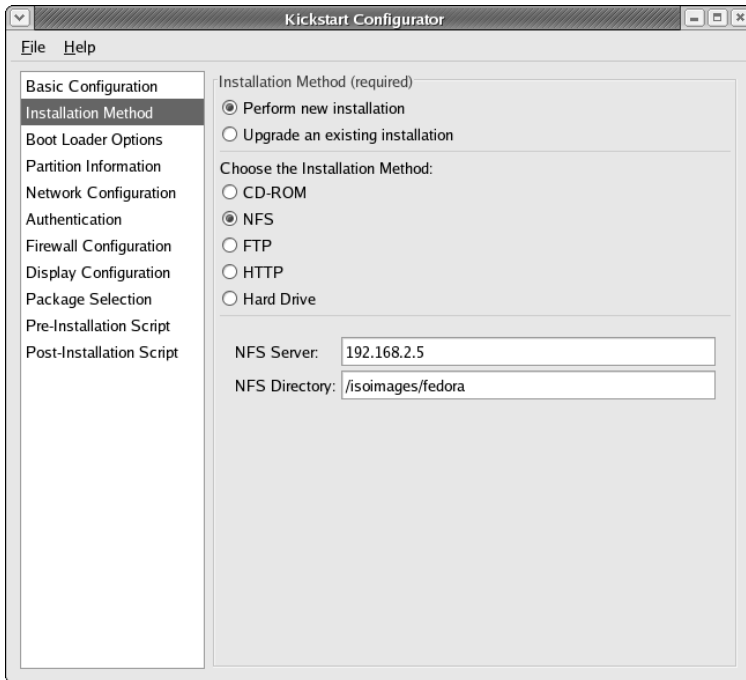


Figure 4-4 The Installation Method dialog box is where you choose how to perform the installation.

You can also choose the type of media you will be using for the installation. The default choice is CD-ROM. You also have the following choices:

- **NFS** — If you choose this method, two additional fields will appear where you need to enter the name of the NFS server and the directory to use.
 - **FTP** — If you choose this method, four additional fields will appear. You need to enter the name of the FTP server and the directory to use in two of the fields. You are also given the opportunity to show an FTP username and password by clicking the check box and entering the appropriate information.
 - **HTTP** — If you choose this method, two additional fields will appear where you need to enter the name of the HTTP server and the directory to use.
 - **Hard Drive** — If you choose this method, two additional fields will appear where you need to enter the partition of the hard drive and the directory to use.
3. When you are satisfied with your choices, click **Boot Loader Options** to open the **Boot Loader** dialog box shown in Figure 4-5.

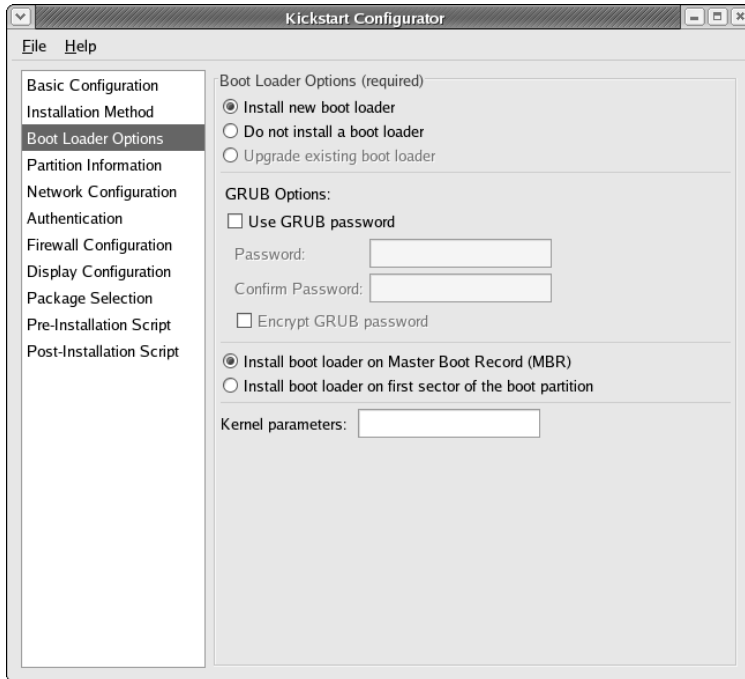


Figure 4-5 The Kickstart Configurator program window, showing the Boot Loader Options screen.

Boot Loader Options Screen

On this screen, you have several choices related to the system boot loader. You can choose to install a boot loader, not to install the boot loader, or to upgrade an existing boot loader. Just click the button in front of your choice. You can also set a password for the GRUB boot loader as well as encrypt it by clicking the appropriate check box.

You can choose the location of the boot loader by clicking the radio button in front of either Master Boot Record (MBR) or first sector of boot partition. The final field of this screen allows you to pass additional parameters to the kernel if necessary.

After making your choices, click Partition Information to open the Partition Information dialog box shown in Figure 4-6.

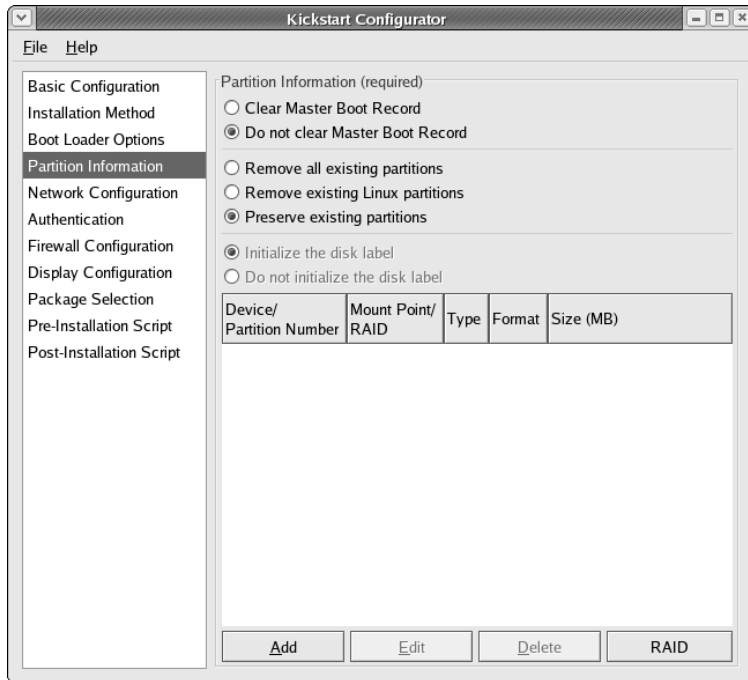


Figure 4-6 The Kickstart Configurator program window, showing the Partition Information dialog box.

Partition Information Screen

In this dialog box, you can create your disk partitions and set the mount points for your directories. By default, the master boot record (MBR) is cleared during installation. If you do not want to clear the MBR, click the Do not clear MBR radio button.

Also by default, all existing partitions are removed during the installation. If you do not want to do this, click the radio button in front of your choice. You can choose to remove only existing Linux partitions, or you can keep the existing partitions.

If you are installing on a new hard drive, you will want to keep the default setting of Initialize the disk label. If you are installing on a previously used drive and want to keep the existing drive label, check the Do not initialize the disk label radio box.

The partition window in this dialog box is currently empty, since you haven't created any partitions yet. To create a partition, click the Add button at the bottom of the screen. You see a Partition Options dialog box, as shown in Figure 4-7.

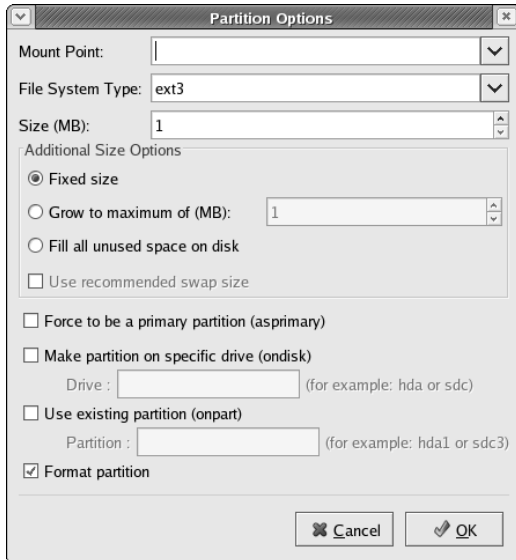


Figure 4-7 The Partition Options dialog box is where you add and configure partitions.

NOTE If you aren't sure about setting up the partitions on your system, you can use the setup shown in the `anaconda-ks.cfg` file as a guide. These partitions are the defaults selected during the Fedora Core installation and generally work well.

The Partition Options dialog box contains the following fields, which you need to fill in:

- Mount Point shows the place in the file system where the partition will be mounted. You can type a mount point or click the down arrow to open the drop-down menu and choose a mount point from the list.
- File System Type is the type of file system you want the partition to be. Click the down arrow to open the drop-down menu, and choose the file system type. The most common choices here are `ext2`, `ext3`, and `swap`.
- Allowable Drives shows the installed hard drives. A check mark in the box means that the drive is available for creating partitions. Click to highlight the drive you want to use for the partitions.
- Size (MB) is the field where you can set the size of the partition. The default size is 100 MB, but you can set the size to whatever you desire.

- Additional size options let you set other size restrictions by clicking the radio button in front of the option. Fixed size is the default. If you choose Grow to maximum of, you need to enter a number in the field to the right of the choice. Fill all unused space on disk will use all remaining space on the drive. Use recommended swap size is grayed out and not selectable unless the file system type selected is Swap.
 - Force to be a primary partition lets you set the drive as either a primary or logical partition. Primary partitions are the first four partitions on the hard drive.
 - Make partition on specific drive lets you choose the hard drive on which to create the partition. Just enter the drive identifier; for example, hda would be the first IDE drive.
 - Use an existing partition lets you specify using an already existing partition.
 - Format partition is checked as the default. If you do not want to format the partition, remove the check mark from the box.
1. After you have filled in the fields, click OK to return to the partition information screen. You will see the partition information you have chosen displayed in the partition window.
 2. You can also create software RAID partitions if you desire. To create a software RAID partition, click the RAID button at the bottom of the partition information screen. You will see the RAID options dialog box, as shown in Figure 4-8.



Figure 4-8 The RAID Options dialog box gives you information about creating RAID partitions.

As shown on the RAID Options dialog box, you need to have two software Raid partitions to be able to create a RAID device.

3. The only option you have on this dialog box is to accept the option to create a software RAID partition; click Next to continue. A Partition Options dialog box like the one shown in Figure 4-7 appears with software RAID already selected as the file system type.
4. Enter the remaining options as you do when creating a partition earlier. Click OK, and the first RAID partition is shown on the partition window. Click RAID again, and create another RAID partition.
5. After you have created the second RAID partition and have returned to the partition information window, click RAID again. Now you have the option of creating a RAID device that has already been selected for you.
6. Click OK, and you see the Make RAID Device dialog box shown in Figure 4-9.

This dialog box contains the following fields:

- Mount Point shows the place in the file system where the partition will be mounted. You can type a mount point or click the down arrow to open the drop-down menu and choose a mount point from the list.
- File System Type is the type of file system you want the partition to be. Click the down arrow to open the drop-down menu, and choose the file system type. The most common choices here are ext2, ext3, swap, and vfat.
- RAID Device shows the identifier for the RAID device. By default, md0 is selected. To change this selection, click the down arrow to the right of the field and select a different identifier.

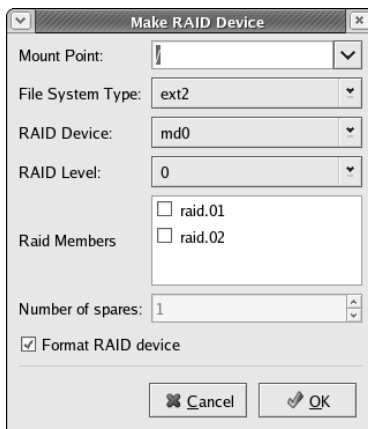


Figure 4-9 The Make RAID Device dialog box gives you choices about creating RAID devices.

- RAID Level is the field where you can choose the RAID level you want to use. The choices here are levels 0, 1, and 5. See Chapter 6 for an explanation of RAID levels.
 - RAID Members lets you select the partitions that will be a part of the RAID device. In Figure 4-9, the two partitions created previously are shown. Click the box in front of the partition to select it.
 - Number of spares lets you specify another partition to use as a spare if you are using RAID level 1 or 5. You need to create an additional software RAID partition for each spare you select.
 - Format RAID device is checked as the default. If you do not want to format the device, remove the check mark from the box.
7. After you have made your selections, click OK to create the RAID device. You should now see the device listed on the Partition Information window.
 8. Click Network Configuration from the list on the left to open the Network Configuration dialog, as shown in Figure 4-10.

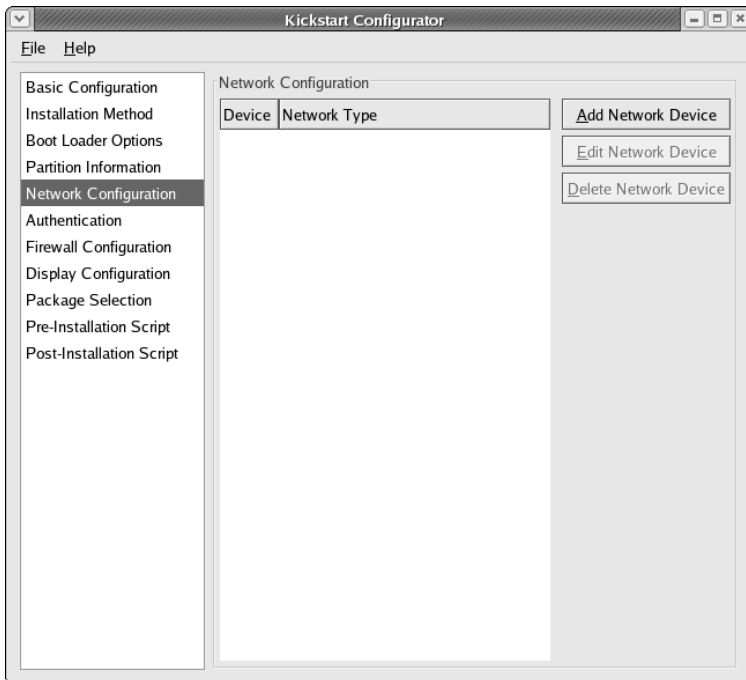


Figure 4-10 The Network Configuration dialog box is where you configure your network interface card (NIC).

Network Configuration

On this screen, you can configure a network card. Click Add Network Device, and you see the Network Device Information dialog box shown in Figure 4-11.

In this dialog box, the first network interface card is shown as eth0, and the network type is shown as DHCP. DHCP, or Dynamic Host Configuration Protocol, is used to provide an IP address, netmask, gateway address, and a DNS name server address for your system. If you are using DHCP on your network, you can accept the defaults for this device. If you want to use a static IP address, click the down arrow in the network type field and choose static IP. You need to fill in the appropriate network information in the fields at the bottom of the dialog box. Another choice shown when you click the down arrow for Network Type is BOOTP. BOOTP is a protocol that lets you boot your system across the network. For example, if you are planning to set up a terminal server with attached clients, like the Linux Terminal Server Project, you would want to configure BOOTP.

1. After you create or modify your NIC settings, click OK to return to the Network Configuration window, and your device is shown in the list. If you want to configure additional network interfaces, click Add Network Device and make your choices. To make changes to an existing device, click Edit and make your changes as described above.
2. Click Authentication from the list on the left to open the Authentication dialog box, as shown in Figure 4-12.



Figure 4-11 The Network Device Information dialog box is where you configure your network interface card.

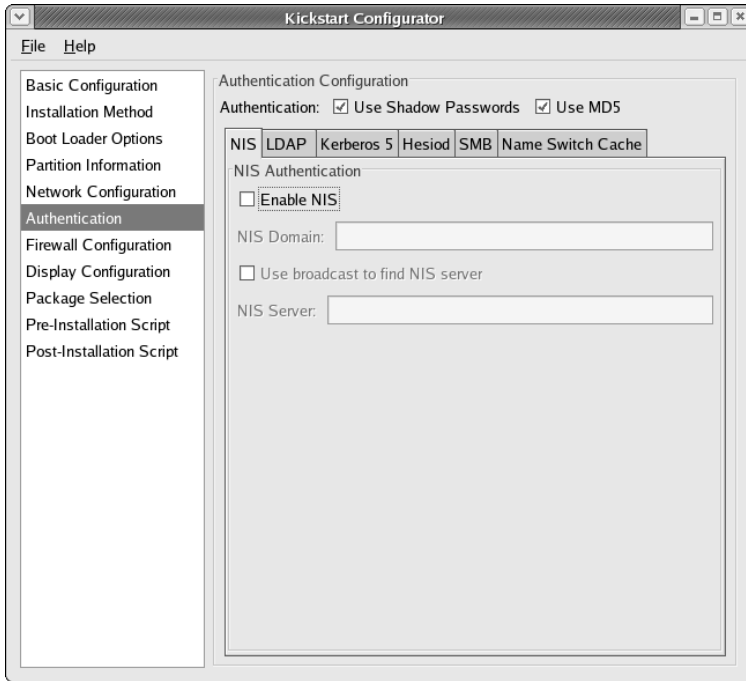


Figure 4-12 The Kickstart Configurator program window, showing the Authentication screen.

Authentication

By default, Use shadow passwords and MD5 are selected. Both of these options provide increased system security and should be used unless you have good reason not to use them. If you want to use any of the other services available from this screen, click the tab for the service you want to use. The following services are available.

- **NIS** — This acronym stands for Network Information Service (or system depending on whom you ask) and was developed by Sun Microsystems to provide a distributed database of system resources and authorized users. If you want to use NIS, click the enable NIS check box and enter the name of your NIS domain and sever information. See Chapter 13 for more details about NIS.

- **LDAP** — LDAP is an acronym for Lightweight Directory Access Protocol and is used to provide single sign-on services for users. LDAP is based on the X.500 directory standard and uses a systemwide database of resources and authorized users. To use LDAP click the Enable check box and enter your LDAP server and base names. You can learn more about LDAP in Chapter 17.
- **Kerberos 5** — Kerberos is the mythical three-headed dog that guarded the gates to the underworld. It is also the name chosen by the MIT-based group for their network authentication system. To use Kerberos click the enable check box and enter your Kerberos realm, domain controller, and master server names. You can learn more about Kerberos in Chapter 17.
- **Hesiod** — Hesiod was a poet from ancient Greece and also the name chosen by the MIT group for their network database that was based on DNS and used to provide information across the network. To use Hesiod click the Enable check box and enter the domain prefix in the LHS field and the Hesiod domain in the RHS field. For more information about Hesiod read the `hesiod.conf` man page.
- **SMB** — SMB is an acronym for Server Message Block, which is the protocol used by Microsoft for its networking implementation. If you want to use Samba, you need to install the Samba package on your system to use SMB. This option only enables SMB authentication. To enable SMB authentication, click the Enable check box and enter the names of the SMB servers and workgroup. You can learn more about SMB in Chapter 14.
- **Name Switch Cache** — To use this service, click the Enable `nscd` check box. `NSCD` is the daemon that handles requests for password and group lookups and caches this information. Caching provides faster response for future lookups. For more information about `NSCD`, read the `nscd` main page.

After you have made your choices, click Firewall Configuration on the left to open the Firewall Configuration dialog box, as shown in Figure 4-13.

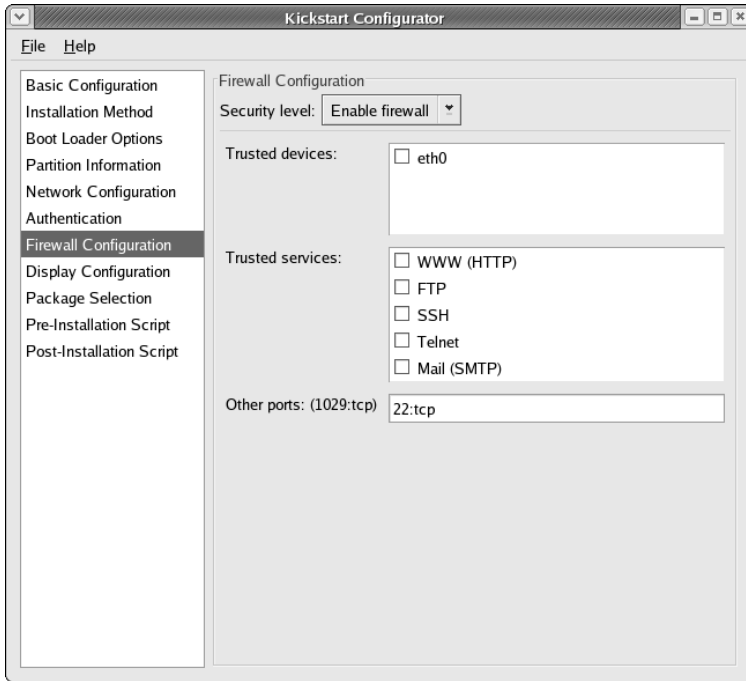


Figure 4-13 The Kickstart Configurator program window showing the Firewall Configuration dialog box.

Firewall Configuration

From this dialog box, you can set the level of security you want on your system. You can make your choices from the following fields.

- **Security Level** — Either enable or disable the firewall for your system by clicking the down arrow in the Security Level field and choosing one or the other. If the firewall is enabled, it will allow only inbound traffic in response to outbound requests. Choosing Disable firewall allows remote systems to access any services on your system that are running. Unless your system is already protected by another firewall it is probably a good idea not to disable the firewall.
- **Trusted devices** — If you configured a NIC earlier, this device is listed in this field. Click the check box for the device to allow any connections to the selected device.
- **Trusted services** — If you check the check box in front of a listed service, requests for that service are accepted by your system.

- **Other ports** — In this field you can enter specific ports to open to allow incoming traffic. The example shows 1029:tcp, which means that incoming TCP traffic on port 1029 is allowed. You can specify whatever ports you need based on the services you plan to provide. For example, if you were going to run a VNC server you would want to open ports 590x.

After you have made your choices, click Display Configuration on the left to open the Display Configuration dialog box shown in Figure 4-14.

Display Configuration

If you plan to use a graphical interface, you need to configure the X window system and your system display.

1. After the Display Configuration dialog box opens, click the Configure the X Window system check box. After you do this, the three tabs become selectable, and you need to enter the appropriate information for each tabbed screen.

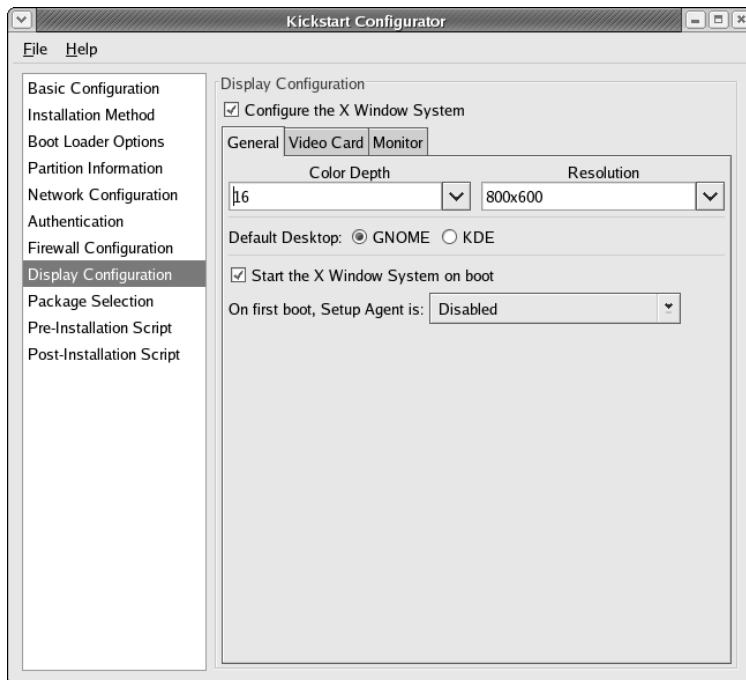


Figure 4-14 The Kickstart Configurator program window, showing the Display Configuration dialog box.

- On the General tab you can set the color depth, or number of colors your system will display. The screen resolution is the size of the screen you want displayed. Click the down arrow next to the fields and choose the desired values from the drop-down lists. Also choose the default desktop, either Gnome or KDE. Gnome is selected as the default choice. If you want the X Window system to start at system boot, click the check box. Choosing this option gives the user a graphical login window instead of a console login terminal. Finally, you can choose whether you want to run the Firstboot program the first time the system boots. Make your choice by clicking the down arrow next to the field and clicking what you want.

NOTE If you are installing the desktop or workstation versions of Fedora Core or Enterprise Linux, it is probably a good idea to start the X Window system and enable the Firstboot option. This will give your users the opportunity to easily fine-tune their displays.

- When you have finished configuring the General settings, click the Video Card tab to open the Video Card dialog box shown in Figure 4-15.

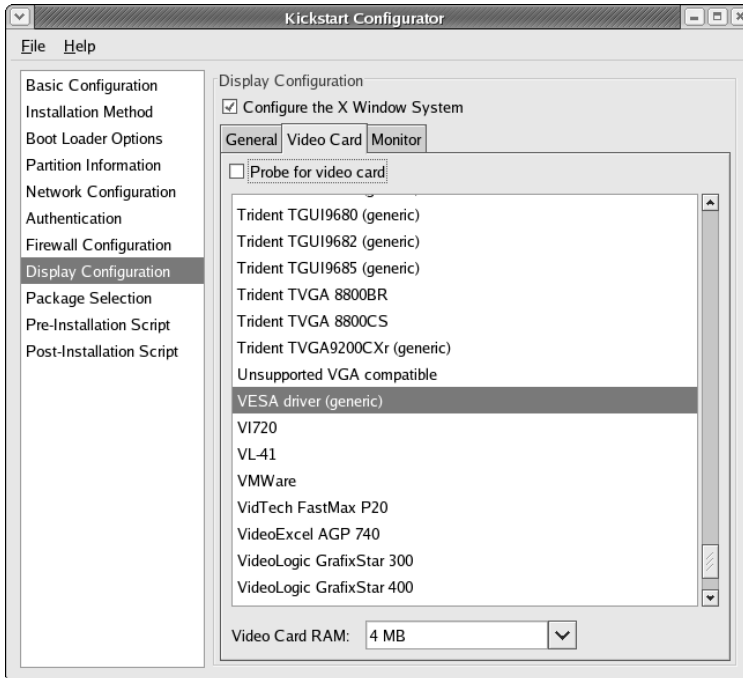


Figure 4-15 The Kickstart Configurator program window showing the Video Card dialog box.

The default selection here is to have the installation program probe for the video card installed in the system. This is nearly always a good choice for most video cards, especially cards produced within the last two or three years. You can disable this option by removing the check from the check box. If you disable probing for the video card, you will have to select the appropriate card from the list of cards and set the amount of video memory.

4. When you finish configuring your video card, click the Monitor tab to open the Monitor dialog box shown in Figure 4-16.

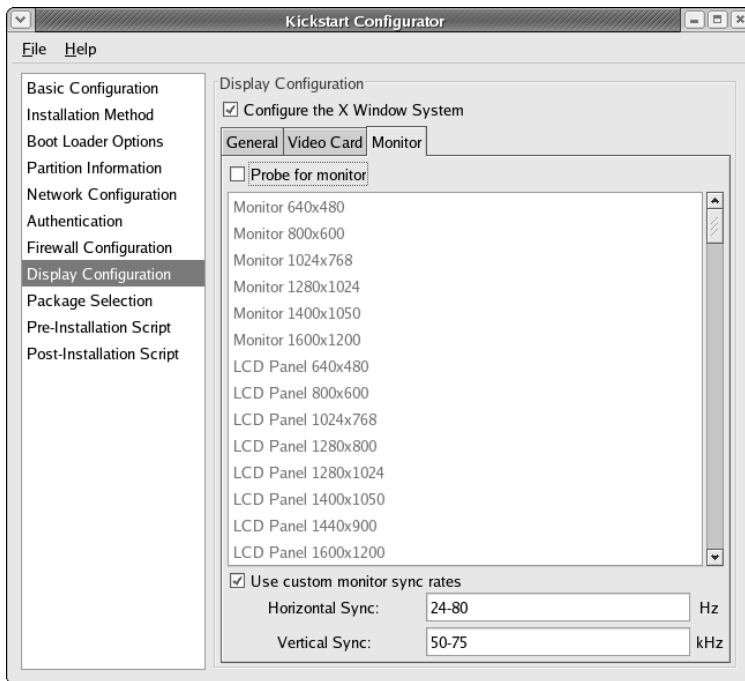


Figure 4-16 The Kickstart Configurator program window, showing the Monitor dialog box.

The default selection here is to have the installation program probe for the monitor installed in the system. This is nearly always a good choice for most monitors, especially monitors produced within the last two or three years. You can disable this option by removing the check from the check box. If you disable probing for the monitor, you will have to select the appropriate monitor from the list of cards. Rather than selecting a monitor, you may set the vertical and horizontal sync of your monitor if you know it.

5. When you have finished configuring your monitor, click Package Selection from the list on the left to open the Package Selection dialog box shown in Figure 4-17.

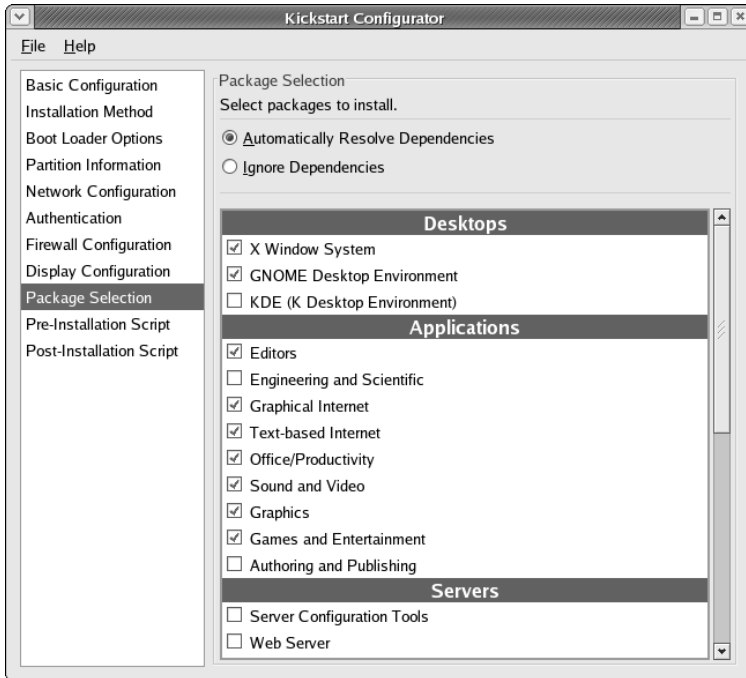


Figure 4-17 The Kickstart Configurator program window, showing the Package Selection dialog box.

Package Selection

From the Page Selection dialog box, you can select the packages you want to install on your system. Click the box in front of the package name to select it. By default, the installation automatically resolves package dependencies and installs additional packages if required. You can choose to ignore dependencies by selecting this option, and no additional packages will be installed to resolve dependency problems. There is a possibility that your selected packages may not work with unresolved dependencies, so it is best to let the installation program automatically resolve the dependency.

NOTE If you aren't sure about which packages to select to install, you can refer to the `anaconda-ks.cfg` file to see which packages were selected for installation. Remember that the packages installed are based on the type of installation you chose when you installed. For example, if you chose a workstation install, then the `anaconda-ks.cfg` file would show the packages installed for that type.

When you have finished selecting your packages, click Pre-Installation Script from the list on the left to open the Pre-Installation Script dialog box shown in Figure 4-18.

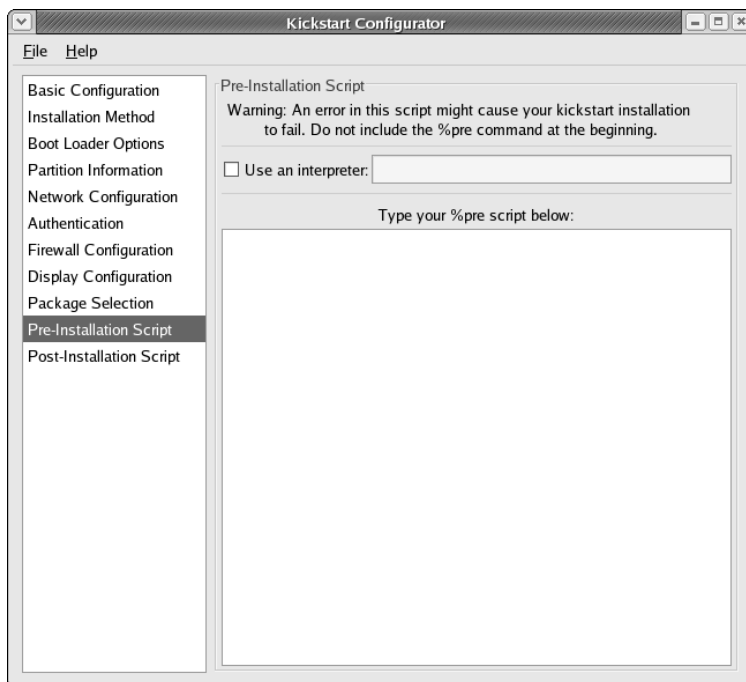


Figure 4-18 The Kickstart Configurator program window, showing the Pre-Installation Script dialog box.

Pre-Installation Script

If you want to have commands run on your system before the installation starts, you can enter them in the area indicated on the screen. The system parses the Kickstart file and then runs your commands before beginning the

installation. You can have your script interpreted by the scripting language of your choice by selecting the Use an Interpreter option and entering the interpreter to use.

After you have entered your information, if any, click Post-Installation Script from the list on the left to open the Post-Installation Script dialog box shown in Figure 4-19.

Post-Installation Script

If you want to have commands run on your system after the installation ends, you can enter them in the area indicated on the screen. If you desire, you can have the post-installation script run outside of the chroot environment by selecting this option. A chroot environment is basically a minimal root file system created within the actual root file system. The minimal root environment lets your programs run, but prevents anyone from doing any damage to your actual root file system. Also, you can have your script interpreted by the scripting language of your choice by selecting the Use an Interpreter option and entering the interpreter to use.

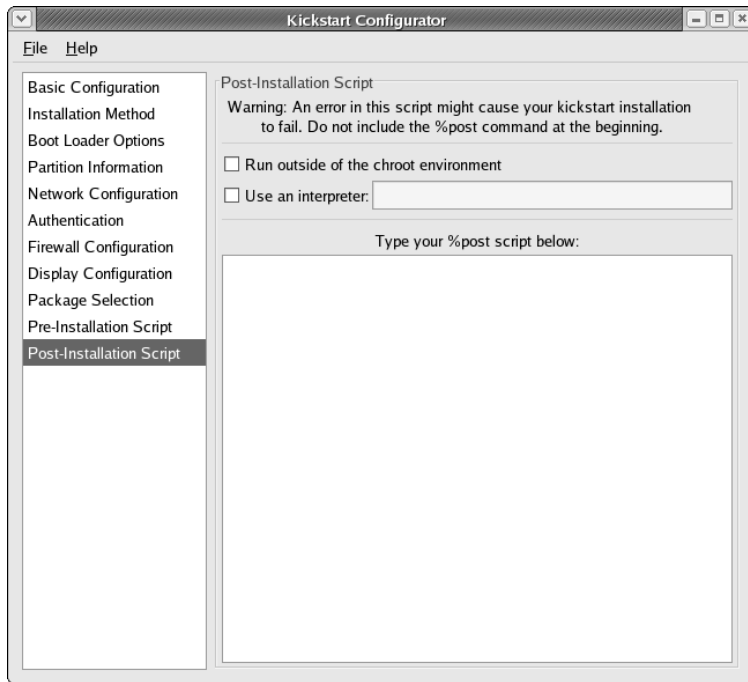


Figure 4-19 The Kickstart Configurator program window, showing the Post-Installation Script dialog box.

After you have entered your information here, you are finished configuring the Kickstart configuration file. Choose File ⇨ Save File to save all your work. Be sure to name the file `ks.cfg` if it isn't already named that in the Save File dialog box.

Starting the Kickstart Installation

After you have completed configuring your `ks.cfg` file, you are ready to use it to do your installation. But, before you can begin a Kickstart installation, there are a few additional steps that you need to do. These steps are:

- You must place a copy of the configuration file in one of three locations. The file can be placed on a bootable floppy, a bootable CD-ROM, or a network drive.
- You need to make the installation tree available. The installation tree is a copy of the Fedora Core or Red Hat Enterprise Linux CD-ROMs with the same directory structure. For a CD-ROM installation, you can put the Installation CD 1 into the CD-ROM drive. For a hard drive or network installation, you need to have the ISO images of the binary Fedora Core or Red Hat Enterprise Linux CD-ROM accessible from those locations.
- If you are planning to use a bootable floppy or CD-ROM to start the Kickstart installation, you will have to make those bootable disks. Also, if you are planning to place the `ks.cfg` file on a bootable floppy or CD-ROM, you can do it when you create the bootable floppy or CD-ROM.

NOTE You need to be logged in as the root user (or `su` to become root) to create the bootable disks.

Creating a Bootable Floppy

To create a bootable floppy, follow these instructions:

1. Place the Red Hat Linux Installation CD-ROM 1 into your CD-ROM drive.
2. The CD will be automatically mounted and will appear in the `/media/CDROM` directory. Change to the `images` directory on the CD.
3. Place a floppy disk into your floppy drive. At a terminal prompt, enter

```
dd=didiskboot.img of=/dev/fd0 bs=1440k
```

4. If you want a copy of your Kickstart configuration file, `ks.cfg`, on your bootable floppy, mount the floppy by entering the command `mount /dev/fd0 /media/floppy`; then copy the `ks.cfg` file that you created to the `/media/floppy` directory.

Creating a Bootable CD-ROM

To create a bootable CD-ROM, follow these instructions.

1. Place the Red Hat Linux Installation CD-ROM 1 into your CD-ROM drive.
2. The CD will be automatically mounted and will appear in the `/media/CDROM` directory.
3. Copy the `isolinux` directory from the CD-ROM to a directory on your hard drive. You should create a temporary directory for this purpose. I created a directory called `/root/tempdir` on my system to use for an example. Be sure to use the correct path for your system. Use the following command:

```
cp -r isolinux/ /root/tempdir/
```

4. If you want a copy of your Kickstart configuration file, `ks.cfg`, on your bootable CD-ROM, copy the `ks.cfg` file that you created to the `/isolinux` directory on your hard drive at the location where you created it.

```
cp /root/ks.cfg /tempdir/isolinux/
```

5. Change to the temporary directory, `cd /root/tempdir`, that you created.
6. Use the `chmod` command to be sure the files you copied have the correct permissions.

```
chmod u+w isolinux/*
```

7. Create the ISO image file with this command. (The command should be entered on one line.)

```
mkisofs -o file.iso -b isolinux.bin -c boot.cat -no-emul-boot \-boot-load-size 4 -boot-info-table -R -J -v -T isolinux/
```

8. The last step is writing the image file you created, `file.iso`, to a CD-ROM using the process you would normally use to write to a CD-ROM.

Starting a Kickstart Installation

To begin installation, boot the system from a boot disk, and when the boot prompt appears, enter one of the following boot commands.

If the Kickstart file is on the floppy or CD boot disk, enter one of the following boot commands:

- `linux ks=floppy`
- `linux ks=cdrom:/ks.cfg`

If the Kickstart file is on a server, use the following boot command:

```
linux ks=<file location>
```

The installation program, Anaconda, looks for a Kickstart file at the location specified in the `ks` command line argument that is passed to the kernel. You can enter any of the following as file locations for the `ks=<file location>` command.

- **`ks=hd:<device>:/<file>`** — Use this command to have the installation program mount the file system on `<device>`. (`<device>` must be VFAT or ext2). Kickstart searches for the configuration file as `<file>` in that file system (for example, `ks=hd:sda3:/mydir/ks.cfg`).
- **`ks=file:/<file>`** — Use this command to have the installation program read the file `<file>` from the file system without mounting anything. You would use this option if the Kickstart file is already on the `initrd` image.
- **`ks=nfs:<server>:/<path>`** — Use this command if you want the installation program to look for the Kickstart file on the NFS server `<server>`, as file `<path>`. The Ethernet card will be configured using DHCP.
- **`ks=http:<server>:/<path>`** — Use this command if you want the installation program to look for the Kickstart file on an HTTP server `<server>`, as file `<path>`. The Ethernet card will be configured using DHCP.
- **`ks=cdrom:/<path>`** — Use this to force the installation program to look for the Kickstart file on CD-ROM, as file `<path>`.
- **`ks`** — If you enter `ks` without any command line options, the installation program will use the boot server field of the configuration file to find a DHCP server to get information about an NFS server. The Ethernet card

will be configured using this information. Finally, the installation program tries to find the Kickstart configuration file in one of the following places:

- If the name of the bootfile begins with a `/`, Kickstart will search the NFS server's root directory.
- If the name of the bootfile does not begin with a `/`, Kickstart will search the NFS server's `/kickstart` directory.
- If you don't list a bootfile, the PC receiving the IP address is searched.

NOTE To use Kickstart for a network installation, you need to configure a DHCP server. See Chapter 11 for details about configuring a DHCP server.

Summary

In this chapter, you learned about installing Fedora Core and Red Hat Enterprise Linux using the Kickstart program. First, you learned about the Kickstart configuration file and how to configure it using the Kickstart Configurator. Then you learned how to start the Kickstart installation.

Exploring the Desktops

IN THIS CHAPTER

- Examining the Graphical Login Screen
- Logging in and Using the GNOME Desktop
- Using the Nautilus File Manager
- Configuring GNOME
- Taking a Look at KDE
- Using the Konqueror File Manager

The GNU Network Object Model Environment (GNOME) desktop is a graphical user interface (GUI) that is installed as the default user interface during the installation process. Another popular desktop, KDE (K Desktop Environment), can also be selected as an option to be installed during system installation. Each of these user interfaces is similar to that of MS Windows or Mac OS X but with some notable differences. One large difference is the ability of the user to select which desktop to use upon system login. In this chapter, you examine both of these GUIs to discover some of the features that they offer and how to configure them to your liking.

Examining the Graphical Login Screen

Before you can do any exploring of the GNOME or KDE desktops, you must first log in. You log in from the graphical login window that is shown in Figure 5-1.

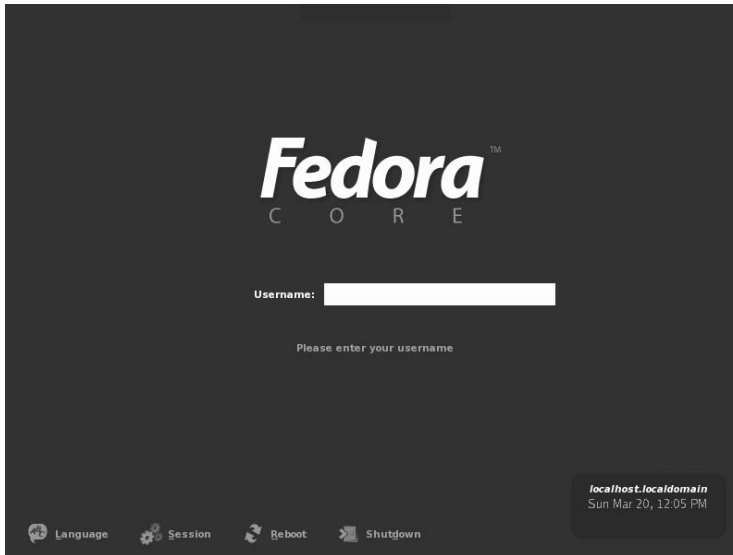


Figure 5-1 The graphical login window waits for you to log in.

Take a quick look at the options that you can choose from the login window. At the bottom of the window are four choices that you can click to make additional selections:

- **Language** — Clicking this opens a box displaying the languages available on your system. If you want to use the system default language, which was installed during system installation, you don't need to do anything with this choice. In most cases, only one language is listed unless additional languages were installed during the system installation. The default language will typically be the language used at your location. If other languages have been installed, just click the language that you want to use.
- **Session** — Clicking Session gives you the opportunity to select the desktop that you use after you log in. GNOME is the default desktop, so you need to use this choice only if you want to change to a different desktop, such as KDE.
- **Reboot** — When you click Reboot a message asks whether you want to reboot the system.
- **Shut Down** — When you click Shut Down a message asks whether you want to shut down your system.

Directly in the center of the window is the login field. This is where you enter your username and password to login. To login, do the following:

1. Type your username.
2. Press Enter.
3. Type your password.
4. Press Enter again.

Logging In and Using the GNOME Desktop

In this section you log in to the GNOME desktop and do some exploring to help you become familiar with its features. As mentioned earlier, the GNOME desktop is installed as the default desktop, so to enter GNOME, you can just enter your username and password in the graphical login window without having to make any choices from the four options, as explained in the preceding section.

NOTE The GNOME desktop in Fedora Core 4 is slightly different from the GNOME desktop in Enterprise Linux 4 because they use different versions of GNOME. These differences are related to the menus displayed at the top of the desktop and are explained in this chapter.

After entering your username and password, you see the Fedora Core 4 GNOME desktop, as shown in Figure 5-2.

The GNOME desktop has a similar appearance to other well-known desktop environments such as MS Windows or Mac OS X. If you can use either of these desktops, you can easily master GNOME in a short time. Notice that the GNOME desktop has a rather clean, almost Spartan, appearance.

The three icons in the upper-left corner of the desktop are links to your home directory, the system trash can that holds your deleted files until you empty the trash, and the Computer icon that opens the Nautilus graphical shell. The Nautilus File Manager gives you access to your files and directories so that you can do typical file management tasks such as copying and moving files. In addition to regular file management tasks, the Nautilus File Manager lets you perform desktop management as well. You look more closely at Nautilus later in this chapter. This list provides a closer look at these icons:

- **Computer** — This icon also opens a Nautilus window the system. Your system devices may be different. The Computer window on my system contains four icons that are links to the following options:

- *Floppy Drive* — The Floppy Drive icon is a link to the folder that contains the system mount point for the floppy drive. Double-clicking this icon displays the contents of the floppy disk that you inserted in the floppy drive.
- *CD-R Drive* — The CD-R Drive icon is a link to the folder that contains the system mount point for the CD-R drive. Double-clicking this icon displays the contents of the CD-ROM disk that you inserted in the CD-R drive.
- *Filesystem* — This icon is a link to the file system. Double-clicking this icon opens a new window displaying the root directory of the file system.
- *Network* — Clicking the Network icon gives you access to the network file systems. Any files or directories that are available across your network are shown here.

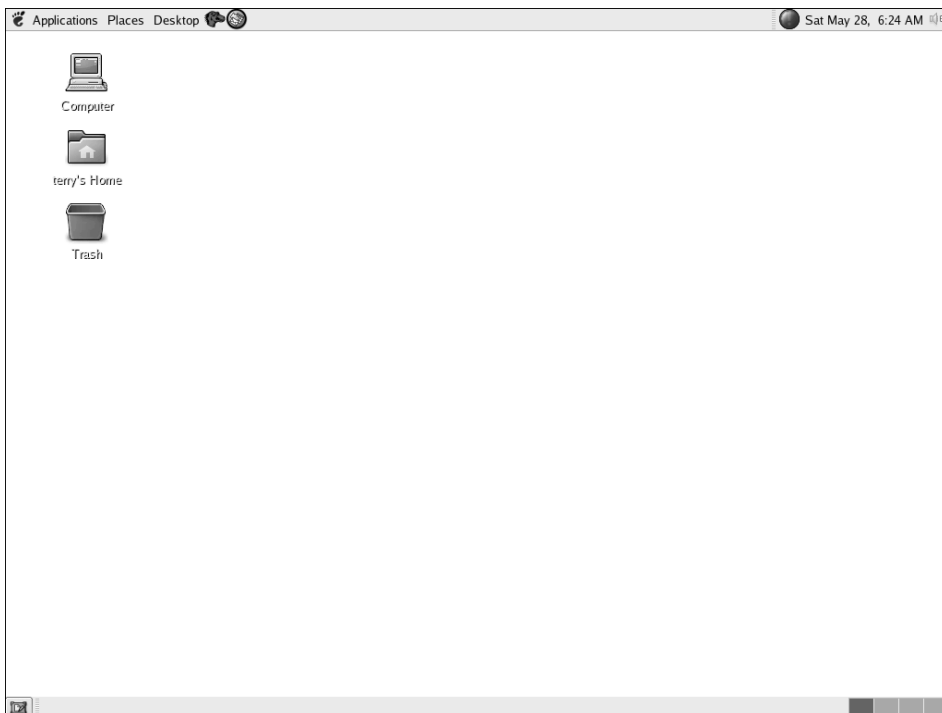


Figure 5-2 The Fedora Core 4 GNOME desktop after logging in for the first time.

- **Home directory** — This icon is a link to the user's home directory. The name of the user shown on the desktop corresponds to the user who is logged in. For example, Figure 5-2 shows the icon labeled as *terry's Home* because I logged in with that username. You can double-click this icon — or right-click and choose Open from the contextual menu — to open a window that displays the user's home directory. If you right-click and choose Browse Folder, a File Browser window opens.
- **Trash** — This icon is a link to the system trash can. You can drag any icon, file, or directory and drop it here. When you're ready to empty the trash, just right-click and select Empty Trash from the contextual menu.

Playing with the Panel

At the top and bottom of the desktop is a gray, horizontal bar. This area of the desktop is the *panel* and is similar to the taskbar in Windows. The left side of the top panel differs slightly between the Fedora Core version and the Enterprise Linux 4 version because Fedora Core and Enterprise Linux use different versions of GNOME. The description here will explain the differences. On the far left of the top panel of both Enterprise Linux and Fedora Core is the Applications icon, indicated by the word Applications and the Red Hat icon. To the right of the Applications icon is an Actions menu on Enterprise Linux that contains some actions you can perform, such as locking the desktop or logging out. To the right of the Applications icon on a Fedora system is the Places menu that gives you access to your file system and other locations such as a remote computer or network connection. To the right of the Places menu is the Desktop menu that gives you access to system settings and preferences and also lets you log out. To the right of the Actions menu on Enterprise Linux and the Desktop menu on Fedora are icons representing programs that were installed during the system installation. You can start any of these programs by clicking them from the panel. Just move your mouse over any icon, and a pop-up message appears with a description of the program represented by the icon.

At the far right of the top panel is a small speaker icon that you can click to open a slider box to change the system volume setting. To the left of the speaker icon is the date and time display. You can change the system date and time properties by right-clicking on the date and time display and selecting your choice from the pop-up context menu. If you left-click on the date and time display a small calendar window opens and displays the current month. Click again to close the calendar. To the left of the date and time display is the system status icon, which shows update status for the system. This icon can be shown as a blue check mark, a red exclamation point, green arrows, or a gray question mark.

CROSS-REFERENCE Refer to Chapter 26 for more information about the system status icon and how to update your system.

At the far right of the bottom panel is a square gray area — the Workspace Switcher — that is divided into four sections. When you first log in to GNOME the leftmost section of Workspace Switcher should be blue, indicating that you are in workspace one. You can switch between four workspaces in GNOME, so you actually get four distinct desktops that you can use. You can open different programs on the different desktops and switch between them by clicking the Workspace Switcher for the desktop that you want to see. Open some programs on the different desktops and then try clicking each of the four squares to see the effect of changing to a different workspace.

On the far left of the bottom panel is a Close Window icon that will hide, if visible, all open windows on the desktop. If the windows are already hidden, clicking this icon displays the windows. The open area on the bottom panel between the Workspace Switcher and the Close Window icon is used to show any programs that you're running on your desktop. You can switch between programs running on a single desktop by clicking the program name from the bottom panel. Also shown in this area are icons that you can add to the panel as well as *applets* — applications that provide some type of useful information or entertainment.

Managing Applets on the Panel

The icons on the top and bottom panels are small programs called applets that have a variety of uses. For example, there is a weather applet that you can place on the panel to give you weather forecasts for any area you desire. In addition to the applets that are already on the panel, you can add your own. You also can move applets that are already there or delete them to make more room.

To add applets to the panel, do the following:

1. Right-click an empty area of the panel.
2. Choose Add to Panel from the contextual menu.
3. Choose the applet that you want to add.
4. Click Add to add it to the panel.

To move applets to another location on the panel:

1. Right-click the applet you want to move.
2. Click Move from the contextual menu.
3. Drag the applet to the desired location.
4. Click to release the applet to its new location.

To remove an applet from the panel:

1. Right-click the applet you want to remove.
2. Choose Remove from Panel from the contextual menu.

To modify the properties of an applet (or the panel):

1. Right-click the applet (or an empty area of the panel).
2. Choose Properties from the contextual menu.
3. Change the parameters in the Properties dialog box.
4. Click Close to accept your changes and close the Properties dialog box.

TIP Right-clicking the panel or any applets on it presents a contextual menu, which gives you access to Help and some useful utilities for panel configuration. Contextual menus may be different, depending on the type of applet that you're selecting.

Choosing Items from the Applications Menu in Fedora Core

The Applications menu, represented by the Red Hat icon, is on the far-left corner of the top panel. The Applications menu button gives you access to a large number of applications. Click the Red Hat icon to open the Applications menu, and you see a menu, as shown in Figure 5-3, listing the many categories of applications from which you can choose.

Notice that many of the categories contain a right-pointing arrow. Moving your cursor over categories with a right-pointing arrow opens additional menus from which you can choose even more applications in that category. There are probably more than 100 applications from which you can choose, many more than we can describe in this chapter. However, we do provide a brief description of the main category of applications here so you can have some idea what they do. Begin by starting at the top of the menu and then work your way toward the bottom.

- **Accessories** — Here you can find applications that don't fit well into the other categories, such as the calculator, as well as some text editors.
- **Graphics** — This menu choice contains graphical programs. Here you find image viewing and editing applications.
- **Internet** — Here you will find applications related to the Internet. For example, the Web browsers are located here as well as an FTP program.

- **Office** — This menu choice gives you access to the OpenOffice.org office suite. The OpenOffice suite contains word processing, spreadsheet, and presentation software, and much more. You can also start several of the OpenOffice applications by clicking the icons on the left side of the panel.
- **Sound & Video** — Choosing this item gives you access to programs and utilities related to system sound and video. For example, if you want to adjust the system volume, use the utility here.
- **System Tools** — This menu choice gives you access to many Enterprise Linux system administration utilities. You explore many of these tools in other chapters of this book.
- **Run Application** — This menu item opens a dialog box where you can enter the name of a program that you want to run.

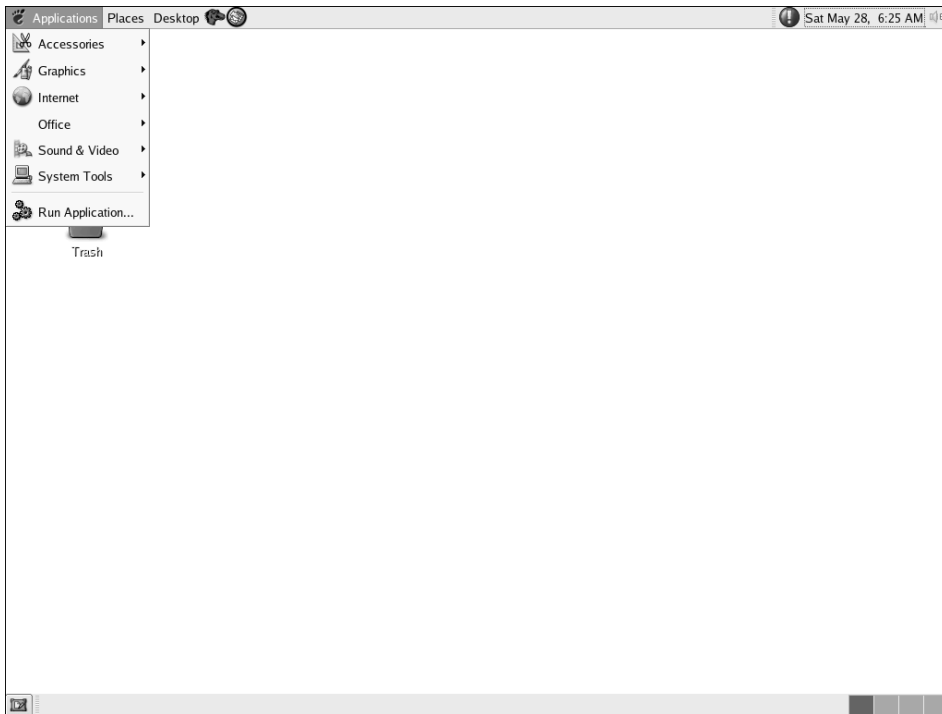


Figure 5-3 The Applications menu on the GNOME desktop in Fedora Core.

Choosing Items from the Places Menu in Fedora Core

The Places menu in Fedora Core is where you can choose the locations you want to view or go to. Figure 5-4 shows the Places menu.

Items on this menu include:

- **Home Folder** — This menu item opens the home folder of the user who is logged in to the system.
- **Desktop** — This menu item gives you a quick way to get to your desktop. It is really useful when you have several windows open and want to go to the desktop without having to close the open windows.
- **Computer** — This menu item opens the computer folder of the user who is logged in to the system.
- **Connect to Server** — Choosing this menu item opens the Nautilus File Manager and displays any network servers that you can connect to.
- **Search for Files** — Choosing this menu item opens a file search dialog box.
- **Recent Documents** — Documents that you have recently opened appear in this list.

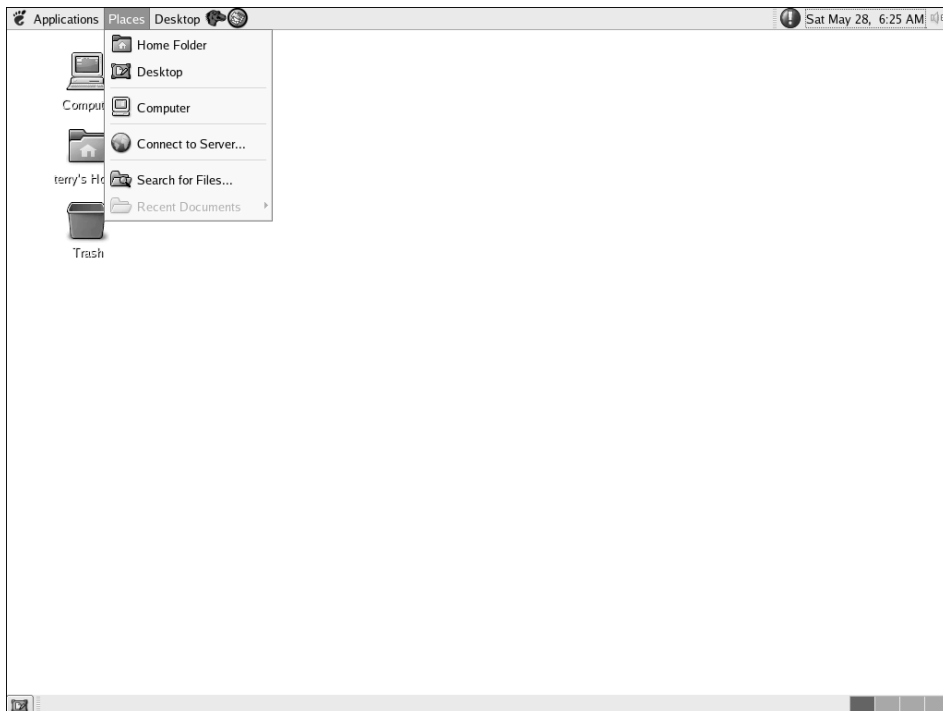


Figure 5-4 The Places menu in Fedora Core lets you choose places to go.

Choosing Items from the Desktop Menu in Fedora Core

Items you can choose from the Desktop menu let you make changes to your system preferences and settings. You can also get systemwide help and lock your desktop and log out. Figure 5-5 shows the Desktop menu.

The following items are available on the Desktop menu:

- **Preferences** — This menu choice opens the System Preferences window. Most of the GNOME settings can be modified with this menu choice. Selecting this from the menu is the same as double-clicking the Computer icon on the desktop.
- **System Settings** — This menu item contains Enterprise Linux system administration utilities and some GNOME configuration utilities as well.
- **Help** — This menu item opens the Help browser. You can get help on using GNOME by choosing this item.
- **About GNOME** — Choosing this item gives you information about the GNOME version you are using.



Figure 5-5 The Desktop menu in Fedora Core lets you change system preferences.

- **Lock Screen** — This menu option starts your system screensaver and locks your desktop. Move your mouse or press a key to open a dialog box that lets you enter your password to unlock the desktop.
- **Log Out** — Choosing Log Out opens a dialog box giving you the option to log out, shut down, or restart the computer. Select the radio button of your choice, and then click OK.

Choosing Items from the Applications Menu on Enterprise Linux

The Applications menu, represented by the Red Hat icon, is on the far-left corner of the top panel. The Applications menu button gives you access to a large number of applications. Click the Red Hat icon to open the Applications menu, and you see a menu, as shown in Figure 5-6, listing the many categories of applications from which you can choose.

Notice that many of the categories contain a right-pointing arrow. Moving your cursor over categories with a right-pointing arrow opens additional menus from which you can choose even more applications in that category. There are probably more than 100 applications from which you can choose, many more than we can describe in this chapter. However, we do provide a brief description of the main category of applications here so that you can have some idea what they do. Start at the top of the menu and then work your way toward the bottom.

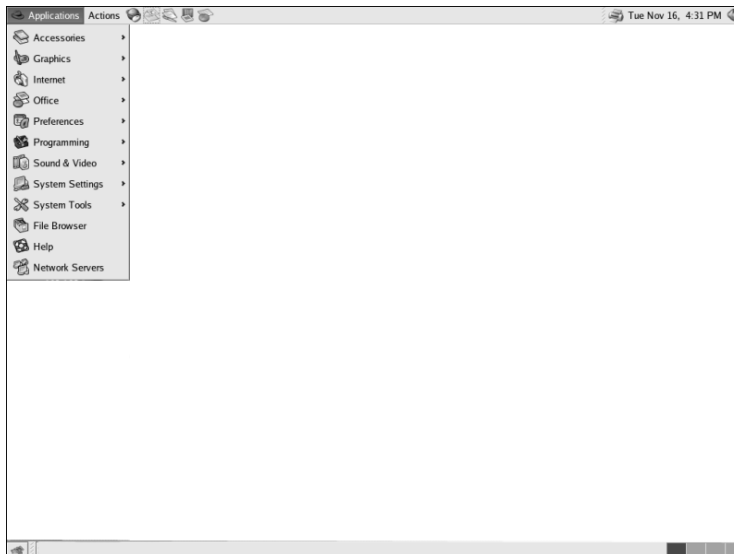


Figure 5-6 The Applications menu on the GNOME desktop in Enterprise Linux.

TIP Your Applications menu might not be exactly as described in this section, depending on the type of system (Desktop, Workstation, Server) that you have installed.

- **Accessories** — Here you can find applications that don't fit well into the other categories, such as the calculator, as well as some text editors.
- **Graphics** — This menu choice contains graphical programs. Here you find image viewing and editing applications.
- **Internet** — Here you will find applications related to the Internet. For example, the Web browsers are located here as well as an FTP program.
- **Office** — This menu choice gives you access to the OpenOffice.org office suite. The OpenOffice suite contains word processing, spreadsheet, presentation software, and much more. You can also start several of the OpenOffice applications by clicking the icons on the left side of the panel.
- **Preferences** — This menu choice opens the System Preferences window. Most of the GNOME settings can be modified with this menu choice. Selecting this from the menu is the same as double-clicking the Computer icon on the desktop.
- **Programming** — This menu item gives you access to some programs that can be used for debugging programs.
- **Sound & Video** — Choosing this item gives you access to programs and utilities related to system sound and video. For example, if you want to adjust the system volume, use the utility here.
- **System Settings** — This menu item contains Enterprise Linux system administration utilities and some GNOME configuration utilities as well.
- **System Tools** — This menu choice gives you access to many Enterprise Linux system administration utilities. You explore many of these tools in other chapters of this book.
- **File Browser** — This menu item is a link to the Nautilus File Manager and opens in the user's home directory.
- **Help** — This menu item opens the Help browser. You can get help on using GNOME by choosing this item.
- **Network Servers** — Choosing this menu item opens the Nautilus File Manager and displays any network servers that you might have.

TIP Fedora Core and Enterprise Linux offer you several ways to start applications. You can click the Applications menu icon in the left corner of the panel. You can also start any executable application by double-clicking its icon from the Nautilus File Manager.

Choosing Actions from the Actions Menu in Enterprise Linux

To the right of the Applications menu is the Actions menu, as shown in Figure 5-7.

Items on this menu, listed from top to bottom, include the following:

- **Run Application** — This menu item opens a dialog box where you can enter the name of a program that you want to run.
- **Search for Files** — Choosing this menu item opens a file search dialog box.

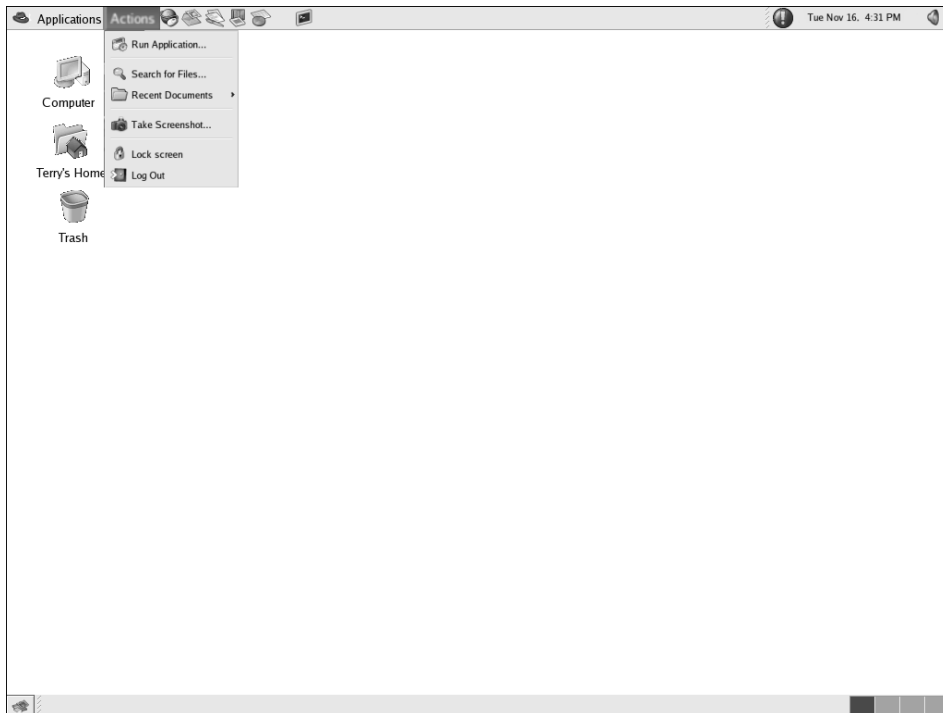


Figure 5-7 The GNOME Actions menu in Enterprise Linux 4.

- **Recent Documents** — Documents that you have recently opened appear in this list.
- **Take Screenshot** — You can use this menu choice to capture an image of your current display.
- **Lock Screen** — This menu option starts your system screensaver and locks your desktop. Move your mouse or press a key to open a dialog box that lets you enter your password to unlock the desktop.
- **Log Out** — Choosing Log Out opens a dialog box giving you the option to log out, shut down, or restart the computer. Select the radio button of your choice and then click OK.

Using the Nautilus File Manager

The Nautilus File Manager is a graphical shell for GNOME. You can use Nautilus not only to manage the files and directories on your system but also to perform many GNOME and system configurations. With Nautilus, you can even access your system applications.

To start the Nautilus File Manager, use any of the following methods:

- In Enterprise Linux select File Browser from the Applications menu. In Fedora Core choose System Tools ⇨ File Browser.
- Right-click any folder and choose Browse Folder from the contextual menu.

Using either of these methods will open the Nautilus File Manager, as shown in Figure 5-8.

A brief explanation of the items on the Nautilus File Manager window is in order:

- **Menu bar** — At the top of the window is the menu bar, which is similar to menu bars from other programs. From the menu bar, you can access choices for performing various actions.
- **Toolbar** — Below the menu bar is the toolbar. The toolbar holds buttons that you can use to perform the actions such Back, Forward, and Reload.
- **Location bar** — The location bar contains a text field where you can enter a file, folder, or FTP site to go to. The location bar also has a zoom-in and a zoom-out button (magnifying glass icons) with which you can change the size of items. Finally, the View As Icons drop-down list lets you choose how you want to view the items.

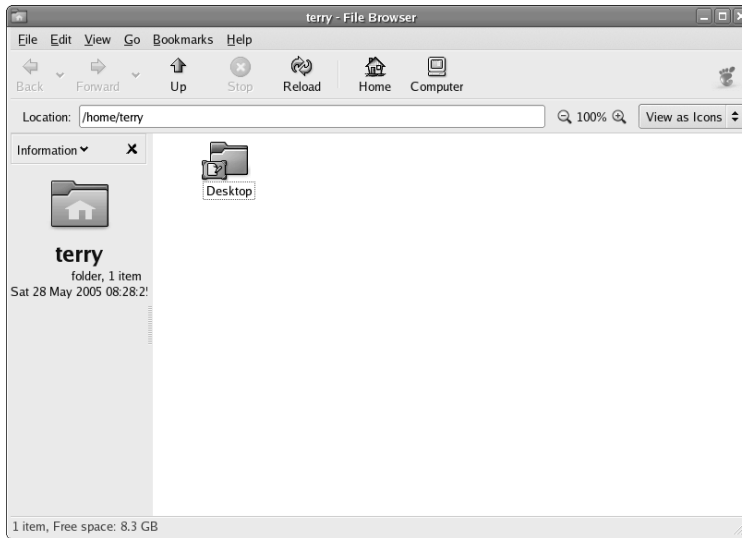


Figure 5-8 The Nautilus File Manager window.

- **Window panes** — Beneath the location bar, the Nautilus window is divided into two panes. The left, smaller pane (Information) shows a drop-down list that lets you choose what is displayed about items appearing in the larger, right pane. If you choose Tree from the list, you can see your entire file system tree in the left pane.

The larger, right pane displays the contents of the files or directories that you're viewing. *Note:* All directories appear as folders in Nautilus. You can view the contents of folders as either a list or as icons by choosing from the View As Icons drop-down list (in the location bar). You can also access FTP sites by entering the URL into the location text area.

- **Status bar** — At the bottom of the Nautilus window is the status bar, which displays status information about the files or folders that you are viewing.
- **Resize handle** — In the lower-right corner is a handle that you can use to resize the window. Move your mouse over the handle and then click and drag to resize the window. Release the mouse button when the window is the desired size.

TIP When using the Nautilus File Manager, all directories are shown as folders. For the remainder of this section, we refer to directories as folders.

Displaying Your Home Folder

If you start Nautilus by using one of the methods explained earlier, Nautilus opens to your home folder. However, if you changed folders while in Nautilus, you might want to return to your home folder. You can do this in one of two ways:

- Choosing Go ⇨ Home from the Nautilus menu bar
- Clicking the Home icon on the Nautilus toolbar

If you want to refresh the display, click Reload on the toolbar.

Displaying the Contents of a Folder

You can easily move from one folder to another in Nautilus. Again, you have more than one way to navigate through your file system.

- **Double-click the folder** — If the folder that you want to view is visible in the large, right pane of the Nautilus window, you can open it by double-clicking it.
- **Enter the location** — You can enter the name of the folder that you wish to view by typing it into the location bar text field.
- **Use the tree view** — Choose Tree from the drop-down list in the small, left pane of the Nautilus window and select the folder that you wish to view.
- **Use the Search tool** — Click the Actions menu button and choose Search for Files from the menu.

To move forward and backward through your file system, you can use the Forward and Back buttons from the toolbar or you can choose Go ⇨ Forward/Back from the menu bar. To move up a level, you can use the Up button on the toolbar or you can choose Go ⇨ Up from the menu bar.

Opening Files

Whenever you double-click a file, Nautilus is configured by default to perform some type of action on the file, depending on the type of file. Nautilus either opens the file by using a preconfigured viewer or runs the file if it is an executable file.

Nautilus has been configured to open the following types of files in the large, right pane:

- **Graphic image files** — Nautilus automatically displays a small icon of the graphic image, typically called a thumbnail, in the folder view. Double-clicking the thumbnail opens the file in the left window. Click the Back button on the toolbar to return to the folder view. Nautilus can display GIF, JPEG, and PNG images.
- **Text files** — Nautilus opens any text files in the default text editor, which is `gedit`. With `gedit` you can edit the text as you desire. When you are finished, close `gedit` by clicking File ⇨ Exit or click the X in the upper-right corner of the `gedit` window.

Accessing FTP Sites

You can use the Nautilus File Manager to access an FTP site. All you need to do is enter the URL of the site in the location bar text field. If you need to log in to the site, you can use the following syntax.

```
ftp://username:password@hostname.domain
```

You can drag and drop files to move them from the FTP site to your desired folder.

Using Bookmarks

With Nautilus, you can use bookmarks to keep track of your favorite locations. You can bookmark files, folders, and FTP sites as desired.

Adding a Bookmark

To add a bookmark, do the following:

1. Click the item that you wish to bookmark.
2. Choose Bookmarks ⇨ Add Bookmark from the menu bar.

Editing Bookmarks

To edit a bookmark, do the following:

1. Choose Bookmarks ⇨ Edit Bookmarks to open the Edit Bookmarks dialog box, as shown in Figure 5-9.

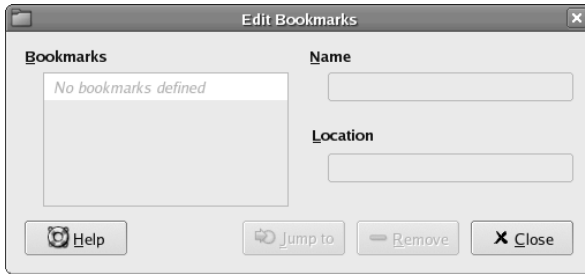


Figure 5-9 The Edit Bookmarks dialog box.

2. Select the bookmark from the list on the left side of the dialog box.
3. Change the name and location as desired. If you enter a different location, you can click Jump to to immediately go to that location.
4. Click Close to finish editing the bookmark.

Deleting Bookmarks

To delete a bookmark:

1. Choose Bookmarks ⇨ Edit Bookmarks from the menu bar. The Edit Bookmarks dialog box opens.
2. Select the bookmark that you want to remove.
3. Click the Remove button.
4. Click the Close button to close the dialog box.

Managing Your Files and Folders

You can take many actions when managing your file system with Nautilus. Table 5-1 briefly explains the action that you want to perform and how you should do it.

Table 5-1 Performing Tasks Using Nautilus

ACTION	METHOD
Move an item	Click item and drag it to desired location.
Copy an item	Click item and hold Ctrl while dragging item.
Link to an item	Click item and press Ctrl+Shift while dragging.

ACTION	METHOD
Select single item	Click item.
Select contiguous items	In icon view, click and drag box around items. In list view, press Shift; click the first item, and then click the last.
Select multiple items	Press Ctrl; click desired items.
Select all items	Choose Edit ⇨ Select All Files from menu bar.
Create folder	Right-click and choose Create Folder from contextual menu.
Rename item	Right-click and choose Rename from the contextual menu.
Move to trash	Right-click and choose Move to Trash from the contextual menu.
Delete item	Right-click and choose Move to Trash.
Change permissions	Right-click, choose Properties, and click the Permissions tab.
Display trash	Right-click the Trash icon and choose Open from the contextual menu.
Restore trashed item	Open Trash folder and drag item to desired location.
Empty trash	Right-click the Trash icon and choose Empty Trash.
Add emblem	Right-click, choose Properties, click the Emblems tab, and choose desired emblem.
Change single icon	Right-click, choose Properties, click Select Custom Icon, and choose desired icon.
Change item size	Choose Zoom In or Zoom Out from toolbar.

Customizing the Nautilus File Manager

A very nice feature of Nautilus is that you can configure it to make it work the way you want it to. You can change many preferences; in this section, I tell you about them and how to change them.

Editing File Manager Preferences

To open the Nautilus File Management Preferences dialog box, choose Edit ⇨ Preferences from the menu bar in a Nautilus window. The dialog box shown in Figure 5-10 appears.

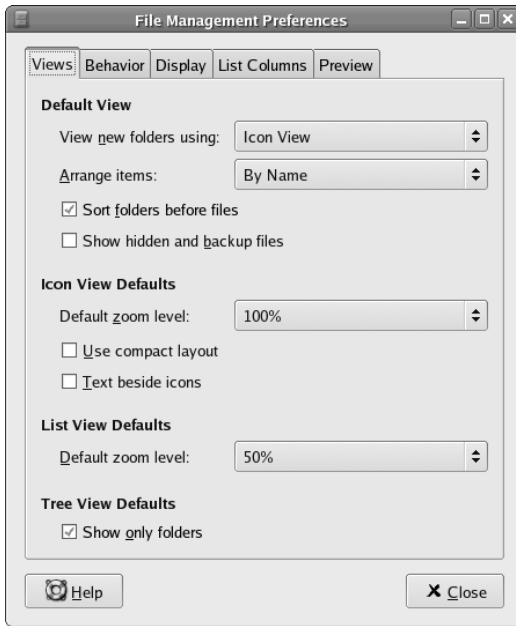


Figure 5-10 The Nautilus Preferences dialog box.

On this dialog box are five tabbed pages:

- **Views** — Preferences on this tab give you options for setting the default view parameters for Nautilus, such as icon view, sort order, and showing hidden files.
- **Behavior** — Preferences on this tab are concerned with the way Nautilus handles executable files and trash. You can also choose between single- and double-clicking here.
- **Display** — This tab lets you decide what information you want displayed with your icons, such as size, date created, date modified, and date format.
- **List Columns** — The settings on this tab let you choose what information is displayed as well as its order, when you choose list view.
- **Preview** — The settings on this tab determine how your files are displayed in their folders. For example, you can decide here whether you want thumbnail views of graphic files.

TIP You can change many preferences to alter the appearance and performance of Nautilus. You have seen only a few of them, so experiment with them to see for yourself what they do.

Changing the File Manager Background and Icon Emblems

Another nice feature of Nautilus is the ability to display colors and patterns in the File Manager window. For example, I like to change the background color for my home directory to light blue. That way, I can tell immediately when I'm in my home directory when I see the blue background. You can also assign emblems to icons. *Emblems* are small graphics that are used to make your icons more meaningful. You can easily change the colors and patterns or add emblems by doing the following:

1. Choose Edit ⇨ Backgrounds and Emblems from the Nautilus menu bar to open the Backgrounds and Emblems dialog box, as shown in Figure 5-11.
2. Click the Patterns, the Colors, or the Emblems button on the left side of the dialog box.
3. Click and drag the pattern, color, or emblem from the right side of the dialog box to where you want to place it. You can drag a color or pattern to the large, right pane of the File Manager window to change the color or pattern. You can drag an emblem to an icon to attach it to the icon. Click close when you are finished.

TIP You can also drag the patterns or colors directly to the desktop and drop them there. Your desktop will change to reflect your new color or pattern.

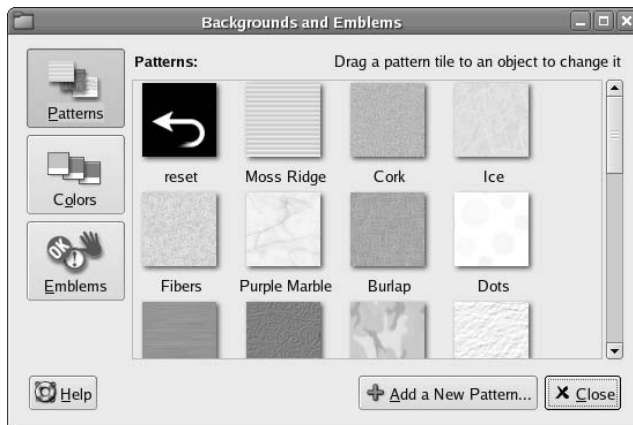


Figure 5-11 The Background and Emblems dialog box.

Showing and Hiding Views

You can decide what you want to view and what you don't in your File Manager. You can view or hide the side pane, the status bar, the toolbar, and the location bar by clicking the appropriate item from the View menu on the menu bar. These items are toggled items. If the item is checked, it is available for viewing; if not checked, it is not available. Clicking the item toggles it on or off.

Configuring GNOME

You can also customize your entire desktop as easily as you configure your Nautilus File Manager. Quite a few preferences can be modified in GNOME. We can't possibly explain all of them here in this chapter, but we can show you how to change one of them. You can play around with the rest and make changes as you desire. Take a look at setting up a screensaver. To set the preferences for the screensaver, do the following:

1. Choose Applications ⇨ Preferences ⇨ Screensaver in Enterprise Linux or Desktop ⇨ Preferences ⇨ Screensaver in Fedora Core. The Screensaver Preferences dialog box, as shown in Figure 5-12, opens.
2. Choose the mode for the screensaver by making your choice from the drop-down list.
3. Select the image or images that you want for your screensaver by selecting the check box in front of your choice.
4. Pick the times that you want to use.

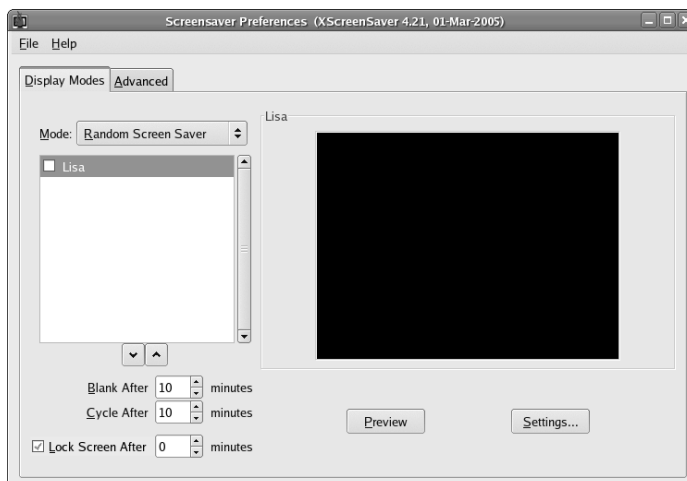


Figure 5-12 Configure the screensaver here.

TIP Also be sure to look at the Advanced tab to see whether you want to change any items there. Items on the Advanced tab include image manipulation, display power managements, color map, and diagnostic settings.

5. When you finish making choices, test your screensaver by clicking the Preview button.

TIP Don't forget to have a look at the settings for the screensavers that you chose. (Click the Settings button to see them.) In many cases, you can create some interesting effects by changing the settings. For example, you can change the speed of the screensaver or the number of colors displayed.

6. Click the Close button when you're finished. Your new screensaver is enabled.

Logging Out

After you finish working in GNOME, you should log out before leaving the PC. Logging out is always a good idea to prevent anyone from using your system. You can log out of GNOME as follows:

1. Choose Actions ⇨ Log Out in Enterprise Linux or Desktop ⇨ Log Out in Fedora Core.
2. From the Log Out dialog box, you can choose to log out, restart the system, or shut down the system by selecting the radio button in front of your choice.
3. After making your choice, click OK to execute it.

Taking a Look at KDE

The default desktop in Fedora Core and Enterprise Linux is GNOME, but another desktop — KDE — is available if you want to give it a try. If you want to use it, you'll have to make sure that it is installed on your system because the default installation of Fedora Core and Enterprise Linux does not install KDE.

In this section, we give you a brief overview of KDE just to make you aware of it and perhaps tempt you to try it. We will briefly explain the KDE desktop, show you the Applications menu where you can find some applications to try, and tell you about the Konqueror File Manager. After that, you are on your own to explore if you like.

You can check whether KDE is installed from the graphical login screen. Click Session (refer to Figure 5-1) and select KDE from the choices. If KDE is not a choice, it isn't installed — but you can easily install it by using the Package Management tool.

After selecting KDE for your session, enter your username and password to login. You will see the KDE desktop, as shown in Figure 5-13.

The KDE desktop has an appearance similar to other well-known desktop environments such as GNOME or MS Windows or Mac OS X. If you can use these desktops, you will easily master KDE in a short time. Notice that the KDE desktop has a rather clean appearance with little desktop clutter — just one icon at the top and a panel at the bottom. A description of the KDE desktop is in order here.

At the bottom of the desktop is a gray, horizontal bar. This area of the desktop is the *panel* and is similar to the taskbar in Windows. On the far left of the panel is the Applications icon, indicated by the Red Hat icon. To the right of Applications are icons representing programs that were installed during the system installation. You can start any of these programs by clicking them from the panel. Just move your mouse over any icon, and a contextual menu appears with a description of the program represented by the icon.

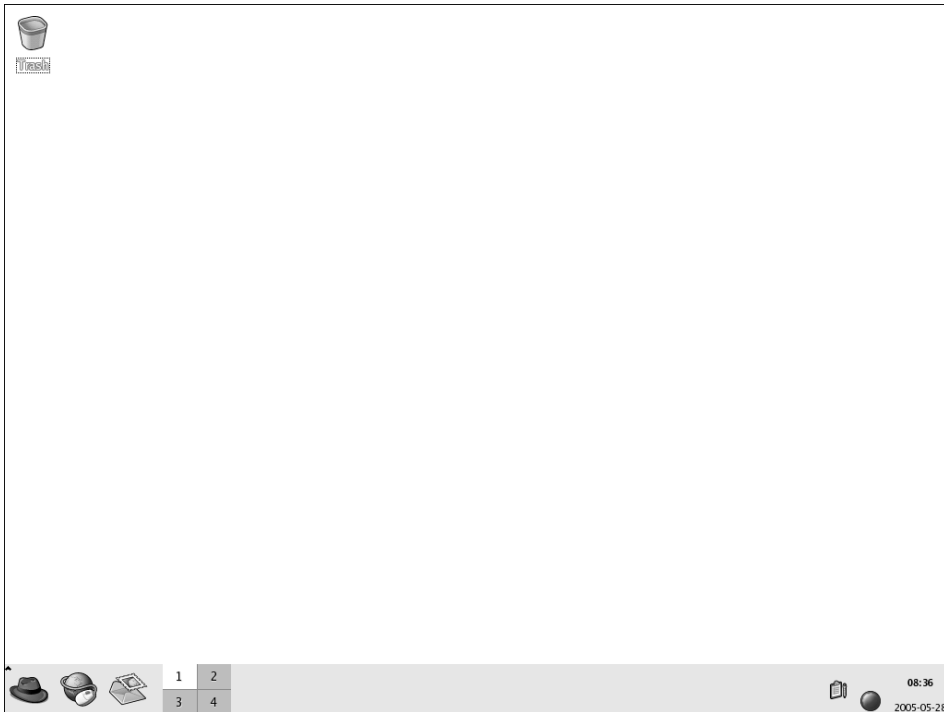


Figure 5-13 The KDE desktop after logging in.

To the right of the program icons on the panel is a square gray area — the Workspace Switcher — that is divided into four sections. When you first log in to KDE, the leftmost section of Workspace Switcher should be white, indicating that you are in workspace one. You can switch between four workspaces in KDE, so you actually get four distinct desktops that you can use. You can open different programs on the different desktops and switch between them by clicking the Workspace Switcher for the desktop that you want to see. Open some programs on the various desktops and then try clicking each of the four squares to see the effect of changing to a different workspace.

On the far right of the panel is a display of the current date and time. The open area on the panel between the Workspace Switcher and the date and time display is used to show any programs that you're running on your desktop. You can switch between programs running on a single desktop by clicking the program name from the bottom panel. Also shown in this area are icons that you can add to the panel as well as applets. *Applets* are applications that provide some type of useful information or entertainment.

Managing Applets

The icons on the bottom panel are small programs called applets that have a variety of uses. For example, there is a weather applet that you can place on the panel to give you weather forecasts for any area you desire. In addition to the applets that are already on the panel, you can add your own. You also can move applets that are already there or delete them to make more room.

To add applets to the panel, do the following:

1. Right-click an empty area of the panel.
2. Choose Add to Panel from the contextual menu.
3. Choose the application that you want to add.
4. Click Add to add it to the panel.

To move applets to another location on the panel:

1. Right-click the applet that you want to move.
2. Click Move from the contextual menu.
3. Drag the applet to the desired location.
4. Click to release the applet to its new location.

To remove an applet from the panel:

1. Right-click the applet that you want to remove.
2. Choose Remove from Panel from the contextual menu.

To modify the properties of an applet (or the panel):

1. Right-click the applet (or an empty area of the panel).
2. Choose Properties from the contextual menu.
3. Change the parameters in the Properties dialog box.

TIP Right-clicking the panel or any applets on it presents a contextual menu, which gives you access to Help and some useful utilities for panel configuration. Contextual menus are different depending on the type of applet that you're selecting.

Choosing Applications from the Applications Menu

The Applications menu, represented by the Red Hat icon, is on the far-left corner of the bottom panel. The Applications button gives you access to a large number of applications. Click the Red Hat icon to open the Applications menu, and you see a menu, as shown in Figure 5-14, listing the many categories of applications from which you can choose.

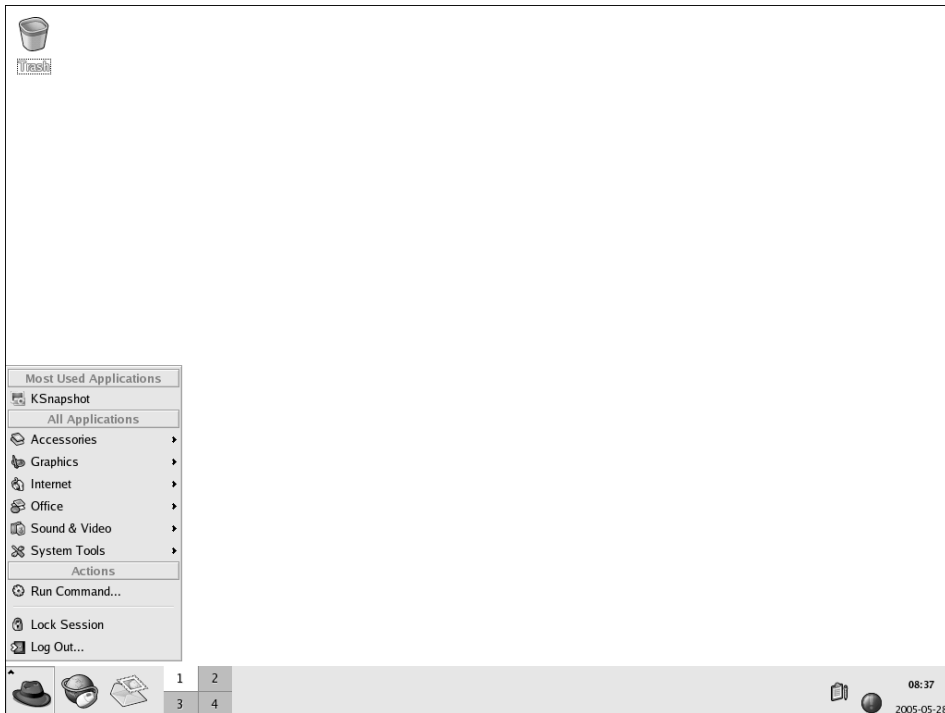


Figure 5-14 The Applications menu on the KDE desktop.

Notice that many of the categories contain a right-pointing arrow. Moving your cursor over categories with a right-pointing arrow opens additional menus from which you can choose even more applications in that category. There are probably more than 100 applications from which you can choose, many more than I can describe in this book. However, I do provide a brief description of the main category of applications here so you can have some idea what they do. Begin by starting at the bottom of the menu and work your way toward the top.

TIP Your Applications menu might not be exactly as described in this section, depending on the type of Fedora Core installation or version of Enterprise Linux you have installed.

- **Logout** — This menu item gives you a quick way to get to your desktop. It is really useful when you have several windows open and want to go to the desktop without having to close the open windows. Choosing Logout opens a dialog box giving you the option to log out or cancel. Select the radio button of your choice and then click OK.
- **Lock Session** — This menu option starts your system screensaver and locks your desktop. Move your mouse or press a key to open a dialog box that lets you enter your password to unlock the desktop.
- **Run Command** — This menu item opens a dialog box where you can enter the name of a program that you want to run.
- **Home** — This menu item is a link to the user's home directory.
- **Help** — This menu item opens the Help browser. You can get help on using KDE by choosing this item.
- **Control Center** — The Control Center is used for making configuration changes to the KDE desktop.
- **System Tools** — This menu choice gives you access to many Enterprise Linux system administration utilities. Tools for configuring your network and printers are located here.
- **System Settings** — This menu item contains Enterprise Linux system administration utilities and some KDE configuration utilities as well. Some of the tools here can be used to configure your Web server as well as other servers.
- **Sound & Video** — Choosing this item gives you access to programs and utilities related to system sound and video. For example, if you want to adjust the system volume, use the utility here.

- **Programming** — This menu item gives you access to some programs that can be used for debugging programs.
- **Preferences** — This menu choice opens the System Preferences window. Most of the GNOME settings can be modified with this menu choice. Selecting this from the menu is the same as double-clicking the Computer icon on the desktop.
- **Office** — This menu choice gives you access to the OpenOffice.org office suite. The OpenOffice suite contains word processing, spreadsheet, and presentation software, and much more. You can also start several of the OpenOffice applications by clicking the icons on the left side of the panel.
- **Internet** — Here you will find applications related to the Internet. For example, the Web browsers and FTP program are located here.
- **Graphics** — This menu choice contains graphical programs. Here you find image viewing and editing applications.
- **Accessories** — Here you can find applications that don't fit well into the other categories, like the calculator, as well as some text editors.

Using the Konqueror File Manager

The Konqueror File Manager is a graphical shell for KDE. You can use Konqueror not only to manage the files and directories on your system but also as a Web browser to access the Internet.

To start the Konqueror File Manager, shown in Figure 5-15, select Home from the Applications menu.

A brief explanation of the items on the Konqueror File Manager window is in order:

- **Menu bar** — At the top of the window is the menu bar, similar to menu bars from other programs. From the menu bar, you can access tools to perform various actions.
- **Toolbar** — Below the menu bar is the toolbar. The toolbar holds buttons that you can use to perform the action indicated by the button, such as back, forward, or reload. The toolbar also has a zoom-in and a zoom-out button (magnifying glass icons) with which you can change the size of items. Finally, the toolbar contains icons that let you choose how you want to view the items in the folder.

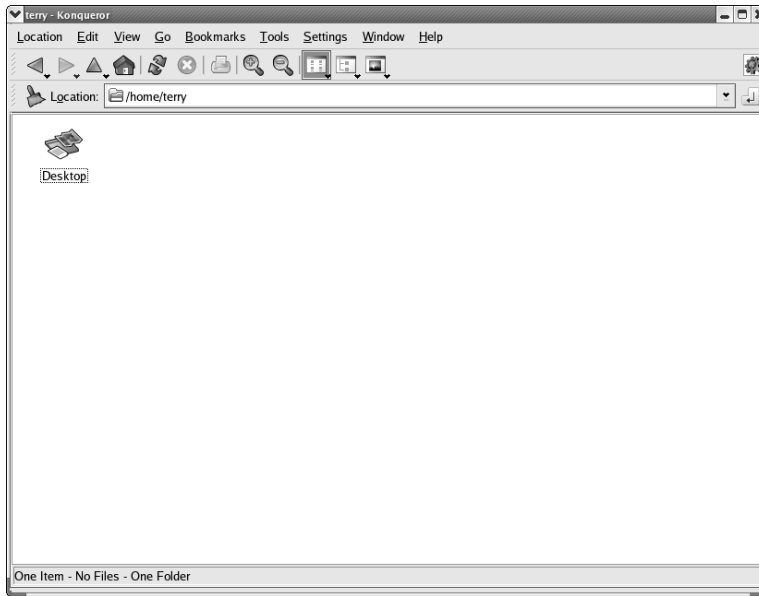


Figure 5-15 The Konqueror File Manager window.

- **Location bar** — The location bar contains a text field where you can enter a file, folder, or URL to go to.
- **Window panes** — Beneath the location bar, the Konqueror window is divided into two panes. The left, smaller pane shows information about the icon selected from the far left side of the File Manager window. Moving your mouse over an icon displays information about the icon. Clicking an item from the list in the left pane displays items in the larger, right pane. If you choose the Root Folder icon, you can see your entire file system tree in the left pane.

The larger, right pane displays the contents of the files or directories that you're viewing. *Note:* All directories appear as folders in Konqueror. You can view the contents of folders as either a list or as icons by choosing from the View As icons (in the toolbar). You can also access Web or FTP sites by entering the URL into the location text field.

- **Status bar** — At the bottom of the Konqueror window is the status bar, which displays status information about the files or folders that you are viewing.

Logging Out of KDE

After you finish working in KDE, you should log out before leaving the PC. Logging out is always a good idea to prevent anyone from using your system. You can log out of KDE as follows:

1. Choose Applications ⇨ Log Out.
2. From the Log Out dialog box, you can choose to log out or cancel to return to the desktop.
3. After making your choice, click OK to execute your choice.

Summary

In this chapter you took the express tour of the two most common desktops provided with Fedora Core and Enterprise Linux. The GNOME desktop is the default desktop that is installed when the system is installed. GNOME is very powerful, and you can use it as is or configure it to your liking as you saw in some of the examples in this chapter. The KDE desktop is not installed by default but can be selected to be installed during system installation, or after the system is already installed. KDE is also a very powerful graphical environment as you saw from some of the examples in this chapter.

System Startup and Shutdown

IN THIS CHAPTER

- Examining the Boot Process
- Exploring Runlevels
- Starting Programs at System Boot
- Shutting Down the System
- Changing the GRUB Configuration

All Red Hat systems, whether Fedora Core or Enterprise Linux, use a similar procedure for starting up the operating system. As the system boots, the operating system loads programs in an orderly fashion. You are able to make changes to the programs that load and their configurations after the system has booted. The changes you make will then affect the boot process the next time and all subsequent times that your system boots.

The process of shutting down the system also follows a consistent, orderly method that you can customize as you desire. For a clear understanding of how your system works, it is good to know the methodology behind the orderly process of bringing your system up as well as shutting it down. By knowing this process in depth, you can make any desired changes to the configuration files and gain total control over the functionality of your system. You will also be able to easily find problems that may be keeping your system from booting properly and quickly correct them. This chapter gives you the details about what happens when you start and shut down your system.

Examining the Boot Process

There are some basic steps that all systems must perform to boot. When you turn on your PC, it runs a program called the basic input/output system (BIOS). The BIOS is the only way to communicate with the system components until the operating system is up and running and able to take over system management functions. Unlike the operating system, which is installed on a user-writable disk, such as a floppy, CD-ROM, or hard drive, the system BIOS is typically installed on a read-only memory (ROM) chip physically attached to the system board. This ROM chip is a type of chip usually referred to as an electronically erasable programmable read-only memory (EEPROM) chip, meaning that it is not normally writable by the end user. It is possible to rewrite an EEPROM BIOS chip, but this requires a program from the chip manufacturer and is not a process that should be taken lightly as any errors here could make your system totally unusable.

After the BIOS loads, it performs some diagnostics on the hardware, checks the installed components to be sure they are functioning, and checks the system RAM. Next, the BIOS tries to find a system drive from which it can load the boot program to begin the process of starting the operating system. You can specify the search order for the drives by changing the settings in the system BIOS configuration, which you can typically access by entering some key combination while the system is performing its power-on self test (POST). If you don't make any changes to the BIOS configuration, most systems by default will look for a bootable floppy disk or CD-ROM before moving on to the system hard drives. Usually the first hard drive that boots is the master IDE device of the primary IDE bus, but you can also change this setting in the BIOS if you desire. The first sector of the drive has an area called the Master Boot Record (MBR), which holds the program that is used to begin the actual loading of the operating system. As soon as the BIOS finds the MBR, it gives up control of the boot process. In the case of Fedora Core and Enterprise Linux, a program called a boot loader begins the loading of the operating system. The boot loader program used is called the *Grand Unified Boot loader*, or GRUB. In the next section you take a closer look at GRUB and how it works.

The Boot Loader

The GRUB program used by Fedora Core and Enterprise Linux uses a two-step process to begin loading the operating system. These two steps are typically referred to as stages one and two. In stage one, a program on the MBR is used to find the second stage program that will begin the process of loading the operating system into system memory. GRUB uses a configuration file

called `/boot/grub/grub.conf` to provide information to the second-stage loader. Later in this chapter you learn how to make changes to the `/boot/grub/grub.conf` configuration file. The first thing the second stage loader does is present you with a nice graphical menu screen, as shown in Figure 6-1.

As you can see from Figure 6-1, there are two versions of the kernel listed with one highlighted. This is the kernel that will be loaded by default. But you can use the GRUB menu to select different Linux kernels, or even different operating systems, to load. In many cases, when someone decides to try Linux for the first time, he or she is already running MS Windows and is planning to set up the system to do a dual boot. So, when the GRUB menu appears there is an additional choice for the other operating system. Most of the time Windows is already installed and Linux is installed later. In this case, the Linux installation would take care of making the changes to the `/etc/boot/grub.conf` file to present the other operating system as a choice on the GRUB menu.

NOTE If you have more than one processor in your system, or you have a Pentium 4 processor with hyper-threading, you will see two kernels listed even on a freshly installed system. The second kernel will end with the letters *smp*, which means symmetrical multiprocessor. In this case, you should choose to boot the SMP kernel.

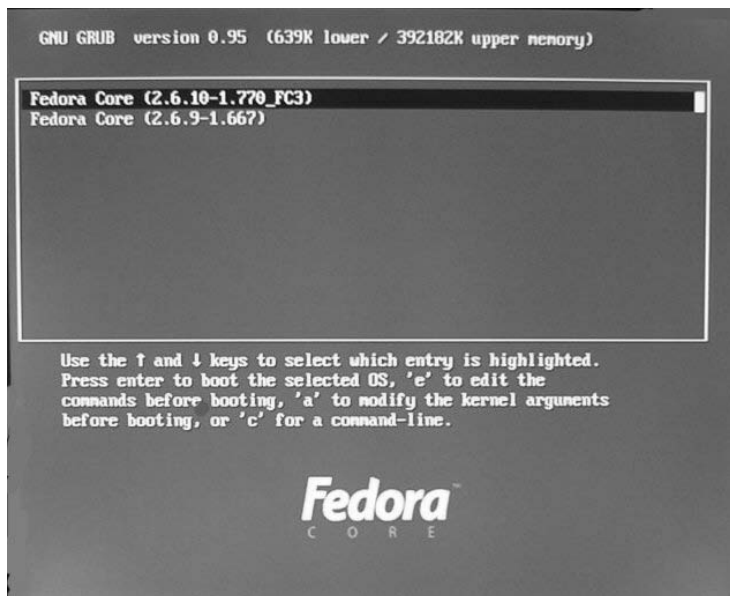


Figure 6-1 The GRUB graphical menu screen shows the kernel(s) available to boot.

If you don't choose a kernel to load, GRUB will load whichever kernel is specified in the configuration file as the default kernel. If you want to select the kernel to load, you can use your cursor keys to highlight the kernel you want loaded. Regardless of whether you choose the kernel or let GRUB do it for you, the next step in the boot process is the actual loading of the kernel. The kernel is always located in the `/boot` directory and will have a name similar to `vmlinuz-2.6.10-1.737_FC3`. Your kernel version number will most likely be different from the version shown here. GRUB has one more task to do and that is to load a ramdisk image, called `initrd` that has the same version number as the kernel you are going to load into system memory. `initrd` loads any special drivers that might be needed by the kernel to load the OS. And that is it for GRUB; its work is done and the kernel is now responsible for continuing the boot process. But before you say good-bye to GRUB, take a look at some of the things you can do with GRUB while the system is booting.

Using GRUB during Boot

In addition to loading a default kernel or operating system when GRUB takes over from the BIOS, GRUB lets you make changes to the parameters it passes to the kernel. You'll notice that Figure 6-1 shows some instructions for selecting the OS as well as which keys to press to edit the parameters that will be passed by GRUB to the kernel. To edit the boot parameters, highlight the kernel you want to edit and press the `e` key. A new screen opens, as shown in Figure 6-2.



Figure 6-2 Selecting a kernel to edit boot parameters.

Figure 6-2 displays the locations of the kernel and `initrd` files that will be used to begin booting the OS. To edit any of the information displayed here, highlight your choice and again press the **e** key. For example, if you want to edit kernel parameters, highlight the line beginning with `kernel` and press **e**. A new screen appears, as shown in Figure 6-3.

To enter any parameters you want to pass to the kernel, type them in at the end of the line. Be sure to separate them by spaces. For example, to tell the kernel not to use ACPI enter the command `acpi=off`. Now when the kernel boots it will disable ACPI. After you have finished entering the parameters you desire press Enter to accept your changes and return to the previous screen. Press the letter **b** and the kernel will boot and begin the process of loading the operating system.

NOTE Using this method to pass parameters to the kernel is only applied to this instance. When the system is rebooted, the parameters will not be passed again. If you want the parameters passed at every boot, you need to put them into the `/boot/grub/grub.conf` file. You learn how to modify this file later in the chapter.

It is also possible to open a command line from GRUB that is similar to a Bash shell except the commands entered are specific to GRUB. To enter the GRUB command line interface, press the letter **c** from the GRUB menu. You can then press **?** to get a listing of the commands you can use. Figure 6-4 shows the GRUB command-line interface listing the possible commands.



Figure 6-3 Editing the kernel boot parameters.

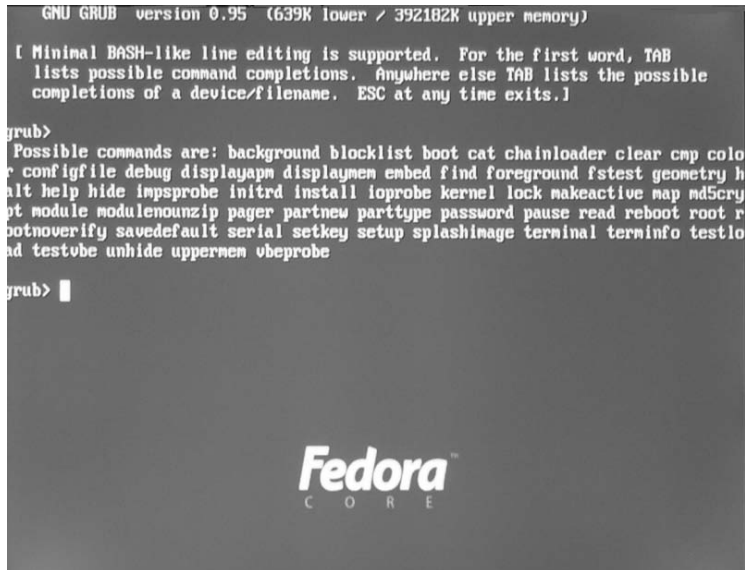


Figure 6-4 The GRUB command-line interface showing possible commands.

TIP You can use the info pages that are already installed on your system to get a complete explanation of all the GRUB commands. Just type `info grub` at a command prompt.

The next section explains the steps taken by the kernel.

The Kernel

The first thing the kernel does after taking over from GRUB is to prepare the system memory for use. Next all system hardware is probed and configured if possible. The kernel will uncompresses the `initrd` in RAM, mount it as a ramdisk, and then runs `linuxrc` in the ramdisk. This can be a command file like a regular `rc` file or a symlink to `init` on the `initrd`. If the former, it runs the commands in there, sets the real root device at the end, and then returns so that `init` starts. If the latter, it runs the commands in `/etc/inittab` on the ramdisk like any other Linux boot process. This can end with a `pivotroot` or `chroot` to the real root device. Fedora's `initrd` files use `/Linux` as a command script; the `initrd` and its `linuxrc` script are very important nowadays because that's what mounts `/proc`, `/sys`, and `/dev/shm`, starts `udev` and `hotplug`, and insmod's special drivers such as SCSI drivers. Most of the time the kernel is able to autodetect and configure hardware devices, but sometimes, especially with new devices, the kernel cannot properly configure them. Usually, this means that your device won't function correctly, or at all, but the

system will still function. If you have set up RAID or Logical Volume Management (LVM) on your system, the kernel will also configure these devices. After the kernel has configured all the system devices and mounted the system drives, it runs the `/sbin/init` command.

The /sbin/init Program

The `/sbin/init` program is the first system process that runs after the kernel has configured the system devices and mounted the system drives. The `/init` program is like the project manager of the system because it manages the remaining steps of booting the system and is the parent or grandparent of all the rest of the automatically started system boot processes. Basically, the `init` program coordinates the order of the many scripts it will run to complete system setup. The first script `/init` runs is the `/etc/rc.d/rc.sysinit` script. This script starts system swap, checks the file systems, and performs other system initialization. Then the `init` command refers to the `/etc/inittab` script to get information about how to start the system, which system initialization script to run and bring the system to the runlevel indicated in the `inittab` script. After a typical system installation, the default runlevel is set to runlevel 5. The rest of this section describes the steps the system takes to boot to runlevel 5.

After reading the `/etc/inittab` script, `init` turns over control to the `rc.sysinit` program which reads the `/etc/rc.d/init.d/functions` file to determine the procedure to use to set the default system path, start and stop programs, find the process ID (PID) of a running process and how to log the success or failure of starting a program. The next script to run is `/etc/rc.d/rc`, which is responsible for starting and stopping services when the runlevel changes and determining the new runlevel.

In the `/etc/rc.d` directory are additional directories `rc0.d`, `rc1.d`, `rc2.d`, `rc3.d`, `rc4.d`, `rc5.d`, and `rc6.d`. The number in the directory name corresponds to the runlevel. Each of these directories contains scripts that are used to stop and start services for the runlevel. In this example, the system is booting to runlevel 5, so the `init` program looks in the `/etc/rc.d/rc5.d/` directory for the processes to start and stop. Listing 6-1 shows the contents of the `rc5.d` directory for a newly installed Fedora Core 3 system.

```
K01yum -> ../init.d/yum
K02NetworkManager -> ../init.d/NetworkManager
K05saslauthd -> ../init.d/saslauthd
K10psacct -> ../init.d/psacct
K20nfs -> ../init.d/nfs
K24irda -> ../init.d/irda
```

Listing 6-1 The scripts used to stop and start services in runlevel 5. *(continued)*

```
K30spamassassin -> ../init.d/spamassassin
K35vncserver -> ../init.d/vncserver
K35winbind -> ../init.d/winbind
K36lisa -> ../init.d/lisa
K50netdump -> ../init.d/netdump
K73ypbind -> ../init.d/ypbind
K74nscd -> ../init.d/nscd
K74ntpd -> ../init.d/ntpd
K85mdmmpd -> ../init.d/mdmmpd
K89netplugd -> ../init.d/netplugd
K90bluetooth -> ../init.d/bluetooth
K94diskdump -> ../init.d/diskdump
K99microcode_ctl -> ../init.d/microcode_ctl
S04readahead_early -> ../init.d/readahead_early
S05kudzu -> ../init.d/kudzu
S06cpuspeed -> ../init.d/cpuspeed
S08iptables -> ../init.d/iptables
S09isdn -> ../init.d/isdn
S09pcmcia -> ../init.d/pcmcia
S10network -> ../init.d/network
S12syslog -> ../init.d/syslog
S13irqbalance -> ../init.d/irqbalance
S13portmap -> ../init.d/portmap
S14nfslock -> ../init.d/nfslock
S15mdmmonitor -> ../init.d/mdmmonitor
S18rpcgssd -> ../init.d/rpcgssd
S19rpcidmapd -> ../init.d/rpcidmapd
S19rpcsvcgssd -> ../init.d/rpcsvcgssd
S25netfs -> ../init.d/netfs
S26apmd -> ../init.d/apmd
S26lm_sensors -> ../init.d/lm_sensors
S28autofs -> ../init.d/autofs
S33nifd -> ../init.d/nifd
S34mDNSResponder -> ../init.d/mDNSResponder
S40smartd -> ../init.d/smartd
S44acpid -> ../init.d/acpid
S55cups -> ../init.d/cups
S55sshd -> ../init.d/sshd
S56xinetd -> ../init.d/xinetd
S80sendmail -> ../init.d/sendmail
S85gpm -> ../init.d/gpm
S90crond -> ../init.d/crond
S90xfs -> ../init.d/xfs
S95anacron -> ../init.d/anacron
S95atd -> ../init.d/atd
S96readahead -> ../init.d/readahead
S97messagebus -> ../init.d/messagebus
```

Listing 6-1 (continued)

```
S97rhnsd -> ../init.d/rhnsd
S98cups-config-daemon -> ../init.d/cups-config-daemon
S98haldaemon -> ../init.d/haldaemon
S99local -> ../rc.local
```

Listing 6-1 *(continued)*

All of the scripts in the `rc5.d` directory are symbolic links to the actual scripts that are located in the `/etc/rc.d/init.d/` directory. The use of symbolic links means that the runlevels can be modified by adding or removing symlinks or changing the order the scripts run in the `rc0.d` through `rc6.d` directories without affecting the scripts to which they are linked.

TIP A symbolic link is a type of file that points to another file in the file system.

As you can see, each symbolic link begins with a *K* and a number or an *S* and a number. The *K* links are processes that are killed on that runlevel, and those beginning with an *S* are started. All of the processes beginning with *K* are stopped first and then all the processes beginning with *S* are started. The processes are stopped or started in numerical order, beginning with the lowest number and continuing in increasing order. Processes are stopped by the `/etc/rc.d/init.d/ process stop` command, and started by `/etc/rc.d/init.d/ process start`. If you desire, you can change the stop and start order by changing the numbers in the symbolic links.

TIP You can stop, start, or restart processes on your system by running the command for the process and then typing `stop`, `start`, or `restart` after the command. For example, the command `/etc/rc.d/init.d/vstpd stop` stops the `vsftp` server.

The `init` program has a few more tasks it needs to do before it is finished. One of these tasks is running the `gettys` specified in the `/etc/inittab` file. This provides the six terminals you can use to login to your server. These terminals are in addition to the standard login screen that is provided by the runlevel 5 login scripts. The last thing the `init` program does is run the `/etc/rc.d/rc.local` script. If you want to run any of your own scripts, you can put the calls to them in this file. See the section titled “Starting Programs at System Boot” later in this chapter for more information. Since the system is starting in runlevel 5, you will see a graphical login prompt.

Exploring Runlevels

The term runlevel has been used a few times so far in this chapter and now is a good time to learn more about runlevels and why they are used. There are typically eight runlevels on Linux systems, but we are only interested in the seven used on Fedora Core or Enterprise Linux systems. Each of the runlevels has a set of processes associated with that runlevel that will be started by entering that runlevel. The runlevels on a Fedora Core or Enterprise Linux system and their purpose are:

- 0 — Halt
- 1 — Single-user mode
- 2 — Not used (user-definable)
- 3 — Full multiuser mode (without a graphical user interface, GUI)
- 4 — Not used (user-definable)
- 5 — Full multiuser mode (with a GUI)
- 6 — Reboot

The `/etc/inittab` file controls the default runlevel for the system to use when it boots. You can easily change the runlevel for your system by making the following change.

Changing the System Runlevel

Look for the following line in the `/etc/inittab` file. It will be just below the listing of available runlevels.

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

The number 5 in the line means the system will boot into runlevel 5. If you want to boot into runlevel 3, just change the number 5 to a number 3, so the line now looks like this.

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

The default runlevel on Fedora Core and Enterprise Linux systems is runlevel 5, which provides a full multiuser environment and starts the X Window system to give the users a graphical environment. Runlevel 3 provides a full multiuser system exactly the same as runlevel 5, with the exception of not providing graphical environment for the users. Runlevel 3 is typically used for server systems that don't require a GUI, because running a GUI uses a lot of system resources unnecessarily.

Runlevels 2 and 4 are not used in Fedora Core or Enterprise Linux systems. These runlevels are there so that the user can use them to configure them as they desire. By doing this, you can make your own custom runlevels for testing or whatever suits your purposes.

Runlevel 1 lets you enter single-user mode. This runlevel is useful for troubleshooting or running diagnostics on your system, but it isn't usually set as a default runlevel in the `/etc/inittab` file. If you are having problems with your system, you would usually enter runlevel 1 by entering the command during system boot, as shown here:

1. When the boot screen appears, highlight the kernel you want to change by using the down-arrow key.
2. Press the letter **e**.
3. Scroll to the second line and press **e** again.
4. At the end of the line type **init 1** and press Enter.
5. Press the letter **b** to boot the system into single-user mode.

Starting Programs at System Boot

The file `/etc/rc.d/rc.local` script is the last file run by the `init` command when the system boots. If you want to run additional scripts to configure other devices, you can place the commands for the script into this file. Listing 6-2 shows a typical `/etc/rc.d/rc.local` file with a call to another script called `/usr/bin/novellconfig`. The `novellconfig` script is used to load the modules and configure IPX on the server. You can also pass options to your scripts by entering them after the call to the script.

NOTE The script `novellconfig` is just used as an example and does not actually exist on a default installation. You can create your own scripts and call them whatever you want.

```
#!/bin/sh
#
# This script will be executed *after* all the other init scripts.
# You can put your own initialization stuff in here if you don't
# want to do the full Sys V style init stuff.

touch /var/lock/subsys/local
/usr/bin/novellconfig
```

Listing 6-2 An `/etc/rc.d/rc.local` file calling another program.

Shutting Down the System

If you are running a GUI on your system, shutting down your system is as easy as a few mouse clicks. The shutdown procedure for the GNOME desktop is as follows:

1. Choose Actions ⇨ Logout from the top panel menu. A screen appears asking if you want to logout, restart or shutdown.
2. Click the shutdown radio button.
3. Click OK, and your system begins to shut down.

If you are not running a GUI on your system, the command to shut down is `shutdown` and uses the following syntax:

```
/sbin/shutdown [-t sec.] [-arkhncfF] time [warning message]
```

You can refer to the man page for `shutdown` to get all the details for the `shutdown` command. To access the man page for `shutdown`, type the following command:

```
man shutdown
```

Regardless of the method used to start the `shutdown` command, the process that follows is the same. Any users who are logged in to the system will receive a notice that the system is going down and anyone trying to log in after the `shutdown` command has been given will not be allowed to do so. Running processes on the system are sent the `SIGTERM` signal, which will attempt to cleanly stop the running processes. The `shutdown` program signals the `init` program to change to a different runlevel depending on the `shutdown` command. Runlevel 0 is used to shut down the system, whereas runlevel 6 is used to reboot the system. Some examples of commonly used `shutdown` commands with explanations of their meanings are:

NOTE In most cases, only the root user can use `shutdown` to shut down the system.

The `/sbin/shutdown -h now` command immediately begins the shutdown process and halts the system. The `/sbin/shutdown -r now` command immediately begins the shutdown process and reboots the system.

GRUB Configuration File

When the system boots, GRUB presents a graphical screen showing the operating systems that you can boot. The `/boot/grub/grub.conf` file controls what information is displayed on the graphical screen. This file even controls whether you see the graphical screen at all. Listing 6-3 shows a typical GRUB configuration file. An explanation of the items in the file follows the listing.

```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE: You have a /boot partition. This means that
#         all kernel and initrd paths are relative to /boot/, eg.
#         root (hd0,0)
#         kernel /vmlinuz-version ro root=/dev/VolGroup00/LogVol100
#         initrd /initrd-version.img
#boot=/dev/hda
default=0
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title Fedora Core (2.6.9-1.667)
    root (hd0,0)
    kernel /vmlinuz-2.6.9-1.667 ro root=/dev/VolGroup00/LogVol100 rhgb quiet
    initrd /initrd-2.6.9-1.667.img
```

Listing 6-3 The `/boot/grub/grub.conf` GRUB configuration file.

All lines beginning with a `#` (hash character) are comments and will be ignored. GRUB also ignores blank lines.

The lines following the lines beginning with the hash character and before the line beginning with `title` are the menu interface commands that GRUB uses to display the menu. These lines have the following meanings:

- **default=0** — This command tells GRUB to boot the first listing beginning with `title`. If you had two entries beginning with `title`, you could set this number to 1 to boot the second entry.
- **timeout=5** — This command tells GRUB to boot the default entry after five seconds. To change the time, increase or decrease the number as desired.
- **splashimage=(hd0,0)/grub/splash.xpm.gz** — This command tells GRUB where to look for the splash image it displays for the menu. You can create your own images if you desire. Just be sure to follow the same format as shown here.

- **hiddenmenu** — This command tells GRUB not to display the menu and to boot the default after the timeout expires. You can still see the menu by pressing any key.
- **title** — This command tells GRUB to list a boot name on the menu using the name following the title command. In this example the title is `Fedora Core (2.6.9-1.667)`, which is the name of the operating system and the kernel version number.

The lines following the title are explained here:

- **root (hd0,0)** — This line tells GRUB to boot the system from the first partition of the first hard drive.
- **kernel...** — This line tells GRUB the location of the kernel, as well as passes kernel parameters to the kernel. All locations are relative to the boot partition, so the listing here indicates that the kernel is in the root of the boot partition. If you want to pass kernel parameters to the kernel before it loads, this is the line to add them to. There are already two options shown here. The `rhgb` on the kernel line tells the system to use a graphical boot. Finally, the `quiet` option tells the system not to display detailed information about system booting.
- **initrd...** — This line tells GRUB the location of the initial ramdisk image that is used to load special drivers for the system. All locations are relative to the boot partition, so the listing here indicates that the initial ramdisk image is in the root of the boot partition.

Summary

In this chapter, you learned about the procedure Fedora Core and Red Hat Enterprise Linux uses to start and stop the system. Each step of the boot process was explained in the order in which it occurs, and the programs that run at each step were listed. In addition, you learned about boot loaders and their significance in starting the system. The GRUB boot loader was explained in detail to give you a better understanding of how it works. The role of the `init` process and runlevels in the system were also explained. You took a look at the GRUB configuration file to learn how it works. Finally, you learned how to gracefully bring the system down using the `shutdown` command.

The File System Explained

IN THIS CHAPTER

- Understanding the File System Structure
- Working with Linux-Supported File Systems
- Memory and Virtual File Systems
- Linux Disk Management

This chapter begins with a description of the file system structure and an explanation of the directories and the files they contain. Following the look at the file system structure are the file system commands, essential to proper file system management. In addition to the native Linux file system, Fedora Core and Red Hat Enterprise Linux support many other file system types. This chapter explains the other file system types and ends with a discussion of Linux disk management.

Understanding the File System Structure

Understanding the organization, or layout, of the file system is one of the most important aspects of system administration. For administrators, programmers, users, and installed software, knowing how and where the files are stored on the system is critical for proper system operation. A standard should be in place that specifies locations for specific types of data. Fortunately, Red Hat has chosen to follow the standards outlined in the Filesystem Hierarchy Standard (FHS). This section briefly explains the FHS and its relevance to proper system administration. For the complete standard, refer to `pathname.com/fhs`.

The FHS provides specific requirements for the placement of files in the directory structure. Placement is based on the type of information contained in the file. Two categories of file information exist: shareable or unshareable, and variable or static. Shareable files are files that can be accessed by other hosts, and unshareable files can be accessed only by the local system. Variable files contain information that can change at any time on their own, without anyone actually changing the file. A log file is an example of such a file. A static file contains information that does not change unless a user changes it. Program documentation and binary files are examples of static files. Figure 7-1 shows the organization of the file system on a typical Fedora Core and Red Hat Enterprise Linux system. Following the figure is an explanation of each directory and the types of files it may contain.

As shown in the illustration, the file system is organized in a flat, hierarchical file system. Linux's method of mounting its file systems in a flat, logical, hierarchical method has advantages over the file system mounting method used by Windows. Linux references everything relative to the root file system point `/`, whereas Windows has a different root mount point for every drive.

If you have a `/` partition that fills up in Linux, you can create another file system called `/usr/local` and move your data from `/usr/local` in the original file system to the new file system definition. This practice frees up space on the `/` partition, and is an easy way to bring your system back up to a fully functional state.

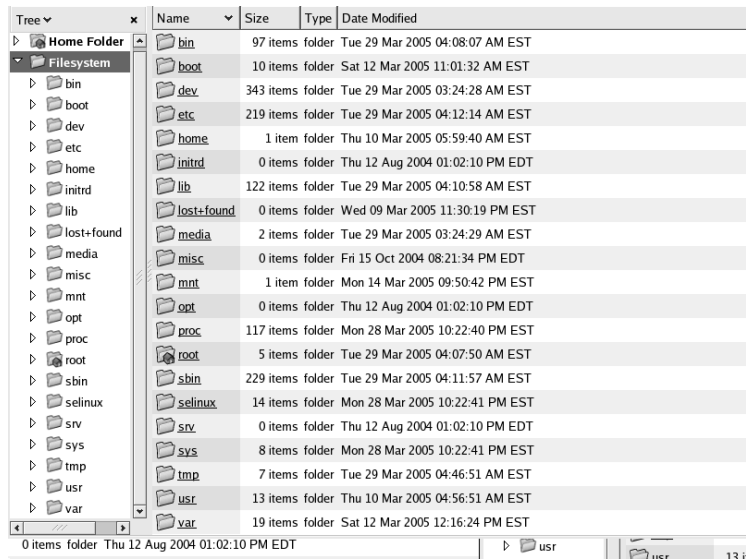


Figure 7-1 The file system organization for a typical Fedora and Red Hat Enterprise Linux system.

This trick wouldn't work on a Windows machine, because Windows maps its file locations to static device disk definitions. You would have to change programs' file references from `c:\` to `d:\`, and so forth. Linux's file system management is another good reason to use Linux on your production servers instead of Windows.

The / Directory

The `/` directory is called the *root* directory and is at the top of the file system structure. In many systems, the `/` directory is the only partition on the system, and all other directories are mounted under it. Figure 7-1 shows a file system with the `/` directory mounted as the only partition, with all other directories contained within it. The primary purpose of the `/` directory is booting the system and correcting any problems that might be preventing the system from booting. According to the FHS, the `/` directory must contain, or have links to, the following directories:

- **bin** — This directory contains command files for use by the system administrator or other users. The `bin` directory cannot contain subdirectories.
- **boot** — On Red Hat systems, this is the directory containing the kernel, the core of the operating system. Also in this directory are files related to booting the system, such as the boot loader and the initial ramdisk.
- **dev** — This directory contains device nodes through which the operating system can access hardware and software devices on the system.
- **etc** — This directory and its subdirectories contain most of the system configuration files. If you have the X Window System installed on your system, the `X11` subdirectory is located here. Networking and system-related files are in the subdirectory `sysconfig`. Another subdirectory of `etc` is the `skel` directory, which holds files used as templates used to create files in users' home directories when the users are created.
- **home** — This directory contains the directories of users on the system. Subdirectories of `home` will be named for the user to whom they belong.
- **initrd** — This directory is used as a mount point when the system is booting. It doesn't contain any data, but it is very important that it be there. This directory is not part of the FHS.
- **lib** — The shared system files and kernel modules are contained in this directory and its subdirectories.

- **media** — This directory contains the mount points for removable media such as floppy drives, CD-ROM drives, and USB devices such as flash memory sticks, which are typically automounted by the system.
- **mnt** — This directory is the location of the mount point for temporary file systems, such as those on floppies or CDs, which traditionally have been manually mounted.
- **opt** — This directory and its subdirectories are often used to hold applications installed on the system.
- **proc** — This directory is a mount point for virtual information about currently running system processes. This directory is empty until the `proc` file system is mounted.
- **root** — This is the home directory of the root user. Don't confuse this with the `/` directory, which has the same name.
- **sbin** — Contained in this directory are system binaries used by the system administrator or the root user.
- **selinux** — This directory is similar to the `/proc` directory in that it contains information about the selinux stored in the memory of the running kernel.
- **srv** — This directory is intended to hold site-specific data for system provided services.
- **sys** — This directory is the mount point for a virtual file system of type `sysfs` that is used to hold information about the system and devices.
- **tmp** — This directory contains temporary files used by the system.
- **usr** — This directory is often mounted on its own partition. It contains shareable, read-only data. Subdirectories can be used for applications, typically under `/usr/local`.
- **var** — Subdirectories and files under `var` contain variable information, such as system logs and print queues.

CAUTION Never remove the `/initrd/` directory. The system will not boot, and you will see a kernel panic error message.

Working with Linux-Supported File Systems

Linux is a very flexible operating system that has a long history of interoperability with other systems on a number of different hardware platforms. A

consequence of this friendliness to other operating systems is that Linux can read and write to several different file systems that originated with other operating systems much different from Linux. This section details the different file systems supported and where they originated.

One reason that Linux supports so many file systems is the design of its Virtual File Systems (VFS) layer. The VFS layer is a data abstraction layer between the kernel and the programs in userspace that issue file system commands.

NOTE Programs that run inside the kernel are in *kernel space*. Programs that don't run inside the kernel are in *userspace*.

The VFS layer avoids duplication of common code between all file systems. It provides a fairly universal backward compatible method for programs to access all of the different forms of file support. Only one common, small API set accesses each of the file system types, to simplify programming file system support.

Support for these file systems comes standard in Red Hat Enterprise Linux. They are compiled into the kernel by default. If for some reason your kernel does not currently support these file systems, a kernel recompile with the proper options turned on should enable you to access all these file systems.

ext3

The extended 3 file system is a new file system introduced in Red Hat 7.2. `ext3` provides all the features of `ext2`, and also features journaling and backward compatibility with `ext2`. The backward compatibility enables you to still run kernels that are only `ext2`-aware with `ext3` partitions. You can also use all of the `ext2` file system tuning, repair, and recovery tools with `ext3`.

You can upgrade an `ext2` file system to an `ext3` file system without losing any of your data. This upgrade can be done during an update to the operating system.

`ext3` support comes in kernels provided with the latest Fedora and Red Hat distributions. If you download a kernel from somewhere else, you need to patch the kernel to make it `ext3` aware, with the kernel patches that come from the Red Hat FTP site. It is much easier to just stick with kernels from Red Hat.

`ext3`'s journaling feature speeds up the amount of time it takes to bring the file system back to a sane state if it's not been cleanly unmounted (that is, in the event of a power outage or a system crash).

Under `ext2`, when a file system is uncleanly mounted, the whole file system must be checked. This takes a long time on large file systems. On an `ext3` system, the system keeps a record of uncommitted file transactions and applies only those transactions when the system is brought back up. So, a complete file system check is not required, and the system will come back up much faster.

A cleanly unmounted `ext3` file system can be mounted and used as an `ext2` file system. This capability can come in handy if you need to revert to an older kernel that is not aware of `ext3`. The kernel sees the `ext3` file system as an `ext2` file system.

`ext3`'s journaling feature involves a small performance hit to maintain the file system transaction journal. Therefore, it's recommended that you use `ext3` mostly for your larger file systems, where the `ext3` journaling performance hit is made up for in time saved by not having to run `fsck` on a huge `ext2` file system.

ext2

`ext2` was the standard file system for Linux until the introduction of `ext3`. The `ext2` implementation has not changed much since it was introduced with the 1.0 kernel back in 1993. Since then, a few new features have been added. One of these was sparse super blocks, which increase file system performance.

`ext2` was designed to make it easier for new features to be added, so that it can constantly evolve into a better file system. Users can take advantage of new features without reformatting their old `ext2` file systems. `ext2` has the added bonus of being designed to be POSIX-compliant. New features that are still in the development phase are access control lists, undelete, and on-the-fly compression.

`ext2` is flexible, can handle file systems up to 4 TB, and supports long filenames up to 1012 characters. In case user processes fill up a file system, `ext2` normally reserves about 5 percent of disk blocks for exclusive use by root so that root can easily recover from that situation. Modern Red Hat boot and rescue diskettes now use `ext2` instead of `minix`.

reiserfs

The Reiser file system is a journaling file system designed for fast server performance, especially in directories containing thousands of files. It is more space efficient than most other file systems, because it does not take up a minimum of one block per file. If you write a bunch of really small files to disk, `reiserfs` squeezes them all into one block instead of writing one small file to one block like other file systems do. `reiserfs` also does not have fixed space allocation for inodes, which saves about 6 percent of your disk space.

SystemV

Linux currently provides read support for SystemV partitions, and write support is experimental. The SystemV file system driver currently supports AFS/EAFS/EFS, Coherent FS, SystemV/386 FS, Version 7 FS, and Xenix file systems.

ufs

ufs is used in Solaris and early BSD operating systems. Linux provides read support, and write support is experimental.

FAT

FAT is one of a few different file systems used with Windows over the years. Almost every computer user has used FAT at one time or another, since it was the sparse base operating system at the heart of all Windows operating systems.

FAT was originally created for QDOS and used on 360K (double density, double-sided) floppy disks. Its address space has since been extended from 12 bit to 32 bit, so it can handle very large file systems. There have been four versions of FAT since its beginnings: FAT12, FAT16, VFAT, and FAT32. Nowadays, it's possible to create FAT32 file systems over a terabyte in size.

NOTE Do not confuse a FAT file system with a FAT32 file system. They are named similarly but are two different beasts!

NTFS

NTFS is the next generation of HPFS. It comes with all versions of Microsoft operating systems beginning with Windows NT. Unlike FAT, it is a b-tree file system, meaning it has a performance and reliability advantage, including journaling, and support for encryption and compression, over FAT.

IBM JFS

IBM JFS is an easy-to-use journaling file system created by IBM. It is designed for high-throughput server environments. This is the same file system that will be provided in AIX version 5.1. Linux support for JFS was written by IBM. IBM has contributed quite a bit of code to the Linux cause and is a staunch supporter of Linux. It has also decided to make Linux its main server file system in the future.

SGI XFS

SGI's Extended File System (XFS) is SGI's newest file system for all Silicon Graphics systems, from workstations to its supercomputer line (before it sold that line to Terra computers.) It has been available for use on Linux since May 2001.

XFS is designed for high performance. It rapidly recovers from system crashes and can support extremely large disk farms (it can handle files as large as a million terabytes.) It is one of a few journaling file systems that have had a proven track record in production environments for several years now.

NOTE Its other features include access control lists, volume management, guaranteed rate I/O, and journaling for faster recovery. XFS can be backed up while still in use, which comes in handy since it reduces system administration time. This is a fast file system, and now you can read and write to and from it with your Red Hat Linux machine.

Nonstandard Linux File Systems

Support for these file systems needs to be explicitly compiled into the Linux kernel, since kernel support for them is not configured by default.

FREEVxFS

VxFS is the Veritas file system developed by the Veritas Corporation. It is used in SCO UnixWare, HP-UX, Solaris, and other systems. Some of its features include access control lists, journaling, online backup, and support for files up to 2 TB.

Three different versions of VxFS are in use. Version 1 is the original VxFS, which is not commonly used anymore. Version 2 includes support for filesets and dynamic inode allocation. Version 4 is the latest version, and it supports quotas and large files.

GNU utilities available for Linux called VxTools can read VxFS versions 2 and 4. The tools included in the VxTools package are vxmount, vxumount, vxls, vxcat, vxidump, vxcd, and vxpwd. Currently there is only read support in Linux for VxFS file systems.

GFS

GFS is Sistina's Global File System. It is a clustered journaling file system for SANs that enables multiple servers to have read/write access to a single file system on shared SAN devices.

GFS is scalable, since storage devices and servers can be added without taking the system down or taking the disks offline. It also makes a single image of all the data in the SAN, so that if a server fails it can be removed and replaced while the load is rebalanced amongst the remaining servers.

In a proper cluster setup, all nodes in the cluster share the same storage devices through a fiber channel, SCSI hookup, or network block device. Each node sees the file system as being local to their machine, and GFS synchronizes files across the cluster. GFS is fully symmetric, so no server is a bottleneck or single point of failure. GFS uses regular UNIX-style file semantics.

Memory and Virtual File Systems

These file systems do not exist on disk in the same way that traditional file systems do. They either exist entirely in system memory or they are virtual, because they are an interface to system devices, for example.

cramfs

`cramfs` is designed to cram a file system onto a small flash memory device, so it is small, simple, and able to compress things well. The largest file size is 16 MB, and the largest file system size is 256 MB.

Since `cramfs` is so compressed, it isn't instantly updateable. The `mkcramfs` tool needs to be run to create or update a `cramfs` disk image. The image is created by compressing files one page at a time, so this enables random page access. The metadata is not compressed, but it has been optimized to take up much less space than other file systems. For example, only the low 8 bits of the GID are stored. This saves space but also presents a potential security issue.

tmpfs

`tmpfs` is structured around the idea that whatever is put in the `/tmp` file system is accessed again shortly. `tmpfs` exists solely in memory, so what you put in `/tmp` doesn't persist between reboots.

Mounting a special-purpose file system on `/tmp` as an in-memory file system is a performance boost but is rarely done in Linux because of the performance available from the traditional Linux file system. But for those who feel that they need the performance gains from storing `/tmp` in memory, this option is now available in Linux.

ramfs

`ramfs` is basically `cramfs` without the compression.

romfs

This is a read-only file system that is mostly used for the initial ramdisks of installation disks. It was designed to take up very little space, so you could fit a kernel and some useful code into a small boot disk, without having the file system overhead taking up too much precious space in memory or on the disk.

The kernel on the disk has only this file system linked into it, and it can load any modules it needs later, after bootup. After the kernel is loaded, it can call other programs to help determine what SCSI drivers are needed, if any, or what IDE or floppy drives should be accessed after bootup. This method is perfect for rescue diskettes or installation diskettes, where only a very bare minimum kernel needs to be loaded into memory, so after the initial boot it can then load from a CD-ROM whatever `ext2` modules or other drivers are necessary to mount the system's regular drives.

The `romfs` file system is created with a program called `genromfs`.

proc

`proc` is a virtual file system that acts as an interface to the kernel's internal data structures. `proc` can be used to get detailed information about a system's hardware and to change kernel parameters at runtime. Even the process listing command, `ps`, gets its information from the `proc` file system. The kernel parameters can be changed with the `sysctl` command.

Proc Software Information

The `/proc` directory contains a great deal of information about your currently running system software. If you look at the `/proc` directory on Linux, you see one subdirectory for each process running on the system. The subdirectories are named after the process's ID (PID) number. Each of those subdirectories has several standard files, and each of them gives you a different set of information.

The status file in those `proc` directories contains process status in human-readable format. So, if you want to see the status of your `ssh` server, you first need to know the `ssh` server's PID number. You can find this number in a few different ways. One easy way is to look at a process listing and `grep` for the string `ssh`. The output should look like the lines shown in Listing 7-1.

```
[terry@main terry]$ ps -elf | grep ssh
140 S root      933      1 0 69 0   -   664 do_sel Oct23 ?      00:00:01
/usr/sbin/sshd
140 S root      14807   933  0 69 0   -   882 do_sel 18:36 ?      00:00:00
/usr/sbin/sshd
000 S vnavrat 14883 14808  0 71 0   -   434 pipe_w 18:52 pts/10  00:00:00
grep ssh
```

Listing 7-1 Finding the process ID (PID) number.

The process table contains multiple hits for `ssh`, since there is a master `sshd` process, and one `sshd` process is spawned for each `ssh` session currently open. The first line is the master `sshd` server process. You can tell because its parent process ID is 1, also known as the `init` process that spawns all processes at boot time, and is responsible for respawning important server processes that die during runtime. The second line is an `ssh` daemon handling an incoming `ssh` connection, evident because it lists the previous `ssh` process as its parent. The final line lists the `grep` process that you just ran, so you can disregard that line.

You should look at the status of the master `ssh` daemon, which, as you saw previously, is running with a PID of 933. So, `cd` to the `/proc/933` directory, and take a look at the status file in that directory. The output appears in Listing 7-2.

```
[terry@main terry]$ less /proc/933/status
Name:      sshd
State:     S (sleeping)
Pid:       933
PPid:      1
TracerPid: 0
Uid:       0      0      0      0
Gid:       0      0      0      0
FDSize:    32
Groups:
VmSize:    2656 kB
VmLck:     0 kB
VmRSS:     1236 kB
VmData:    116 kB
VmStk:     16 kB
VmExe:     240 kB
VmLib:     2176 kB
SigPnd:    0000000000000000
SigBlk:    0000000000000000
SigIgn:    8000000000001000
SigCgt:    0000000000016005
CapInh:    0000000000000000
CapPrm:    00000000fffffeff
CapEff:    00000000fffffeff
```

Listing 7-2 Viewing the status information of a running process.

Other useful files in the `/proc/PID` directory and their contents are:

- **cmdline** — Contains the process's command line arguments
- **cpu** — Contains the current and last CPU on which the process was executed
- **cwd** — Contains a link to the process's current working directory
- **environ** — Contains values of the process's environmental variables
- **exe** — Contains a link to the process's executable
- **fd** — A directory that contains all the process's file descriptors
- **maps** — Contains memory maps to the process's executables and library files
- **mem** — Contains the memory held by this process
- **root** — Contains a link to the root directory of the process
- **stat** — Contains the process status
- **statm** — Contains the process memory status information
- **status** — Contains the process status in human-readable format

Proc Hardware Information

As mentioned previously, the `/proc` directory also contains some useful hardware information. This information comes in handy when you compile a new kernel. If you've forgotten the specific details about your hardware, you can look through the files in the `/proc` directory to get information about what's installed and running on your Linux machine.

If you suspect that you're having hardware problems due to an interrupt request (IRQ) conflict, you can also see your hardware's interrupts by looking at the `/proc/interrupts` file.

The interrupts file from my desktop machine at work is shown below. Each number corresponds to an IRQ. The acronyms at the end of the IRQ listing are NMI (Non-Maskable Interrupt), LOC (local interrupt counter of the internal APIC of each CPU), and ERR. ERR is a counter that starts out at 0 at boot time and is incremented each time there is an error in the IO-APIC bus. The IO-APIC bus connects the CPUs in an SMP system. When an error happens, the information is immediately retransmitted, so you shouldn't worry too much about a moderate number of errors in this field. Listing 7-3 shows the `/proc/interrupts` information.

```
[terry@main terry]$ less /proc/interrupts
CPU0
0:    9720704      XT-PIC  timer
1:    30515       XT-PIC  keyboard
2:      0         XT-PIC  cascade
5:   9869566      XT-PIC  Crystal audio controller
8:      1         XT-PIC  rtc
11:   1233943     XT-PIC  usb-uhci, eth0
12:   6822220     XT-PIC  PS/2 Mouse
14:    77739      XT-PIC  ide0
15:   2694731     XT-PIC  ide1
NMI:      0
LOC:   9720557
ERR:      0
MIS:      0
```

Listing 7-3 Viewing the `/proc/interrupts` information.

In the main `/proc` directory, quite a few files contain detailed information on your system hardware. The kind of details listed are things such as what hardware it is, the model, and the manufacturer.

Listing 7-4 shows the contents of the `cpuinfo` file in `proc`. This tells you what kind of processor you have, and most importantly, how fast it is.

```
[terry@main terry]$ less /proc/cpuinfo
processor       : 0
vendor_id     : AuthenticAMD
cpu family    : 6
model         : 6
model name    : AMD Athlon(tm) XP 1800+
stepping      : 2
cpu MHz       : 1535.822
cache size    : 256 KB
fdiv_bug      : no
hlt_bug       : no
f00f_bug      : no
coma_bug      : no
fpu           : yes
fpu_exception : yes
cpuid level   : 1
wp            : yes
flags         : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge
mca cmov
pat pse36 mmx fxsr sse
bogomips      : 3022.84
```

Listing 7-4 Viewing the contents of the `/proc/cpuinfo` file.

Some important `/proc` files are:

- **`/proc/cpuinfo`** — Contains info about the CPU
- **`/proc/interrupts`** — Tells you what interrupts are in use
- **`/proc/scsi`** — A directory that contains information about SCSI devices
- **`/proc/parport`** — Contains info about the parallel ports on your system
- **`/proc/tty`** — A directory that contains info about `ttys` that are available and in use
- **`/proc/acpi`** — Contains power management information
- **`/proc/bus`** — A directory that contains bus-specific information
- **`/proc/devices`** — Lists available character and block devices
- **`/proc/dma`** — Lists used DMS channels
- **`/proc/filesystems`** — Lists supported file systems
- **`/proc/fs`** — A directory that contains file system parameters
- **`/proc/ide`** — A directory that contains information about the IDE subsystem
- **`/proc/ioports`** — Contains information about system I/O port usage
- **`/proc/modules`** — Contains a list of currently loaded modules
- **`/proc/net`** — Contains networking information
- **`/proc/uptime`** — Contains the system uptime
- **`/proc/version`** — Contains the system version

`/dev/pts`

`/dev/pts` is a lightweight version of `devfs`. Instead of having all the device files supported in the virtual file system, it provides support for only virtual pseudoterminal device files. `/dev/pts` was implemented before `devfs`.

`devfs`

The Device File System (`devfs`) is another way to access “real” character and block special devices on your root file system. The old way used major and minor numbers to register devices. `devfs` enables device drivers to register devices by name instead. `devfs` is deprecated in the 2.6 kernel in favor of `udev`.

sysfs

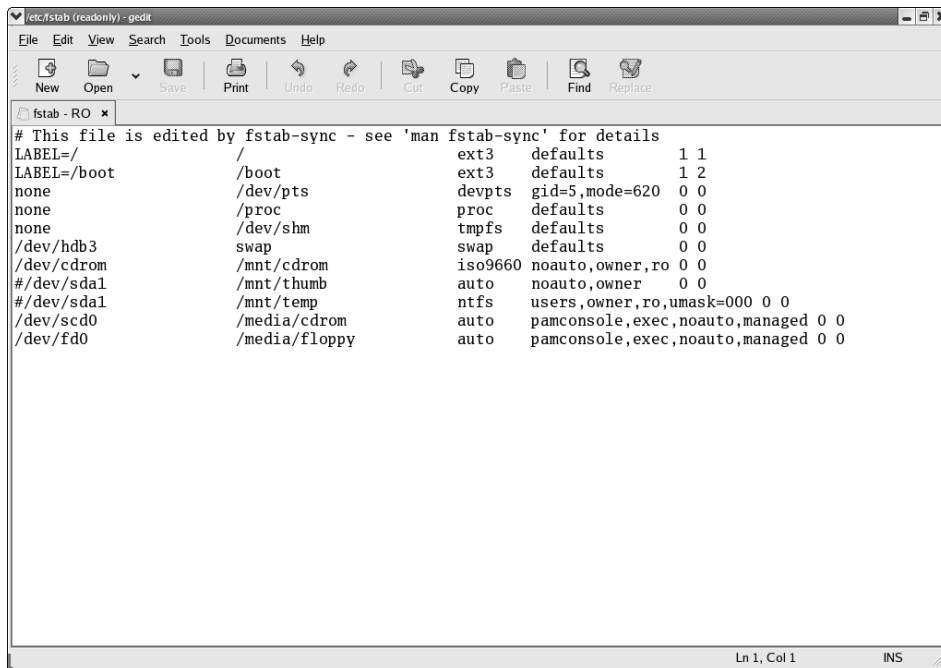
`sysfs` is a virtual file system that acts as an interface to the kernel's internal data structures. Information is stored in the `/sys` directory and can be used to get details about a system's hardware and to change kernel parameters at run-time. Information in the `/sys` directory is similar to the information provided in the `/proc` directory and can be accessed in a similar fashion.

Linux Disk Management

This section explains some basics about disk partitioning and disk management under Linux. To see how your Linux disks are currently partitioned and what file systems are on them, look at the `/etc/fstab` file.

In Figure 7-2, you can see what a simple `/etc/fstab` file looks like.

TIP To see how your Linux disks are currently partitioned and what file systems are on them, look at the `/etc/fstab` file. You could also use the `fdisk -l` command to obtain partition information about your disks.



The screenshot shows a text editor window titled "etc/fstab (readonly) - gedit". The menu bar includes File, Edit, View, Search, Tools, Documents, and Help. The toolbar contains icons for New, Open, Save, Print, Undo, Redo, Cut, Copy, Paste, Find, and Replace. The file name "fstab - RO" is shown in the tab. The content of the file is as follows:

```
# This file is edited by fstab-sync - see 'man fstab-sync' for details
LABEL=/                                /                                ext3      defaults    1 1
LABEL=/boot                            /boot                            ext3      defaults    1 2
none                                    /dev/pts                         devpts    gid=5,mode=620 0 0
none                                    /proc                            proc      defaults    0 0
none                                    /dev/shm                         tmpfs     defaults    0 0
/dev/hdb3                               swap                             swap      defaults    0 0
/dev/cdrom                              /mnt/cdrom                       iso9660   noauto,owner,ro 0 0
#/dev/sda1                              /mnt/thumb                       auto      noauto,owner 0 0
#/dev/sda1                              /mnt/temp                        ntfs      users,owner,ro,umask=000 0 0
/dev/scd0                               /media/cdrom                     auto      pamconsole,exec,noauto,managed 0 0
/dev/fd0                                /media/floppy                    auto      pamconsole,exec,noauto,managed 0 0
```

The status bar at the bottom indicates "Ln 1, Col 1" and "INS".

Figure 7-2 The contents of the `/etc/fstab` file.

Disk Partitioning on an x86 Machine

When disk partitioning on an x86 PC, you need to be mindful of the limitations present in the x86 architecture. You are allowed to create four primary partitions. Primary partitions are the only partitions that are bootable. You can create more partitions if you make extended partitions.

Extended partitions are set into a primary partition. So, if you choose to make extended partitions, you are allowed to make only three primary partitions for operating system use, and the fourth partition is dedicated to hosting the extended partitions.

Mounting Other OS Partitions/Slices

Not only can Linux read other operating systems' file systems; it can mount disk drives from other systems and work with their partition tables. However, it is necessary to compile two options into the kernel to do this. You must have the file system support and the file partitioning support turned on in the kernel. Usually file system support is compiled as a module by default, but disk partition support usually has to be explicitly compiled.

Some common partitioning schemes that Linux supports are x86 partitions, BSD disklabel, Solaris x86, Unixware, Alpha, OSF, SGI, and Sun.

Mounting other operating systems' partitions is helpful if you need to put a Sun hard disk into a Linux machine, for example. You may need to do this if the original Sun system has gone bad, and you need to recover the information that was on its disk, or if it's the target of a forensic computer crime investigation, and you need to copy the disk contents to another machine to preserve evidence. This method takes advantage of the fact that copying a large amount of data is much faster across a SCSI connection than across a network.

If you need to copy a large amount of raw disk data across a network, you can use the Network Block Device, which enables other machines to mount a disk on your machine as if it were on their machine.

TIP When running the Network Block Device, make sure that you have the appropriate partition support compiled into the kernel. For more information about NBD refer to it.uc3m.es/~ptb/nbd.

Metadevices

Virtual block devices that are made up of other block devices are referred to in this book as a metadevice. An example of a metadevice is a disk array that makes many disks look like one large disk. When a disk that's mounted as a

regular block device dies, then the data on it becomes unavailable. If a disk dies in a metadvice, the metadvice is still up. As long as the criteria are met for the minimum number of working devices in the metadvice, the metadvice still functions.

Logical Volumes

Logical Volume Manager (LVM) enables you to be much more flexible with your disk usage than you can be with conventional old-style file partitions. Normally if you create a partition, you have to keep the partition at that size indefinitely.

For example, if your system logs have grown immensely, and you've run out of room on your `/var` partition, increasing a partition size without LVM is a big pain. You would have to get another disk drive, create a `/var` mount point on there too, and copy all your data from the old `/var` to the new `/var` disk location. With LVM in place, you could add another disk, create a physical volume, and then add the physical volume to the volume group that contains the `/var` partition. Then you'd use the LVM file system resizing tool to increase the file system size to match the new partition size.

Normally, you might think of disk drives as independent entities, each containing some data space. When you use LVMs, you need a new way of thinking about disk space. First, you have to understand that space on any disk can be used by any file system. A Volume Group is the term used to describe various disk spaces (either whole disks or parts of disks) that have been grouped together into one volume.

The way it works is like this. First you need to have a physical volume which is then divided into Volume groups that are then combined to form logical volumes. Logical volumes are akin to the historic idea of partitions. You can then use a file system creation tool such as `fdisk` to create a file system on the logical volume. The Linux kernel sees a logical volume in the same way it sees a regular partition.

NOTE When the system is installed, LVM is enabled by default and you will need to use the LVM tools described here to make changes to your logical volumes. You can, if you desire, choose not to use logical volumes during the system installation.

In Fedora Core and Enterprise Linux, LVM has been updated to LVM2. The basic syntax for using the `lvm` command is:

```
lvm <command> file
```

There are many commands available when using LVM. You can obtain a complete listing of the commands by entering `lvm help` at a command prompt. You will see the list shown in Listing 7-5.

```
dumpconfig    Dump active configuration
formats      List available metadata formats
help         Display help for commands
lvchange     Change the attributes of logical volume(s)
lvcreate     Create a logical volume
lvdisplay    Display information about a logical volume
lvextend     Add space to a logical volume
lvmdiskscan  List devices that may be used as physical volumes
lvmsadc      Collect activity data
lvmsar       Create activity report
lvreduce     Reduce the size of a logical volume
lvremove     Remove logical volume(s) from the system
lvrename     Rename a logical volume
lvresize     Resize a logical volume
lvs         Display information about logical volumes
lvscan      List all logical volumes in all volume groups
pvchange    Change attributes of physical volume(s)
pvcreate    Initialize physical volume(s) for use by LVM
pvdata     Display the on-disk metadata for physical volume(s)
pvdisplay   Display various attributes of physical volume(s)
pvmove     Move extents from one physical volume to another
pvremove   Remove LVM label(s) from physical volume(s)
pvresize   Resize a physical volume in use by a volume group
pvs        Display information about physical volumes
pvscan     List all physical volumes
segtypes   List available segment types
vgcfgbackup Backup volume group configuration(s)
vgcfgrestore Restore volume group configuration
vgchange   Change volume group attributes
vgck       Check the consistency of volume group(s)
vgconvert  Change volume group metadata format
vgcreate   Create a volume group
vgdisplay  Display volume group information
vgexport   Unregister volume group(s) from the system
vgextend   Add physical volumes to a volume group
vgimport   Register exported volume group with system
vgmerge    Merge volume groups
vgmknodes  Create special volume group file devices in /dev
vgreduce   Remove physical volume(s) from a volume group
vgremove   Remove volume group(s)
vgrename   Rename a volume group
vgs        Display information about volume groups
vgscan     Search for all volume groups
vgsplit    Move physical volumes into a new volume group
version    Display software and driver version information
```

Listing 7-5 Output from the `lvm help` command.

You can get more detailed help about each command by entering `lvm help` and the name of the command for which you want help. For example, to find out more about the `pvccreate` command enter `lvm help pvccreate` at a terminal prompt to go to the `pvccreate` help page.

Let's take a look at using a few of the commands. To get a listing of the physical volumes on the system enter `lvm pvdisplay` at a terminal prompt. You will see output similar to Listing 7-6.

```
--- Physical volume ---
PV Name           /dev/hda2
VG Name           VolGroup00
PV Size           9.41 GB / not usable 0
Allocatable       yes
PE Size (KByte)   32768
Total PE          301
Free PE           1
Allocated PE      300
PV UUID           mwGHdm-M7no-X118-D8kE-i5YS-btzV-w8Og1f
```

Listing 7-6 Using the `pvdisplay` command to get a listing of system physical volumes.

To get a list of the logical volumes on your system, enter `lvm lvdisplay` at a terminal prompt. You will see a listing similar to Listing 7-7.

```
--- Logical volume ---
LV Name           /dev/VolGroup00/LogVol00
VG Name           VolGroup00
LV UUID           QAVcFn-Jrjy-7sAs-0zih-vyTk-SWqX-fVC1M6
LV Write Access    read/write
LV Status          available
# open            1
LV Size            9.00 GB
Current LE         288
Segments           1
Allocation         inherit
Read ahead sectors 0
Block device       253:0
```

Listing 7-7 Using the `lvdisplay` command to see the logical volumes on the system.

One last example: To get a listing of the volume groups on your system, enter `lvm vgdisplay` at a terminal prompt. You will see a listing similar to Listing 7-8.


```

--- Volume group ---
VG Name                VolGroup00
System ID
Format                 lvm2
Metadata Areas         1
Metadata Sequence No   3
VG Access               read/write
VG Status               resizable
MAX LV                  0
Cur LV                 2
Open LV                 2
Max PV                  0
Cur PV                 1
Act PV                  1
VG Size                 9.41 GB
PE Size                 32.00 MB
Total PE                301
Alloc PE / Size         300 / 9.38 GB
Free PE / Size           1 / 32.00 MB
VG UUID                 KKrG4a-HaUw-7Fpo-DyL5-sU8F-wFcq-nnGC1Q

```

Listing 7-8 Using the `vgdisplay` command to see the volume groups on the system.

By now you should have a pretty good idea of the syntax to follow and how to use some of the commands when working with logical volumes.

RAID

RAID is an acronym for Redundant Array of Inexpensive, or Independent (depending on who you ask), Disks. There are two types of RAID that can be used on computer systems. These types are hardware RAID and software RAID. In addition, there are six different RAID levels commonly used regardless of whether hardware or software RAID is used. A brief explanation of hardware and software RAID is in order. Following this explanation is a description of the six RAID levels.

- **Hardware Raid** — In hardware RAID the disks have their own RAID controller with built-in software that handles the RAID disk setup, and I/O. The controller is typically a card in one of the system's expansion slots, or it may be built onto the system board. The hard RAID interface is transparent to Linux, so the hardware RAID disk array looks like one giant disk. The operating system does not control the RAID level used, it is controlled by the hardware RAID controller. Most dedicated servers use a hardware RAID controller.

- **Software RAID** — In software RAID there is no RAID controller card. The operating system is used to set up a logical array, and the operating system controls the RAID level used by the system.

NOTE Software RAID must be configured during system installation. Refer to Chapter 3 for more details about configuring RAID on your system.

As mentioned earlier, there are six RAID levels that can be used, but in actual practice usually only three of them are used. And of these three one doesn't provide redundancy even though it is identified as a RAID level. The three most commonly used RAID levels are:

- **RAID level 0** — This RAID level requires at least two disks and uses a method called striping that writes data across both drives. There is no redundancy provided by this level of RAID, since the loss of either drive makes it impossible to recover the data. This level of RAID does give a speed increase in writing to the disks.
- **RAID level 1** — This RAID level requires at least two disks and uses a method called mirroring. With mirroring, the data is written to both of the drives. So, each drive is an exact mirror of the other one, and if one fails the other still holds all the data. There are two variants to level 1 with one variant using a single disk controller that writes to both disks as described above. The other variant uses two disk controllers, one for each disk. This variant of RAID level 1 is known as duplexing.
- **RAID level 5** — This RAID level, which is the most widely used, requires at least three disks and uses striping to write the data across the two disks similarly to RAID level 1. But unlike RAID level 1, this level of RAID uses the third disk to hold parity information that can be used to reconstruct the data from either, but not both, of the two disks after a single disk failure.

There are some system files that you can use to get information about RAID on your system. You can look in `/etc/raidtab` to get information about the system's RAID configuration. RAID devices are identified in Fedora Core and Enterprise Linux as `md` devices. The `/etc/raidtab` file lists which block devices are associated with the `md` device.

NOTE The commands discussed here are only useful when using software RAID. Hardware RAID is invisible to the operating system.

You can also look at the contents of the `/proc/mdstat` file to get information about the running status of your `md` devices.

Also available to you are several command-line tools. You can use `lsraid` to list and query `md` devices as well. This command is similar to the `ls` command and more information is available by reading the `lsraid` man page. You can also use the `man` command with the following RAID commands:

- **raidstart** — This command will start an existing RAID device.
- **raidstop** — This command will stop an existing RAID device.
- **raidreconf** — This command is used to add disks to an existing array or to convert an array to a new type.

Summary

In this chapter you learned how Fedora Core and Enterprise Linux provide support for many file systems. Linux supports those from other operating systems, remote file systems, memory file systems, CD-ROM file systems, virtual file systems, and metadvice file systems. This makes Linux very good at managing and accessing any file or file systems that you may ever come across in a multiplatform environment.

Examining the System Configuration Files

IN THIS CHAPTER

- Examining the System Configuration Files
- Examining the `/etc/sysconfig/` Directory
- Examining the Network Configuration Files
- Managing the init Scripts

This chapter describes the file system and configuration files in a typical Fedora Core and Red Hat Enterprise Linux server.

The system configuration files in the `/etc` directory are the first places a system administrator goes after installing a system to set it up. The `/etc` directory is probably the most often visited directory by a system administrator after his or her own home directory and `/var/log`.

All of the systemwide important configuration files are found either in `/etc` or in one of its many subdirectories. An advantage to keeping all system configuration files under `/etc` is that it's easier to restore configurations for individual programs, as opposed to having all the system's configurations rolled up into a monstrous registry hive as some operating systems do.

CAUTION Be vigilant that your files in `/etc` are modifiable only by appropriate users. Generally, this means being modifiable only by root.

Because these files are so important and their contents so sensitive (everything from users' hashed passwords to the host's SSH key are stored in `/etc`), it is important to keep the file permissions set properly on everything in `/etc`. Almost all files should be owned by root, and *nothing* should be world-writable.

Most files should have their file permissions set to user readable and writable, and group and world readable, like this:

```
-rw-r--r--    1 root    root          172 Aug  6 02:03 hosts
```

Some notable exceptions are files such as `/etc/shadow`, where users' hashed passwords are stored, and `/etc/wvdial.conf`, which stores dial-up account names and passwords. These files' permissions should be set to owned by root, and read by root only, like this:

```
-rw-----    1 root    root          1227 Sep  2 13:52 /etc/shadow
```

The `/etc/sysconfig` directory contains configuration scripts written and configured by Red Hat and Red Hat administration tools as well as files containing variable settings used by system startup scripts. `/etc/sysconfig` contains both system and networking configuration files. Putting these files in `/etc/sysconfig` distinguishes them from other `/etc` configuration files not designed by Red Hat. You should keep these files in a separate directory so that the risk of other developers writing configuration files with the same names and putting them in the same place as existing configuration files is reduced.

Examining the System Configuration Files

The Red Hat system configuration files can fall within a few different functions. Some specify system duties, such as logging and automatically running programs with cron. Some set default configurations for important programs such as Sendmail and Bash. And many other system configuration files are responsible for arranging the appearance of the system, such as setting the colors that show up when a directory listing is shown and the banners that pop up when someone logs in. This section discusses the more important system configuration files on your Red Hat system.

Systemwide Shell Configuration Scripts

These files determine the default environment settings of system shells and what functions are started every time a user launches a new shell.

The files discussed next are located in `/etc`. These configuration files affect all shells used on the system. An individual user can also set up a default configuration file in his or her home directory that affects only his or her shells. This ability is useful in case the user wants to add some extra directories to his or her path or some aliases that only he or she can use.

When used in the home directory, the names are the same, except they have a `.` in front of them. So `/etc/bashrc` affects bash shells systemwide, but `/home/kelly/.bashrc` affects only the shells that the user `kelly` starts.

Shell Config Scripts: *bashrc*, *csch.cshrc*, *zshrc*

Bashrc is read by bash; *csch.cshrc* is read by *tcsh*; and *zshrc* is read by *zsh*. These files are read every time a shell is launched, not just upon login, and they determine the settings and behaviors of the shells on the system. The following are places to put functions and aliases.

- **profile** This file is read by all shells except *tcsh* and *csch* upon login. bash falls back to reading it if there is no `bash_profile`. Zsh looks for `zprofile`, but if there is none, it reads `profile` as well. Listing 8-1 shows a typical `/etc/profile` file.

```
# /etc/profile
# System wide environment and startup programs, for login setup
# Functions and aliases go in /etc/bashrc

pathmunge () {
    if ! echo $PATH | /bin/egrep -q "^(|:)$1($|:)" ; then
        if [ "$2" = "after" ] ; then
            PATH=$PATH:$1
        else
            PATH=$1:$PATH
        fi
    fi
}

# Path manipulation
if [ `id -u` = 0 ]; then
    pathmunge /sbin
    pathmunge /usr/sbin
    pathmunge /usr/local/sbin
fi

pathmunge /usr/X11R6/bin after

# No core files by default
ulimit -S -c 0 > /dev/null 2>&1

USER=`id -un`
LOGNAME=$USER
MAIL="/var/spool/mail/$USER"
```

Listing 8-1 A typical `/etc/profile` file. (continued)

```

HOSTNAME=`/bin/hostname`
HISTSIZE=1000

if [ -z "$INPUTRC" -a ! -f "$HOME/.inputrc" ]; then
    INPUTRC=/etc/inputrc
fi

export PATH USER LOGNAME MAIL HOSTNAME HISTSIZE INPUTRC

for i in /etc/profile.d/*.sh ; do
    if [ -r "$i" ]; then
        . $i
    fi
done

unset i
unset pathmunge

if [ $LD_LIBRARY_PATH ]
then
    if ! set | grep LD_LIBRARY_PATH | grep /usr/X11R6/lib:/usr/X11R6/lib/modules >
/dev/null
    then
        LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/X11R6/lib:/usr/X11R6/lib/modules
        export LD_LIBRARY_PATH
    fi
else
    LD_LIBRARY_PATH=/usr/X11R6/lib:/usr/X11R6/lib/modules
    export LD_LIBRARY_PATH
fi

```

Listing 8-1 (continued)

`/etc/profile` is a good place to set paths because it is where you set environmental variables that are passed to child processes in the shell. If you want to change the default path of your shells in `/etc/profile`, you can add another path statement in the path manipulation section of `/etc/profile`. For example, suppose that you create a directory called `/music` on your system and you want this directory to be in the system search path. You could add the following line to the end of the other similar lines:

```
pathmunge /music
```

Do not add too many paths to this section because users can set their own paths using a `.profile` in their home directories. Adding more default paths than are necessary can pose a security risk. For example, a user named `katie` may want to run her own version of `pine`, which she keeps in her home directory.

In that case, she may want to have `/home/$USER` or `/home/katie` at the beginning of her path so that when she types `pine`, the version in her home directory is found by the shell first, before finding the copy of `pine` in `/usr/bin/pine`. Generally, putting `/home/$USER` or any other directory whose contents are not controlled by root in `/etc/profile` is not a good idea.

The reason for this warning is that a rogue user or cracker can compile a backdoor, a way to enter the system unexpectedly, or corrupted version of a program and somehow get it in a user's home directory, perhaps even by mailing it to the user. If users' paths are set to check their home directories first, they may think that they are running a system program but instead are unknowingly running an alternate version.

On the other hand, if this path modification is set only in `katie's .profile`, only she runs this risk. She should also be aware of this risk since she has to perform the extra step of adding this path modification herself.

Another useful variable to change in the system profile is the number of user commands saved in the `.history` file in the user's directory. This command history is especially useful, since you can scroll through your previous commands by using the up and down arrows. To change the number of commands saved in the `.history` file, modify this line:

```
HISTSIZE=1000
```

bash, tcsh, zsh, and Their Config File Read Orders

The shells read a few configuration files when starting up. It is good to know which files are read in what order, so that you know where to set variables that will only apply to certain users.

- **bash** — `bash` reads the following files on startup: `/etc/profile`, all the files in `/etc/profile.d` `~/.bash_profile`, `~/.bash_login`, and `~/.profile`. Upon logout, `bash` reads `~/.bash_logout`.
- **tcsh** — `tcsh` reads the following files when starting up: `/etc/csh.cshrc`, then `/etc/csh.login`. After these come the config files in the user's home directory: `~/.tcshrc` (or if not present, `~/.cshrc`), `~/.history`, `~/.login`, `~/.cshdirs`.
- **zsh** — `zsh` reads the following when starting up: `/etc/zshenv`, `~/.zshenv`, `/etc/zprofile`, `~/.zprofile`, `/etc/zshrc`, `~/.zshrc`, and `/etc/zlogin`. Nonlogin shells also read `~/.bashrc`. Upon logout, `zsh` reads the `~/.zlogout` and `/etc/zlogout` files.

System Environmental Settings

The files discussed in this section deal with system environmental settings.

/etc/motd

This file contains the message that users see every time they log in. It's a good place to communicate messages about system downtime and other things that users should be aware of. On the other hand, you can put amusing quotes here to entertain your users. Usually, the `motd` contains a message like:

```
Welcome to Generic University's UNIX mail system.  
This system is monitored. Unauthorized use prohibited.  
System downtime scheduled this Sunday night from 10 pm to 1 am.
```

NOTE `motd` is a plain-text file, which you can edit with any text editor. You can use it to display any message you want users to see when they login. If you don't have this file in your `/etc` directory you can easily create it.

issue

Whatever is in this file shows up as a prelogin banner on your console. By default, this file tells which version of Red Hat is running on the system and the kernel version.

The default file looks like this:

```
Red Hat Linux release 7.2 (Enigma)  
Kernel \r on an \m
```

So when you log in, you see this message (or something similar, depending on the kernel running on your system):

```
Fedora Core release 3 (Heidelberg)  
Kernel 2.6.10-1.770_FC3 on an i686
```

issue.net

This file generally contains the same thing as `/etc/issue`. It shows up when you attempt to telnet into the system. Because it shows up to people who are connecting to your system over the Internet, you should change this message to include a warning such as "Access is being monitored. Unauthorized access

is prohibited.” Displaying this warning is good practice because if you want to prosecute intruders, it helps your case to show that you warned them that unauthorized access was prohibited.

aliases

`/etc/aliases` is the email aliases file for the Sendmail program, and Postfix uses `/etc/postfix/aliases`. By default, it contains many system account aliases. The aliases file sends mail for all the basic system accounts such as `bin`, `daemon`, and `operator` to `root`’s mailbox.

Other common email aliases, for example, send all of `root`’s mail to the user who commonly acts as `root`. So if `taleen` acts as `root` most of the time, she can alias `root`’s mailbox to her mailbox. This way, she doesn’t need to log in as `root` to read important system mail.

To do this, she’d put the following line in the aliases file:

```
root:          taleen
```

Or if she wants to send all `root` mail to her account on a remote machine, the line will read:

```
root:          taleen@buffy.xena.edu
```

Whenever you make changes to this file, you need to run the `newaliases` command to have the changes take affect in Sendmail.

fstab

`fstab` contains important information about your file systems, such as what file system type the partitions are, where they are located on the hard drive, and what mount point is used to access them.

This information is read by vital programs such as `mount`, `umount`, and `fsck`. `mount` runs at start time and mounts all the file systems mentioned in the `fstab` file, except for those with `noauto` in their line. If a partition you want to access is not listed in this file, you have to mount it manually. This can get tedious, so it’s better to list all of your file systems in `fstab`.

When `fsck` is run at bootup, it also checks all the file systems listed in `fstab` for consistency. It then fixes corrupted file systems, usually because they were not unmounted properly when the system crashed or suddenly lost power. File systems with an `fs_passno` value of 0 (the number in the last column) are not checked at boot time. As you can see in Listing 8-2, almost all file systems are checked at startup except for the floppy drive, which is not checked by `fsck` at bootup.

The `fstab` line has six fields, and each field represents a different configuration value. The first field describes the file system, which can be a partition name, the label of a disk partition, a logical volume, or a remote file system. The second field is the mount point used to access the file system. The third field describes the file system type. The fourth field is the place for any mount options you may need. The fifth field is 0 or 1 to determine whether `dump` backs up this file system. The final field sets the order in which `fsck` checks these file systems.

```
# This file is edited by fstab-sync - see 'man fstab-sync' for details
/dev/VolGroup00/LogVol100 / ext3 defaults 1 1
LABEL=/boot /boot ext3 defaults 1 2
none /dev/pts devpts gid=5,mode=620 0 0
none /dev/shm tmpfs defaults 0 0
none /proc proc defaults 0 0
none /sys sysfs defaults 0 0
/dev/VolGroup00/LogVol01 swap swap defaults 0 0
/dev/hdc /media/cdrecorder auto
pamconsole,exec,noauto,fscontext=system_u:object_r:removable_t,managed 0 0
```

Listing 8-2 A typical `fstab` file.

grub.conf

GRUB stands for the modest acronym *Grand Unified Bootloader*. It is the default boot loader used by Fedora Core and Red Hat Enterprise Linux. GRUB offers a nice graphical interface, giving you a basic choice between which installed operating systems or kernels you want to run. The `/etc/grub.conf` file is a symbolic link to the actual file that is located in `/boot/grub/grub.conf`.

Listing 8-3 shows a typical `grub.conf` file.

```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE: You have a /boot partition. This means that
#          all kernel and initrd paths are relative to /boot/, eg.
#          root (hd0,1)
#          kernel /vmlinuz-version ro root=/dev/VolGroup00/LogVol100
#          initrd /initrd-version.img
#boot=/dev/hda
default=0
timeout=5
splashimage=(hd0,1)/grub/splash.xpm.gz
hiddenmenu
password --md5 $1$ANJi7kLJ$/NODBfkCTkMAPxZgC8WK10
```

Listing 8-3 A typical GRUB configuration file. (*continued*)

```

title Fedora Core (2.6.10-1.770_FC3)
    root (hd0,1)
    kernel /vmlinuz-2.6.10-1.770_FC3 ro root=/dev/VolGroup00/LogVol100 rhgb
quiet
    initrd /initrd-2.6.10-1.770_FC3.img
title Fedora Core (2.6.10-1.766_FC3)
    root (hd0,1)
    kernel /vmlinuz-2.6.10-1.766_FC3 ro root=/dev/VolGroup00/LogVol100 rhgb
quiet
    initrd /initrd-2.6.10-1.766_FC3.img
title Fedora Core (2.6.9-1.724_FC3)
    root (hd0,1)
    kernel /vmlinuz-2.6.9-1.724_FC3 ro root=/dev/VolGroup00/LogVol100 rhgb
quiet
    initrd /initrd-2.6.9-1.724_FC3.img
#title Fedora Core (2.6.9-1.667)
#    root (hd0,1)
#    kernel /vmlinuz-2.6.9-1.667 ro root=/dev/VolGroup00/LogVol100 rhgb quiet
#    initrd /initrd-2.6.9-1.667.img
title Other
    rootnoverify (hd0,0)
    chainloader +1

```

Listing 8-3 *(continued)*

As you can see, the `default=0` line indicates that the first title section should be booted by default. GRUB starts its counting at 0 instead of 1. The title line contains the label that will be shown in the boot menu for that kernel. The root line specifies that Linux will be booted off the first hard drive. The kernel line indicates the kernel's location on the file system.

In the Other title section, notice that GRUB is calling a chain loader to be used for loading a different operating system; in this case it is actually Windows XP. GRUB uses a chain loader because it doesn't support loading Windows XP. GRUB uses a chain loader to load any operating system that it doesn't support.

CROSS-REFERENCE See Chapter 6 for a detailed explanation of GRUB.

cron files

cron is a daemon that executes commands according to a preset schedule that a user defines. It wakes up every minute and checks all cron files to see what jobs need to be run at that time. cron files can be set up by users or by the administrator to take care of system tasks. Basically, users edit their crontab files by telling cron what programs they'd like run automatically and how often they'd like to run them.

NOTE You should never manually edit the files in the `/var/spool/cron` directory.

User crontab files are stored in `/var/spool/cron/`. They are named after the user they belong to. System cron files are stored in the following subdirectories of the `/etc` directory:

- `cron.d`
- `cron.daily`
- `cron.hourly`
- `cron.monthly`
- `cron.weekly`

`crontab` in the `/etc` directory is sort of the master control file set up to run all the scripts in the `cron.daily` directory on a daily basis, all the scripts in the `cron.hourly` directory on an hourly bases, and so on with `cron.monthly` and `cron.weekly`.

`cron.d` is where system maintenance files that need to be run on a different schedule than the other `/etc` cron files are kept. By default, a file in `cron.d` called `sysstat` runs a system activity accounting tool every 10 minutes, 24×7 .

CROSS-REFERENCE Chapter 28 explains the `cron` command in more detail.

syslog.conf

The `syslog` daemon logs any notable events on your local system. It can store these logs in a local file or send them to a remote log host for added security. It can also accept logs from other machines when acting as a remote log host. These options and more, such as how detailed the logging should be, are set in the `syslog.conf` file.

Listing 8-4 is an excerpt that demonstrates the syntax and logic of the `syslog.conf` file. The first entry specifies that all messages that are severity-level info or higher should be logged in the `/var/log/messages` file.

Also indicated by the first entry is that any mail, news, private authentication, and cron messages should be logged elsewhere. Having separate log files makes it easier to search through logs if they are separated by type or program. The lines following this one specify the other places where those messages should be logged.

Authentication privilege messages contain somewhat sensitive information, so they are logged to `/var/log/secure`. That file can be read by root only, whereas `/var/log/messages` is sometimes set to be readable by everyone

or at least has less stringent access control. By default, `/var/log/messages` is set to be read by root only as well.

All mail messages are logged to `/var/log/maillog`, and cron messages are saved at `/var/log/cron`. uucp and critical-level news daemon log messages are saved to `/var/log/spooler`. All of these log files are set by default to be readable by root only. Emergency messages are sent to all the log files listed in the `syslog.conf` file, including to the console.

```
# Log all kernel messages to the console.
# Logging much else clutters up the screen.
#kern.*                                /dev/console

# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;authpriv.none;cron.none    /var/log/messages

# The authpriv file has restricted access.
authpriv.*                                /var/log/secure

# Log all the mail messages in one place.
mail.*                                    -/var/log/maillog

# Log cron stuff
cron.*                                    /var/log/cron

# Everybody gets emergency messages
*.emerg                                    *

# Save news errors of level crit and higher in a special file.
uucp,news.crit                            /var/log/spooler

# Save boot messages also to boot.log
local7.*                                /var/log/boot.log

# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;news.none;authpriv.none;cron.none    /var/log/messages

# The authpriv file has restricted access.
authpriv.*                                /var/log/secure

# Log all the mail messages in one place.
mail.*                                    /var/log/maillog

# Log cron stuff
cron.*                                    /var/log/cron
```

Listing 8-4 An excerpt from the `/etc/syslog.conf` file. (*continued*)

```
# Everybody gets emergency messages
*.emerg                                *

# Save news errors of level crit and higher in a special file.
uucp,news.crit                        /var/log/spooler
```

Listing 8-4 (continued)

ld.so.conf

This configuration file is used by `ldconfig`, which configures dynamic linker runtime bindings. It contains a listing of directories that hold shared libraries. Shared library files typically end with `.so`, whereas static library files typically end with `.a`, indicating they are an archive of objects.

You may need to edit this file if you've installed a program that has installed a shared library to a different library directory that is not listed in the `ld.so.conf` file. In this case, you get an error at runtime that the library does not exist.

An additional troubleshooting step to take in that case is to run `ldd` on the executable in question, which prints shared library dependencies. The output would look something like this:

```
[root@terry root]# ldd /bin/bash
libtermcap.so.2 => /lib/libtermcap.so.2 (0x40026000)
libdl.so.2 => /lib/libdl.so.2 (0x4002a000)
libc.so.6 => /lib/tls/libc.so.6 (0x42000000)
/lib/ld-linux.so.2 (0x40000000)
```

You can see a default listing of library directories in Listing 8-5.

```
include ld.so.conf.d/*.conf
/usr/X11R6/lib
/usr/lib/qt-3.3/lib
/usr/lib/mysql
```

Listing 8-5 A typical `ld.so.conf` file.

logrotate.conf

`logrotate.conf` and the files within the `logrotate.d` directory determine how often your log files are rotated by the `logrotate` program. Log rotation refers to the process of deleting older log files and replacing them with more recent ones. `logrotate` can automatically rotate, compress, remove,

and mail your log files. Log files can be rotated based on size or on time, such as daily, weekly, or monthly.

As you can see from the default `logrotate.conf` file shown in Listing 8-6, most of the options set for how and when to rotate the system logs are pretty self-explanatory.

For every program that has a separate log rotation configuration file in `logrotate.d`, and uses `syslogd` for logging, there should be a `logrot` config file for all log entries in `/etc/syslog.conf`, as well as log files produced by external applications, such as Apache. This is because `syslog` needs to save log entries for these programs in separate files so that their log files can be rotated independently of one another.

```
# see "man logrotate" for details
# rotate log files weekly
weekly

# keep 4 weeks worth of backlogs
rotate 4

# create new (empty) log files after rotating old ones
create

# uncomment this if you want your log files compressed
#compress

# RPM packages drop log rotation information into this directory
include /etc/logrotate.d

# no packages own wtmp -- we'll rotate them here
/var/log/wtmp {
    monthly
    create 0664 root wtmp
    rotate 1
}

# system-specific logs may be also be configured here.
```

Listing 8-6 The `logrotate.conf` file.

Examining the `/etc/sysconfig/` Directory

The following information outlines some of the files found in the `/etc/sysconfig/` directory, their functions, and their contents. This information is not intended to be complete, as many of these files have a variety of options used only in very specific or rare circumstances. The `/usr/share/doc/`
`/initscripts-version-number/sysconfig.txt` file contains a more

authoritative listing of the files found in the `/etc/sysconfig` directory and the configuration options available. Figure 8-1 shows the contents in a Fedora Core 3 `/etc/sysconfig` directory.

NOTE These files are used to pass configuration information to scripts that run when the system starts. It is possible that your system may be missing some of the configuration files described here, or it may have more of the files and directories, depending on whether the corresponding programs that need the files are installed or not. Also, if the service that uses the configuration file is not started, the configuration file will not be read.

`/etc/sysconfig/apmd`

The `/etc/sysconfig/apmd` file is used by `apmd` as a configuration for what things to start, stop, change on suspend, or resume. It provides information to the `apmd` during startup if `apmd` is set to start, depending on whether your hardware supports Advanced Power Management (APM) or whether you choose to use it. APM is a monitoring daemon that works with power management code within the Linux kernel. It can alert you to a low battery if you are using Red Hat Linux on a laptop, among other things.

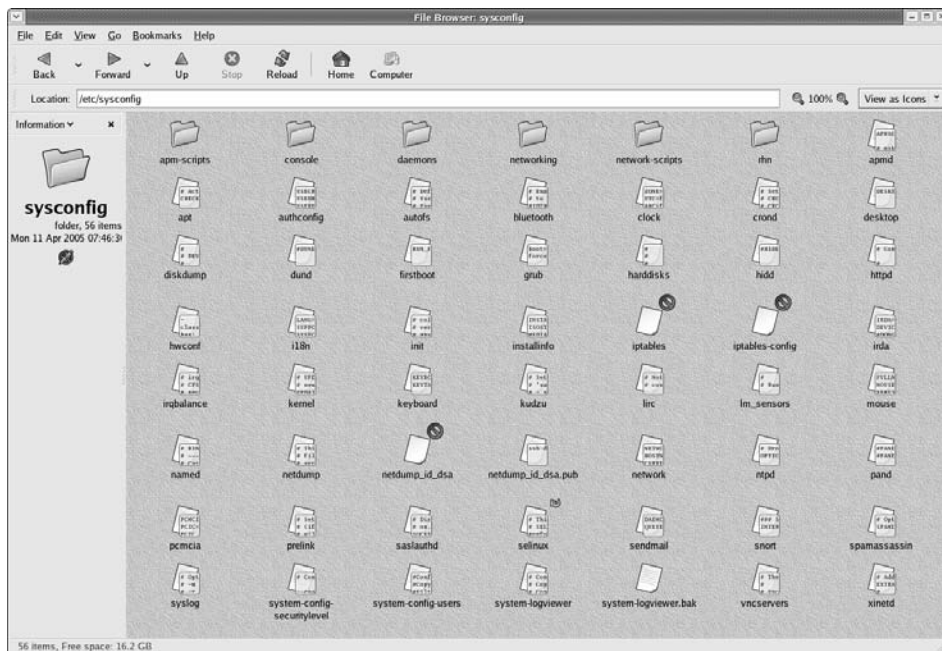


Figure 8-1 The contents of the `/etc/sysconfig` directory.

/etc/sysconfig/authconfig

The `/etc/sysconfig/authconfig` file provides settings to `/usr/sbin/authconfig`, which is called from `/etc/rc.sysinit` for the kind of authentication to be used on the host. The basic syntax for lines in this file is:

```
USE <service name> =<value>
```

Some sample lines from the file are shown here.

- **USEMD5=value**, where `value` is one of the following:
 - `yes` — MD5 is used for authentication.
 - `no` — MD5 is not used for authentication.
- **USEKERBEROS=value**, where `value` is one of the following:
 - `yes` — Kerberos is used for authentication.
 - `no` — Kerberos is not used for authentication.
- **USELDAPAUTH=value**, where `value` is one of the following:
 - `yes` — LDAP is used for authentication.
 - `no` — LDAP is not used for authentication.

/etc/sysconfig/clock

The `/etc/sysconfig/clock` file controls the interpretation of values read from the system clock. Currently, the correct values are as follows:

- **UTC=value**, where `value` is one of the following Boolean values:
 - `true` — Indicates that the hardware clock is set to Universal Time.
 - Any other value indicates that it is set to local time.
- **ARC=value**, where `value` is the following:
 - `true` — Indicates the ARC console's 42-year time offset is in effect.
 - Any other value indicates that the normal UNIX epoch is assumed (for Alpha-based systems only).
- **ZONE=filename** — Indicates the time zone file under `/usr/share/zoneinfo` that `/etc/localtime` is a copy of, such as: `ZONE="America/New York"`. Identifies the time zone file copied into `/etc/localtime`, such as `ZONE="America/New York"`. Time zone files are stored in `/usr/share/zoneinfo`.

/etc/sysconfig/cron

This file contains settings for the cron daemon. You typically do not need to make any changes to this file.

/etc/sysconfig/desktop

The `/etc/sysconfig/desktop` file specifies the desktop manager to be run and is used by the `/etc/X11/xinit/Xclients` script, for example:

```
DESKTOP="GNOME"
```

/etc/sysconfig/firstboot

In Fedora Core and Enterprise Linux, the first time you boot the system after the OS installation, the `/sbin/init` program calls the `etc/rc.d/init.d/firstboot` script. This allows the user to install additional applications and documentation before the boot process completes. The `/etc/sysconfig/firstboot` file tells the `firstboot` command not to run on subsequent reboots. If you want `firstboot` to run the next time you boot the system, simply remove `/etc/sysconfig/firstboot` and execute `chkconfig --level 5 firstboot on`.

/etc/sysconfig/grub

The `/etc/sysconfig/grub` file is used to pass arguments to GRUB at boot time. The information passed is the drive to boot from and whether to use `lba` mode.

/etc/sysconfig/harddisks

The `/etc/sysconfig/harddisks` file allows you to tune your hard drive(s).

CAUTION Do not make changes to this file lightly. If you change the default values stored here, you could corrupt all of the data on your hard drive(s).

The `/etc/sysconfig/harddisks` file may contain the following:

- **USE_DMA=1**, where setting this to 1 enables DMA. However, with some chipsets and hard-drive combinations, DMA can cause data corruption. Check with your hard-drive documentation or manufacturer before enabling this.

- **Multiple_IO=16**, where a setting of 16 allows for multiple sectors per I/O interrupt. When enabled, this feature reduces operating system overhead by 30 to 50 percent. Use with caution.
- **EIDE_32BIT=3** enables (E)IDE 32-bit I/O support to an interface card.
- **LOOKAHEAD=1** enables drive read-lookahead.
- **EXTRA_PARAMS=** specifies where extra parameters can be added.

/etc/sysconfig/hwconf

The `/etc/sysconfig/hwconf` file lists all the hardware that kudzu detected on your system, as well as the drivers used, vendor ID, and device ID information. The kudzu program detects and configures new and/or changed hardware on a system. The `/etc/sysconfig/hwconf` file is not meant to be manually edited. If you do edit it, devices can suddenly show up as added or not show up if removed.

/etc/sysconfig/i18n

The `/etc/sysconfig/i18n` file sets the default language, for example:

```
LANG="en_US"
```

/etc/sysconfig/init

The `/etc/sysconfig/init` file controls how the system will appear and function during the boot process. The following values may be used:

- **BOOTUP=value**, where value is one of the following:
 - **BOOTUP=color** means the standard color boot display, where the success or failure of devices and services starting up is shown in different colors.
 - **BOOTUP=verbose** means an old-style display, which provides more information than purely a message of success or failure.
 - Anything else means a new display, but without ANSI formatting.
- **RES_COL=value**, where value is the number of the column of the screen to start status labels. It defaults to 60.
- **MOVE_TO_COL=value**, where value moves the cursor to the value in the `RES_COL` line. It defaults to ANSI sequences output by `echo -e`.

- **SETCOLOR_SUCCESS=***value*, where *value* sets the color to a color indicating success. It defaults to ANSI sequences output by `echo -e`, setting the color to green.
- **SETCOLOR_FAILURE=***value*, where *value* sets the color to one indicating failure. It defaults to ANSI sequences output by `echo -e`, setting the color to red.
- **SETCOLOR_WARNING=***value*, where *value* sets the color to one indicating warning. It defaults to ANSI sequences output by `echo -e`, setting the color to yellow.
- **SETCOLOR_NORMAL=***value*, where *value* sets the color to “normal.” It defaults to ANSI sequences output by `echo -e`.
- **LOGLEVEL=***value*, where *value* sets the initial console logging level for the kernel. The default is 7; 8 means everything (including debugging); 1 means nothing except kernel panics. `syslogd` will override this once it starts.
- **PROMPT=***value*, where *value* is one of the following Boolean values:
 - **yes** — Enables the key check for interactive mode.
 - **no** — Disables the key check for interactive mode.

/etc/sysconfig/iptables

The `/etc/sysconfig/iptables` file stores information used by the kernel to set up packet-filtering services at boot time or whenever the service is started. You should not modify this file by hand unless you are familiar with how to construct iptables rules. The simplest way to add rules is to use the `/usr/sbin/lokit` command from a terminal prompt if you aren't running an X server. If you are running an X server, you can type **system-config-securitylevel** from a terminal prompt or select Applications ⇨ System Settings ⇨ Security Level from the main menu to start the graphical application to create your firewall. Using these applications automatically edits this file at the end of the process.

If you wish, you can manually create rules using `/sbin/iptables` and then type **/sbin/service iptables save** to add the rules to the `/etc/sysconfig/iptables` file. Once this file exists, any firewall rules saved there are persisted through a system reboot or a service restart.

CROSS-REFERENCE For more information on iptables, see Chapter 34.

/etc/sysconfig/irda

The `/etc/sysconfig/irda` file controls how infrared devices on your system are configured at startup. The following values may be used:

- **IRDA=value**, where `value` is one of the following Boolean values:
 - `yes` — `irattach` will be run, which periodically checks to see whether anything is trying to connect to the infrared port, such as another notebook computer attempting to make a network connection. For infrared devices to work on your system, this line must be set to `yes`.
 - `no` — `irattach` will not be run, preventing infrared device communication.
- **DEVICE=value**, where `value` is the device (usually a serial port) that handles infrared connections.
- **DONGLE=value**, where `value` specifies the type of dongle being used for infrared communication. This setting exists for people who use serial dongles rather than real infrared ports. A dongle is a device attached to a traditional serial port to communicate via infrared. This line is commented out by default because notebooks with real infrared ports are far more common than computers with add-on dongles.
- **DISCOVERY=value**, where `value` is one of the following Boolean values:
 - `yes` — Starts `irattach` in discovery mode, meaning it actively checks for other infrared devices. This needs to be turned on in order for the machine to be actively looking for an infrared connection (meaning the peer that does not initiate the connection).
 - `no` — Does not start `irattach` in discovery mode.

/etc/sysconfig/kernel

The settings in this file specify whether new kernels loaded by the `up2date` utility should be booted by default. You can change the setting from `yes` to `no` to prevent the newly updated kernel from booting.

/etc/sysconfig/keyboard

The `/etc/sysconfig/keyboard` file controls the behavior of the keyboard. The following values may be used:

- **KEYBOARDTYPE=sun|pc**, which is used on SPARCs only. `sun` means a Sun keyboard is attached on `/dev/kbd`, and `pc` means a PS/2 keyboard is connected to a PS/2 port.

- **KEYTABLE=file**, where `file` is the name of a keytable file. For example: `KEYTABLE="us"`. The files that can be used as keytables start in `/usr/lib/kbd/keymaps/i386` and branch into different keyboard layouts from there, all labeled `file.kmap.gz`. The first file found beneath `/usr/lib/kbd/keymaps/i386` that matches the `KEYTABLE` setting is used.

/etc/sysconfig/kudzu

The `/etc/sysconfig/kudzu` is used by `/etc/init.d/kudzu`, and it allows you to specify a safe probe of your system's hardware by `kudzu` at boot time. A safe probe is one that disables serial port probing.

- **SAFE=value**, where `value` is one of the following:
 - `yes` — `kudzu` does a safe probe.
 - `no` — `kudzu` does a normal probe.

/etc/sysconfig/mouse

The `/etc/sysconfig/mouse` file is used by `/etc/init.d/gpm` to specify information about the available mouse. The following values may be used:

- **FULLNAME=value**, where `value` refers to the full name of the kind of mouse being used.
- **MOUSETYPE=value**, where `value` is one of the following:
 - `microsoft` — A Microsoft mouse.
 - `mouseman` — A MouseMan mouse.
 - `mousesystems` — A Mouse Systems mouse.
 - `ps/2` — A PS/2 mouse.
 - `msbm` — A Microsoft bus mouse.
 - `logibm` — A Logitech bus mouse.
 - `atibm` — An ATI bus mouse.
 - `logitech` — A Logitech mouse.
 - `mmseries` — An older MouseMan mouse.
 - `mmhittab` — An mmhittab mouse.

- **XEMU3=value**, where *value* is one of the following Boolean values:
 - **yes** — The mouse has only two buttons, but three mouse buttons should be emulated.
 - **no** — The mouse already has three buttons.
- **XMOUSETYPE=value**, where *value* refers to the kind of mouse used when X is running. The options here are the same as those provided by the **MOUSETYPE** setting in this same file.
- **DEVICE=value**, where *value* is the mouse device. In addition, `/dev/mouse` is a symbolic link that points to the actual mouse device.

/etc/sysconfig/named

The `/etc/sysconfig/named` file is used to pass arguments to the `named` daemon at boot time if the `named` daemon is started. The `named` daemon is a Domain Name System (DNS) server, which implements the Berkeley Internet Name Domain (BIND) version 9 distribution. This server maintains a table of which hostnames are associated with IP addresses on the network. Currently, only the following values may be used:

- **ROOTDIR=/some/where**, where */some/where* refers to the full directory path of a configured `chroot` environment under which `named` will run. This `chroot` environment must first be configured. Type **info chroot** for more information on how to do this.
- **OPTIONS="value"**, where *value* is any option listed in the man page for `named` except `-t`. In place of `-t`, use the preceding **ROOTDIR** line.

For more information about what parameters you can use in this file, type **man named**. By default, the file contains no parameters.

CROSS-REFERENCE For detailed information on how to configure a BIND DNS server, see Chapter 18.

/etc/sysconfig/netdump

The `/etc/sysconfig/netdump` file is the configuration file for the `/etc/init.d/netdump` service. The `netdump` service sends both oops data and memory dumps over the network. In general, `netdump` is not a required service, so you should run it only if you absolutely need to. For more information about what parameters you can use in this file, type **man netdump**.

/etc/sysconfig/network

The `/etc/sysconfig/network` file is used to specify information about the desired network configuration. The following values may be used:

- **NETWORKING=value**, where *value* is one of the following Boolean values:
 - **yes** — Networking should be configured.
 - **no** — Networking should not be configured.
- **HOSTNAME=value**, where *value* should be the fully qualified domain name (FQDN), such as `hostname.domain.com`, but can be whatever hostname you want.

NOTE For compatibility with older software that people might install, the `/etc/HOSTNAME` file and the `/etc/sysconfig/network` file should contain the same value.

- **GATEWAY=value**, where *value* is the IP address of the network's gateway.
- **GATEWAYDEV=value**, where *value* is the gateway device, such as `eth0`.
- **NISDOMAIN=value**, where *value* is the NIS domain name.

/etc/sysconfig/ntpd

The `/etc/sysconfig/ntpd` file is used to pass arguments to the `ntpd` daemon if it is used at boot time. The `ntpd` daemon sets and maintains the system clock to synchronize with an Internet standard time server. It implements version 4 of the Network Time Protocol (NTP). For more information about what parameters you can use in this file, point a browser at the following file: `/usr/share/doc/ntp-version/ntpd.htm` (where *version* is the version number of `ntpd`). By default, this file sets the owner of the `ntpd` process to the user `ntp`.

/etc/sysconfig/pcmcia

The `/etc/sysconfig/pcmcia` file is used to specify PCMCIA configuration information. The following values may be used:

- **PCMCIA=value**, where *value* is one of the following:
 - **yes** — PCMCIA support should be enabled.
 - **no** — PCMCIA support should not be enabled.

- **PCIC=value**, where value is one of the following:
 - **i82365** — The computer has an i82365-style PCMCIA socket chipset.
 - **tcic** — The computer has a tcic-style PCMCIA socket chipset.
- **PCIC_OPTS=value**, where value is the socket driver (i82365 or tcic) timing parameters.
- **CORE_OPTS=value**, where value is the list of pcmcia_core options.
- **CARDMGR_OPTS=value**, where value is the list of options for the PCMCIA cardmgr (such as **-q** for quiet mode, **-m** to look for loadable kernel modules in the specified directory, and so on). Read the cardmgr man page for more information.

/etc/sysconfig/selinux

This file is a link to `/etc/selinux/config` and is used to control selinux on the system. It contains two settings that control the state of selinux — enforcing, permissive, or disabled — and the type of policy, either targeted or strict. A sample of this file is shown here.

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - SELinux is fully disabled.
SELINUX=permissive
# SELINUXTYPE= type of policy in use. Possible values are:
#     targeted - Only targeted network daemons are protected.
#     strict - Full SELinux protection.
SELINUXTYPE=targeted
```

/etc/sysconfig/system-config-users

The `/etc/sysconfig/system-config-users` file is the configuration file for the graphical application User Manager. This file is used to filter out system users such as `root`, `daemon`, and `lp`. This file is edited via the Preferences ⇨ Filter system users and groups pull-down menu in the User Manager application and should not be edited manually.

/etc/sysconfig/system-logviewer

The `/etc/sysconfig/system-logviewer` file is the configuration file for the graphical, interactive log-viewing application Log Viewer. This file is edited

via the Edit ⇨ Preferences pull-down menu in the System Logs application and should not be edited manually.

/etc/sysconfig/samba

The `/etc/sysconfig/samba` file is used to pass arguments to the `smbd` and the `nmbd` daemons at boot time. The `smbd` daemon offers file-sharing connectivity for Windows clients on the network. The `nmbd` daemon offers NetBIOS-over-IP naming services. For more information about what parameters you can use in this file, type **man `smbd`**. By default, this file sets `smbd` and `nmbd` to run in daemon mode.

/etc/sysconfig/sendmail

The `/etc/sysconfig/sendmail` file allows messages to be sent to one or more recipients, routing the message over whatever networks are necessary. The file sets the default values for the Sendmail application to run. Its default values are to run as a background daemon, and to check its queue once an hour in case something has backed up and stalled the process. The following values may be used:

- **DAEMON=***value*, where *value* is one of the following Boolean values:
 - **yes** — Sendmail should be configured to listen to port 25 for incoming mail. **yes** implies the use of Sendmail's `-bd` options.
 - **no** — Sendmail should not be configured to listen to port 25 for incoming mail.
- **QUEUE=***1h*, which is given to Sendmail as `-q$QUEUE`. The `-q` option is not given to Sendmail if `/etc/sysconfig/sendmail` exists and `QUEUE` is empty or undefined.

/etc/sysconfig/vncservers

The `/etc/sysconfig/vncservers` file configures the way the virtual network computing (VNC) server starts up. VNC is a remote display system that allows you to view a desktop environment not only on the machine where it is running but across different networks on a variety of architectures. It may contain the following:

- **VNCSERVERS=***value*, where *value* is set to something like `1:fred` to indicate that a VNC server should be started for user `fred` on display `:1`. User `fred` must have set a VNC password using `vncpasswd` before attempting to connect to the remote VNC server.

Note that when you use a VNC server, your communication with it is unencrypted, and so it should not be used on an untrusted network. For specific instructions concerning the use of SSH to secure the VNC communication, see research.att.com/vnc/sshvnc.html. To find out more about SSH, see Chapter 28.

/etc/sysconfig/xinetd

The `/etc/sysconfig/xinetd` file is used to pass arguments to the `xinetd` daemon at boot time. The `xinetd` daemon starts programs that provide Internet services when a request to the port for that service is received. For more information about what parameters you can use in this file, type **man xinetd**. For more information on the `xinetd` service, see Chapter 25.

Directories in the /etc/sysconfig/ Directory

The following directories are normally found in `/etc/sysconfig/`.

apm-scripts

This contains the Red Hat APM `suspend/resume` script. You should not edit this file directly. If you need customization, simply create a file called `/etc/sysconfig/apm-scripts/apmcontinue`, and it will be called at the end of the script. Also, you can control the script by editing `/etc/sysconfig/apmd`.

daemons

This directory is initially empty after the system installation. It is used to hold the configuration scripts for programs that the user may have installed. For example, the configuration files for the `webmin` program are placed in this directory during its installation.

networking

This directory is used by the Network Configuration tool (`system-config-network`), and its contents should not be edited manually.

CROSS-REFERENCE For more information about configuring network interfaces using the Network Configuration tool, see Chapter 11.

network-scripts

This directory contains files used for network configuration.

- Network configuration files for each configured network interface, such as `ifcfg-eth0` for the `eth0` Ethernet interface.
- Scripts used to bring up and down network interfaces, such as `ifup` and `ifdown`.
- Scripts used to bring up and down ISDN interfaces, such as `ifup-isdn` and `ifdown-isdn`.
- Various shared network function scripts that should not be edited directly.

rhn

This directory contains the configuration files and GPG keys for Red Hat Network. No files in this directory should be edited by hand. For more information on Red Hat Network, see the Red Hat Network Web site at <https://rhn.redhat.com>.

Examining the Network Configuration Files

This section discusses the following topics:

- Files to change when setting up a system or moving the system
- Starting up network services from `xinetd`
- Starting up network services from the `rc` scripts
- Other important network configuration files in the `/etc/sysconfig` directory

Files to Change When Setting Up a System or Moving the System

Whenever you set up a system to work on a new network, either because you've just installed Red Hat or you're moving the machine from one location to another, a set of files needs to be modified to get it working on the new network.

You need to:

- Set up the IP addresses of your network interfaces. Make changes to:

`/etc/sysconfig/network-scripts/ifcfg-eth0`

- Set up the hostname of your machine. Make changes to:

`/etc/sysconfig/network`

`/etc/hosts`

- Set up the DNS servers to reference. Make changes to:

`/etc/resolv.conf`

- Make a local file of hostname to IP address mappings. Make changes to:

`/etc/hosts`

- Set up the device order from which hostnames are looked up. Make changes to:

`/etc/nsswitch.conf`

CROSS-REFERENCE Chapter 20 explains the Domain Name System (DNS) and how to set it up on your network.

TIP Fedora Core and Red Hat Enterprise Linux provide a handy graphical tool, called the **Network Configuration tool** for configuring your network settings. Start up the Network Configuration tool while in X-Window, and enjoy an interface very similar to the Windows control panel for networks. Refer to Chapter 11 for instructions on using the Network Configuration tool. If you use the Network Configuration tool to set up your network, you do not need to edit the files manually as explained in the next sections. Also, if you use DHCP to obtain your IP information, you do not need to do any manual configuration and can skip the sections related to manually configuring your network settings.

Setting Up the IP Address

The first thing you should do is set an IP address on your network interfaces. This step provides your computer with an identity on the network. If you haven't set the IP address already in the installation process, you need to edit the configuration files by hand.

To set the IP address on your first Ethernet interface `eth0`, edit the `/etc/sysconfig/network-scripts/ifcfg-eth0` file. A copy of this file is shown in Listing 8-7.

Insert your interface's IP address on the line that says:

```
IPADDR=" "
```

You should also check that the rest of the lines look all right, but pay special attention to the following two lines:

```
BROADCAST=192.168.1.255
NETMASK="255.255.255.0"

DEVICE="eth0"
BOOTPROTO="static"
BROADCAST=192.168.1.255
IPADDR="192.168.1.10"
NETMASK="255.255.255.0"
NETWORK=192.168.1.0
ONBOOT="yes"
USERCTL=no
```

Listing 8-7 The `/etc/sysconfig/network-scripts/ifcfg-eth0` file.

Setting Up the Hostname

Once you've picked your hostname, you need to put it into two different places: `/etc/sysconfig/network` and `/etc/hosts`.

In `/etc/sysconfig/network`, shown next, change the line that says:

```
HOSTNAME="terry"
```

This is the `/etc/sysconfig/network` file:

```
NETWORKING=yes
HOSTNAME="terry"
GATEWAY="192.168.1.1"
GATEWAYDEV="eth0"
FORWARD_IPV4="yes"
```

You also need to modify the `/etc/hosts` file. Change the first line in the file, which would look something like this by adding the hostname you want:

```
127.0.0.1      terry      localhost.localdomain localhost locala localb localc
```

Setting Up the DNS Name Resolution

Now you should be able to communicate with the other hosts on the network. However, you won't be able to talk to them unless you know their IP addresses, because you haven't set up what DNS servers you should reference to map hostnames to IP addresses.

The program that resolves hostnames to IP addresses reads a file called `resolv.conf`, so you need to put your DNS server IP addresses there. Generally, you need one name server, but you can include up to three, if you'd like. Specifying more than one name server is important. If the first one on the list is not responding, your computer tries to resolve against the next one on the list, and so on, until it finds one that is responding.

Edit `/etc/resolv.conf` to contain a list of name servers, like this:

```
nameserver 1.2.3.4
nameserver 1.2.3.5
nameserver 1.2.3.6
```

Making a Local File of Hostname to IP Address Mappings

Linux gives you the ability to store a list of hostnames and their corresponding IP addresses in `/etc/hosts`, so that you don't have to look them up in DNS every time you use them. While you shouldn't do this with every hostname you ever use, one of the advantages gained by configuring often-used hostnames in this way includes the ability to alias a fully qualified hostname to a shorter version of itself. So instead of typing `ssh foo.xena.edu` every time you want to SSH to that machine, you can just type `ssh foo`, and have it connect to the same host.

Another useful example occurs if you're monitoring several servers' network services from a monitoring host. If you're monitoring SSH connectivity to certain servers, for example, and your DNS server stops responding, then the monitoring software may report that all your hosts are down. This happens because the monitoring software tries to connect to the server via its hostname, and gets no response because DNS is not providing it with an IP address to connect to. In this case it looks as if your whole network fell over, when the real problem is that your DNS service is not responding properly.

To keep this kind of scenario from happening, you should put the hostnames and IP addresses of all your monitored servers in `/etc/hosts`. This way, your monitoring software looks into `/etc/hosts` to get the proper IP addresses, instead of relying on DNS.

The only caveat to keep in mind when putting hosts in `/etc/hosts` is that if the hostname's IP address changes for whatever reason, the hosts file does not automatically update to reflect that change. If you start getting connection errors when connecting to a host in the `/etc/hosts` file, you should do an `nslookup` on the host and update your `/etc/hosts` file accordingly.

Your `/etc/hosts` file should contain IP address to hostname mappings that follow this format

```
IP_address canonical_hostname aliases
```


so that the lines look like this:

```
192.168.1.66    foo.xena.edu    foo
192.168.1.76    buffy.xena.edu   buffy
152.2.210.81    sunsite.unc.edu  sunsite
```

Setting Up Name Service Resolution Order

Now that you've set up your DNS servers and hosts file, you need to tell your Linux server which method it should use first to look up hostnames.

The place to set up this configuration is in the `/etc/nsswitch.conf` file. Edit the following line:

```
hosts:          files nisplus dns
```

The order of the words `files`, `nisplus`, and `dns` determines which method is checked first. `Files` refers to the `/etc/hosts` file, `nisplus` refers to any `nisplus` servers you may have on your network, and `dns` refers to any DNS servers you have set up your machine to reference.

As you can see in Listing 8-8, the `/etc/nsswitch.conf` file contains some other useful settings; for example the following two lines, specify whether the server should authenticate users off the local password file or off the network's NIS plus service:

```
passwd:         files nisplus
shadow:         files nisplus

#
# /etc/nsswitch.conf
#
# An example Name Service Switch config file. This file should be
# sorted with the most-used services at the beginning.
#
# The entry '[NOTFOUND=return]' means that the search for an
# entry should stop if the search in the previous entry turned
# up nothing. Note that if the search failed due to some other reason
# (like no NIS server responding) then the search continues with the
# next entry.
#
# Legal entries are:
#
#      nisplus or nis+      Use NIS+ (NIS version 3)
#      nis or yp            Use NIS (NIS version 2), also called YP
#      dns                  Use DNS (Domain Name Service)
#      files                 Use the local files
```

Listing 8-8 The `/etc/nsswitch.conf` file. (*continued*)

```

#      db                  Use the local database (.db) files
#      compat              Use NIS on compat mode
#      hesiod              Use Hesiod for user lookups
#      [NOTFOUND=return]   Stop searching if not found so far
#

# To use db, put the "db" in front of "files" for entries you want to be
# looked up first in the databases
#
# Example:
#passwd:    db files nisplus nis
#shadow:    db files nisplus nis
#group:     db files nisplus nis

passwd:     files nisplus
shadow:     files nisplus
group:      files nisplus

#hosts:     db files nisplus nis dns
hosts:      files nisplus dns

# Example - obey only what nisplus tells us...
#services:  nisplus [NOTFOUND=return] files
#networks:  nisplus [NOTFOUND=return] files
#protocols: nisplus [NOTFOUND=return] files
#rpc:       nisplus [NOTFOUND=return] files
#ethers:    nisplus [NOTFOUND=return] files
#netmasks: nisplus [NOTFOUND=return] files

bootparams: nisplus [NOTFOUND=return] files

ethers:     files
netmasks:   files
networks:   files
protocols:  files nisplus
rpc:        files
services:   files nisplus

netgroup:   files nisplus

publickey:  nisplus

automount:  files nisplus
aliases:    files nisplus

```

Listing 8-8 (continued)

Starting Up Network Services from xinetd

xinetd is the replacement for inetd. xinetd is started on bootup and listens on ports designated in the `/etc/xinetd.conf` for incoming network

connections. When a new connection is made, xinetd starts up the corresponding network service.

You should disable any unnecessary services from being started from xinetd as part of securing your machine. The way to do this is to edit that service's configuration file. xinetd's main configuration file is `/etc/xinetd.conf`. At the end of the `xinetd.conf` file is a line that indicates that all the files in the `/etc/xinetd.d` are also included in the configuration. This means that you need to go through the files in that directory as well to turn off any services you don't want.

So, to disable Telnet, you would look in `/etc/xinetd.d` for a file called `telnet`. The `telnet` file is shown in Listing 8-9. Edit the line in the config file that says `disable = no`, and change that to **`disable = yes`**, as it appears in Listing 8-9. After that line is set to `disable = yes`, the service is disabled and does not start up the next time you boot up.

```
/etc/xinetd.d/telnet

# default: on
# description: The telnet server serves telnet sessions; it uses \
#      unencrypted username/password pairs for authentication.
service telnet
{
    flags             = REUSE
    socket_type       = stream
    wait              = no
    user              = root
    server             = /usr/sbin/in.telnetd
    log_on_failure    += USERID
    disable           = yes
}
```

Listing 8-9 The Telnet config file in the `xinetd.d` directory.

An automated tool, called `chkconfig`, manages what services are started from xinetd and the `rc` scripts. You can read more about `chkconfig` in the section called “Managing `rc` Scripts Using `chkconfig`.”

Starting Up Network Services from the `rc` Scripts

Network services that are not started out of xinetd are started out of the `rc` scripts at boot time. Network services started at the default boot level 3 (multi-user networked mode) are started out of the `/etc/rc3.d` directory. If you look in that directory, you should see a file with the name of the service you want to stop or start. The script to start the service starts with an `S`, and the kill script starts with a `K`.

So for example, SSH is started from `/etc/rc3.d/S55sshd`, and killed upon shutdown from `/etc/rc6.d/K25sshd`. Runlevel 6 is the shutdown level, so that's why its kill script is located in the `rc6.d` directory.

Detailed information on managing all the services started at boot time can be found in the section “Managing the init Scripts” later in this chapter.

CROSS-REFERENCE See Chapter 6 for details about the system startup process.

Other Important Network Configuration Files in the /etc/sysconfig Directory

You can use the files listed in this section to create routes to other hosts, either on your own network or on outside networks. You also can use these files to set up firewall rules for your network to either allow or disallow connections to your network.

static-routes

If you want to set up some static routes on your machine, you can do so in the `static-routes` file. This config file has lines in the following format:

```
network-interface net network netmask netmask gw gateway
```

Iptables

`iptables` is the current Fedora Core and Enterprise Linux firewall. It supercedes the `ipchains` firewall. It can use `ipchains` rules as a component of its firewall filtering, but `iptables` and `ipchains` cannot be run at the same time. This is the file where the `iptables` rules are stored.

When you install Fedora or Enterprise Linux, the installation asks if you would like to enable a host-based firewall. If you select to enable a host-based firewall, a default set of `iptables` rules installs according to your preferences.

The following is a simplified configuration file. The gist of this configuration is that all incoming traffic to privileged ports (those below 1024) is dropped except for SSH traffic. The first line accepts all traffic from the loop-back interface. The second line accepts all incoming TCP traffic to the SSH port. The third line drops all incoming TCP traffic to ports between 1 and 1024. The last line drops all incoming UDP traffic to ports between 1 and 1024.

```
-A INPUT -i lo -j ACCEPT
-A INPUT -p tcp --dport 22 -j ACCEPT
-A INPUT -p tcp -s ! 192.168.1.0/24 --dport 1:1024 -j DROP
-A INPUT -p udp -s ! 192.168.1.0/24 --dport 1:1024 -j DROP
```

Network Configuration Files in /etc/sysconfig/network-scripts

You can use the files in this directory to set the parameters for the hardware and software used for networking. The scripts contained here are used to enable network interfaces and set other network-related parameters.

ifcfg-networkinterfacename

A few files fall into this specification. Red Hat specifies a separate configuration file for each network interface. In a typical Red Hat install, you might have many different network interface config files that all follow the same basic syntax and format.

You could have `ifcfg-eth0` for your first Ethernet interface, `ifcfg-irlan0` for your infrared network port, `ifcfg-lo` for the network loopback interface, and `ifcfg-ppp0` for your PPP network interface.

ifup and ifdown

These files are symlinks to `/sbin/ifup` and `/sbin/ifdown`. In future releases, these symlinks might be phased out. But for now, these scripts are called when the network service is started or stopped. In turn, `ifup` and `ifdown` call any other necessary scripts from within the `network-scripts` directory. These should be the only scripts you call from this directory.

You call these scripts with the name of the interface that you want to bring up or down. If these scripts are called at boot time, then `boot` is used as the second argument. For instance, to bring your Ethernet interface down and then up again after boot, you would type:

```
ifup eth0
ifdown eth0
```

Managing the init Scripts

This section discusses the following topics:

- Managing rc scripts by hand
- Managing rc scripts using `chkconfig`

Init scripts determine which programs start up at boot time. Red Hat and other Unix distributions have different runlevels, so there are a different set of programs that are started at each runlevel.

Usually Red Hat Linux starts up in multiuser mode with networking turned on. These are some of the other runlevels available:

- 0 — Halt
- 1 — Single-user mode
- 2 — Multiuser mode, without networking
- 3 — Full multiuser mode
- 4 — Not used
- 5 — Full multiuser mode (with an X-based login screen)
- 6 — Reboot

The system boots into the default runlevel set in `/etc/inittab`. (See Listing 8-10.)

NOTE In Red Hat Linux, the default boot level is 3 for a non-GUI system. When booting into an X-windows login, the default boot level is 5. You can get more information about this topic in Chapter 6.

```
#
# inittab          This file describes how the INIT process should set up
#                  the system in a certain run-level.
#
# Author:          Miquel van Smoorenburg, <miquels@drinkel.nl.mugnet.org>
#                  Modified for RHS Linux by Marc Ewing and Donnie Barnes
#

# Default runlevel. The runlevels used by RHS are:
#  0 - halt (Do NOT set initdefault to this)
#  1 - Single user mode
#  2 - Multiuser, without NFS (The same as 3, if you do not have networking)
#  3 - Full multiuser mode
#  4 - unused
#  5 - X11
#  6 - reboot (Do NOT set initdefault to this)
#
id:5:initdefault:

# System initialization.
si::sysinit:/etc/rc.d/rc.sysinit

10:0:wait:/etc/rc.d/rc 0
11:1:wait:/etc/rc.d/rc 1
12:2:wait:/etc/rc.d/rc 2
13:3:wait:/etc/rc.d/rc 3
14:4:wait:/etc/rc.d/rc 4
```

Listing 8-10 A default `/etc/inittab` file. (continued)

```

15:5:wait:/etc/rc.d/rc 5
16:6:wait:/etc/rc.d/rc 6

# Things to run in every runlevel.
ud::once:/sbin/update

# Trap CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now

# When our UPS tells us power has failed, assume we have a few minutes
# of power left. Schedule a shutdown for 2 minutes from now.
# This does, of course, assume you have powerd installed and your
# UPS connected and working correctly.
pf::powerfail:/sbin/shutdown -f -h +2 "Power Failure; System Shutting Down"

# If power was restored before the shutdown kicked in, cancel it.
pr:12345:powerokwait:/sbin/shutdown -c "Power Restored; Shutdown Cancelled"

# Run gettys in standard runlevels
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6

# Run xdm in runlevel 5
# xdm is now a separate service
x:5:respawn:/etc/X11/prefdm -nodaemon

```

Listing 8-10 (continued)

Managing rc Scripts by Hand

If you want to configure which services are started at boot time, you need to edit the rc scripts for the appropriate runlevel. The default runlevel is 3, which is full multiuser mode without a graphical interface and runlevel 5 with a graphical interface. So, to change the services that are started in the default runlevel, you should edit the scripts found in `/etc/rc3.d`, or `/etc/rc5.d` depending on your system.

When you look at a directory listing of the rc directories, notice that the files either start with S or K. The files that start with S are startup files, and the files that start with K are kill files. The S scripts are run in the numerical order listed in their filenames. It should be mentioned that if a startup script is set to S15, the K script should be K85 (or in general, SN becomes SM with $M = 100 - n$; the idea being the last started service is the first killed).

Note that case is important. Scripts that do not start with a capital *S* do not run upon startup. One good way to keep scripts from starting up at boot time without deleting them is to rename the file with a small *s* at the beginning instead of a capital *S*. This way you can always put the script back into the startup configuration by capitalizing the initial letter.

When the system starts up, it runs through the scripts in the *rc* directory of the runlevel it's starting up in. So when the system starts up in runlevel 3, it runs the scripts in the */etc/rc3.d* directory.

Looking at the directory listing included in Figure 8-2, you can see that the first few services start in this order: *kudzu*, *reconfig*, *iptables*, and *network*. That is so because their scripts are named *S05kudzu*, *S06cpuspeed*, *S08iptables*, and *S10network*, respectively.

Kudzu is called first because it detects new hardware. *cpuspeed* runs to put into effect speed stepping for the CPU. *iptables* then starts up the built-in firewall. *Network* then brings up the system's network interfaces. As you can see, the order in which these services are started makes a lot of sense, and their order is enforced by the way their *rc* startup scripts are named. All of the files in *rc#.d* are symbolic links to */etc/init.d* scripts, and the names are used here only to affect what services start or stop and the ordering of those services. Editing the *rc3.d/httpd* file will affect *rc5.d/httpd*.

The *S* scripts are started in this order until they have all been started. When the system shuts down, the corresponding *K* or kill scripts are run to shut down the services started from the *rc* directory. In general, every *S* script should have a corresponding *K* script to kill the service at shutdown.

If you can't find the corresponding *K* script in the startup directory, it is probably located in the shutdown directory. When the system is shut down, it enters runlevel 6. So, most *K* scripts are in */etc/rc6.d*. A typical */etc/rc6.d* directory listing is shown Figure 8-3.

```

root@terry:/etc/rc3.d
[ root@terry rc3.d ]# ls
K01apt      K85ndmpd      S13portmap    S56xinetd
K01yum      K89named      S14nfslock    S58ntpd
K02NetworkManager K89netplugd  S15mdmonitor  S80sendmail
K05sasauthd K90bluetooth  S18rpcgssd    S87iim
K10lirc     K91isdn       S19rpcidmapd  S90cron
K10psacct   K94diskdump   S19rpcsvcgssd S90xfs
K15gpm      K96init.cssd  S25netfs      S95anacron
K15httpd    K99microcode_ctl S26apmd       S95atd
K20nfs      K99readahead  S26lmsensors  S96init.cssd
K24irda     K99readahead_early S28autofs     S97messagebus
K30spamassassin S05kudzu      S33nifd       S97rhnsd
K35vncserver S06cpuspeed   S34mDNSResponder S98 cups-config-daemon
K35winbind  S08iptables   S40smartd     S98hald
K36lisa     S09pcmcia     S40snortd     S99local
K50netdump  S10network    S44acpid      S99usermin
K73ypbind   S12syslog     S55 cups      S99webmin
K74nscd     S13irqbalance S55sshd
[ root@terry rc3.d ]#

```

Figure 8-2 A directory listing of the *rc3.d* directory.



Figure 8-3 The file contents of the `/etc/rc6.d` directory.

If you ever need to restart a service that's started from an rc directory, an easy way to do it properly is to run its startup script with the restart option. This procedure enables all the proper steps to be followed (configuration files read, lock files released, and so forth) when the service starts up again. So, to restart syslog, for example, run the following command from the rc directory:

```
[root@terry rc3.d]# ./S12syslog restart
Shutting down kernel logger:          [ OK ]
Shutting down system logger:         [ OK ]
Starting system logger:               [ OK ]
Starting kernel logger:               [ OK ]
```

Managing rc Scripts Using chkconfig

Fedora Core and Red Hat Enterprise Linux come with a useful tool called `chkconfig`. It helps the system administrator manage rc scripts and xinetd configuration files without having to manipulate them directly. It is inspired by the `chkconfig` command included in the IRIX operating system.

Type **`chkconfig --list`** to see all the services `chkconfig` knows about, and whether they are stopped or started in each runlevel. An abridged example output is shown in the following listing. The `chkconfig` output can be a lot longer than that listed here, so be prepared to pipe it through `less` or `more`.

The first column is the name of the installed service. The next seven columns each represent a runlevel, and tell you whether that service is turned on or off in that runlevel.

Since `xinetd` is started on the system whose `chkconfig` output is excerpted, at the end of `chkconfig`'s report is a listing of what `xinetd` started services are

configured to begin at boot time. The listing is abridged, since a lot of services can be started from xinetd, and there's no need to show all of them.

Listing 8-11 shows how chkconfig can be an effective tool for handling all your network services and controlling which ones get started up at boot time. This is the output of `chkconfig --list`:

```

atd          0:off  1:off  2:off  3:on   4:on   5:on   6:off
rwhod        0:off  1:off  2:off  3:off  4:off  5:off  6:off
keytable     0:off  1:on   2:on   3:on   4:on   5:on   6:off
nscd         0:off  1:off  2:off  3:off  4:off  5:off  6:off
syslog       0:off  1:off  2:on   3:on   4:on   5:on   6:off
gpm          0:off  1:off  2:on   3:off  4:off  5:off  6:off
kudzu        0:off  1:off  2:off  3:on   4:on   5:on   6:off
kdcrotate    0:off  1:off  2:off  3:off  4:off  5:off  6:off
lpd          0:off  1:off  2:on   3:on   4:on   5:on   6:off
autofs       0:off  1:off  2:off  3:on   4:on   5:on   6:off
sendmail     0:off  1:off  2:on   3:off  4:off  5:off  6:off
rhnsd        0:off  1:off  2:off  3:on   4:on   5:on   6:off
netfs        0:off  1:off  2:off  3:off  4:off  5:off  6:off
network      0:off  1:off  2:on   3:on   4:on   5:on   6:off
random       0:off  1:off  2:on   3:on   4:on   5:on   6:off
rawdevices   0:off  1:off  2:off  3:on   4:on   5:on   6:off
apmd         0:off  1:off  2:on   3:off  4:off  5:off  6:off
ipchains      0:off  1:off  2:off  3:off  4:off  5:off  6:off
<snip>

xinetd based services:
  rexec:    off
  rlogin:   off
  rsh:      off
  chargen:      off
  chargen-udp:  off
  daytime:      off
  daytime-udp:  off
  echo:         off
  echo-udp:     off
  time:         off
  time-udp:     off
  finger:      off
  ntalk:       off
  talk:        off
  telnet:      off
  wu-ftp:      on
  rsync:       off
  eklogin:     off
  gssftp:      off
  klogin:      off

```

Listing 8-11 Output from `chkconfig --list`.

To turn a service off or on using `chkconfig`, use this syntax:

```
chkconfig -level[0-6](you must choose the runlevel) servicename off|on|reset
```

So, to turn off the `gpm` daemon turned on previously, type:

```
chkconfig --level 2 gpm off
```

To turn on `xinetd`, type:

```
chkconfig xinetd on
```

Run `chkconfig --list` again to see if the service you changed has been set to the state you desire. Changes you make with `chkconfig` take place the next time you boot up the system. You can always start, stop, or restart a service by running `service (service name)` from a terminal prompt.

Summary

All systemwide configuration files are located in `/etc`. So, if you want to change something across the system, look in `/etc` and its subdirectories first. If you're at a loss in terms of figuring out which configuration file you need to edit, try grepping for keywords in `/etc`.

To change configuration variables for one or a few users, you can usually edit configuration files within the individual users' home directories. Most configuration files in home directories start with a `.` so you need to look for them with the `ls -a` command.

Be mindful of configuration file permissions to ensure that unauthorized parties cannot modify them. Flat out instant root access for unauthorized parties is one possible outcome of a modified configuration file. A more likely outcome is that a configuration file modification would make it easier for a system compromise to take place.

You can either edit startup files by hand or by using one of the system administration tools such as `chkconfig`. You should at least know the format of the startup files and where they are, so that if automatic tools can't do the job for some reason, you can always change things yourself.

PART

Two

Network Services

- Chapter 9:** Managing the X Window System
- Chapter 10:** Configuring Printers
- Chapter 11:** TCP/IP Networking
- Chapter 12:** The Network File System
- Chapter 13:** The Network Information System
- Chapter 14:** Connecting to Microsoft and Novell Networks
- Chapter 15:** Configuring a Database Server
- Chapter 16:** Creating a VNC Server
- Chapter 17:** Providing Additional Network Services
- Chapter 18:** Optimizing Network Services

Managing the X Window System

IN THIS CHAPTER

- Configuring the X Server with the X Configuration Tool
- Manually Configuring the X Server

This chapter introduces you to the X Window system that is used to provide a graphical user interface (GUI) to the operating system. If you have used MS Windows or Apple Macs, then you are already familiar with using a GUI. Fedora Core and Enterprise Linux also give you similar features.

Configuring the X Server with the X Configuration Tool

You have basically two ways to configure the X server on your Fedora Core or Enterprise Linux system. First, you can use the *X Configuration tool*, which is a graphical tool that gives you the ability to change some of the most significant settings, such as display, monitor, and video card settings. The X Configuration tool is a graphical front end to the X configuration file, `xorg.conf`, which is located in the `/etc/X11` directory. Any changes that you make using the graphical utility are written to the `/etc/X11/xorg.conf` file. Second, you can edit the X Configuration file directly by using a text-editing application. In this section, I show you how to make X server configuration changes by using the X Configuration tool, beginning with changing the display resolution.

Changing the Display Resolution

The X Configuration tool makes it easy for you to change your display resolution. To change your display resolution, do the following:

1. On Enterprise Linux 4 choose Applications ⇄ System Settings ⇄ Display to open the Display Settings dialog box, shown in Figure 9-1. On Fedora Core 4 choose Desktop ⇄ System Settings ⇄ Display to open the Display Settings dialog box.

NOTE If you are not logged in as root, you will be prompted to enter the root password.

2. Select your desired resolution from the drop-down Resolution list.
3. Click OK to accept your choice, and close the dialog box.

NOTE Any time you make changes to the X server configuration, you must restart the X server for the changes to take effect. When using the X Configuration tool, you see a window reminding you to restart the X server.



Figure 9-1 Change your screen resolution here.

Changing the Display Color Depth

The system display color depth setting determines the number of colors that are shown on the display. A higher color depth displays more colors on the monitor. To change the system color depth, do the following:

1. On Enterprise Linux 4 choose Applications ⇨ System Settings ⇨ Display to open the Display Settings dialog box. On Fedora Core 4 choose Desktop ⇨ System Settings ⇨ Display to open the Display Settings dialog box. (Refer to Figure 9-1.)
2. Select your desired color depth from the Color Depth drop-down list.
3. Click OK to accept your choice and close the dialog box.

Changing Monitor Type Settings

The Fedora Core or Enterprise installer can usually detect the type of monitor that is connected to your system and set the configuration accordingly. Sometimes, however, the installer might not properly configure your monitor, requiring you to change the monitor settings. You also want to change your monitor settings if you get a different monitor with different parameters than your previous monitor. To change your monitor settings, do the following:

1. On Enterprise Linux 4 choose Applications ⇨ System Settings ⇨ Display to open the Display Settings dialog box. On Fedora Core 4 choose Desktop ⇨ System Settings ⇨ Display to open the Display Settings dialog box. (Refer to Figure 9-1.)
2. Click the Hardware tab. (See Figure 9-2.)
3. Click the top Configure button (to the right of the monitor type listing) to open the Monitor dialog box, shown in Figure 9-3.
4. Find the manufacturer of your monitor in the list, and then click the arrow to the left of the manufacturer's name to see a list of models.
5. Click the model number that matches your monitor.
6. Click OK twice to accept your choice, and exit the Display Settings dialog box.

TIP If you can't find your monitor manufacturer or model number on the monitor list, choose one of the generic monitors from the top of the monitor list.

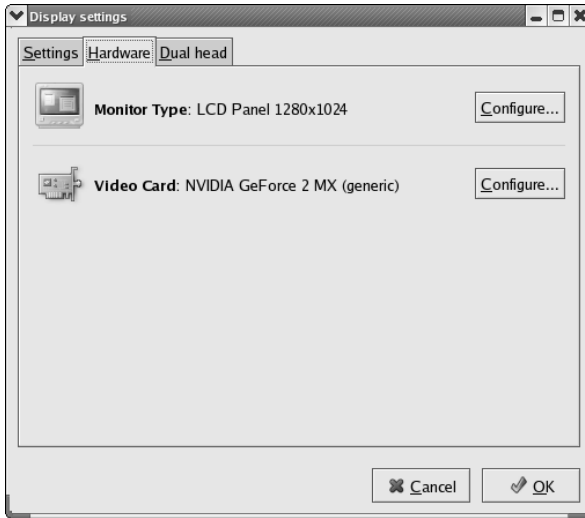


Figure 9-2 Access monitor and video card settings here.

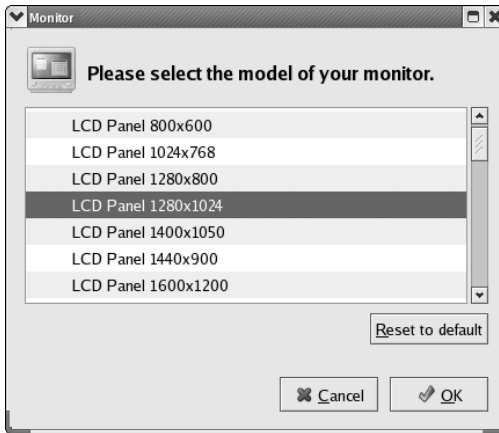


Figure 9-3 Choose your monitor type.

Changing Your Video Card Type

The Fedora Core or Enterprise installer can usually detect the type of video card that is connected to your system and set the configuration accordingly. However, if the installer doesn't properly detect your video card, you might need to change the video card type. You would also want to change your video card type if you install a different video card. To change your video card type, do the following:

1. On Enterprise Linux 4 choose Applications ⇨ System Settings ⇨ Display to open the Display Settings dialog box. On Fedora Core 4 choose Desktop ⇨ System Settings ⇨ Display to open the Display Settings dialog box. (Refer to Figure 9-1.)
2. Click the Hardware tab. (Refer to Figure 9-2.)
3. Click the bottom Configure button (to the right of the video card type listing) to display the Video Card dialog box, shown in Figure 9-4.
4. Find the manufacturer of your video card in the list, and click the appropriate model.

Configuring Dual Monitors

In Fedora Core or Enterprise Linux, you can use two video cards and monitors on your system if you desire. To configure a second video card and monitor, do the following:

1. On Enterprise Linux 4 choose Applications ⇨ System Settings ⇨ Display to open the Display Settings dialog box. On Fedora Core 4 choose Desktop ⇨ System Settings ⇨ Display to open the Display Settings dialog box. (Refer to Figure 9-1.)
2. Click the Dual Head tab, shown in Figure 9-5, in the Display Settings dialog box.

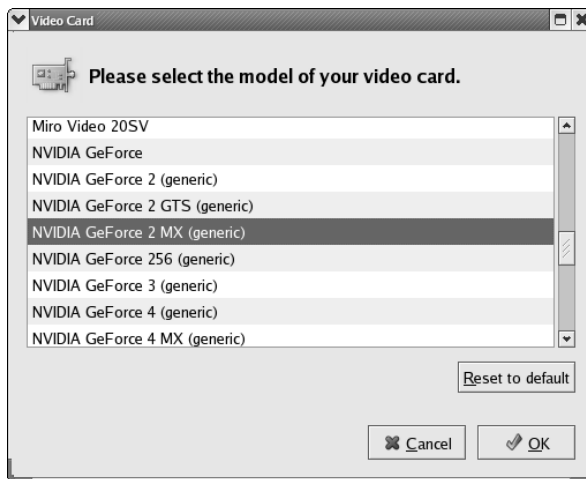


Figure 9-4 Configure your video card.

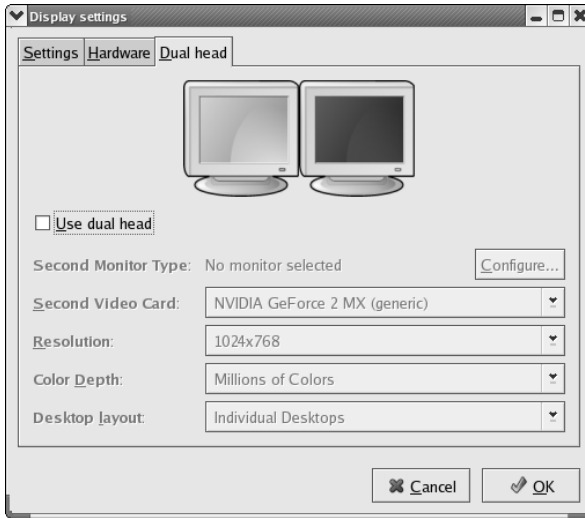


Figure 9-5 Use the Dual Head tab to configure dual monitors.

3. Select the Use Dual Head check box.
4. Click the Configure button (next to the Second Monitor Type), choose your monitor from the list, and then click OK.
5. Enter the appropriate information for the video card type, display resolution, and color depth.
6. Select whether you want individual desktops on each display or a single desktop spanning both displays by selecting the appropriate choice.
7. Click OK twice to exit the configuration tool.

Manually Configuring Your X Server

The Fedora Core or Enterprise installer program usually does a good job of detecting the system mouse, keyboard, video card, and monitor. When you log in to your system, the settings are generally what they should be for a functioning X server. You may never need to configure your X server manually, but if you do, it would be good to know how. In this section you look at the X server configuration file `/etc/X11/xorg.conf`.

The X Server Configuration File

The X server configuration file, like most configuration files in Fedora Core or Enterprise Linux is a plain-text file. Listing 9-1 is a typical X server configuration

file that was created during system installation. A description of the section of the file follows the file.

```
# XFree86 4 configuration created by redhat-config-xfree86
Section "ServerLayout"
    Identifier      "Default Layout"
    Screen 0        "Screen0" 0 0
    InputDevice     "Mouse0" "CorePointer"
    InputDevice     "Keyboard0" "CoreKeyboard"
    InputDevice     "DevInputMice" "AlwaysCore"
EndSection

Section "Files"
# RgbPath is the location of the RGB database.
#Note, this is the name of the
# file minus the extension (like ".txt" or ".db").
#There is normally no need to change the default.
# Multiple FontPath entries are allowed (they are
# concatenated together)
# By default, Red Hat 6.0 and later now use a font
#server independent of the X server to render fonts.
    RgbPath         "/usr/X11R6/lib/X11/rgb"
    FontPath        "unix/:7100"
EndSection

Section "Module"
    Load "dbe"
    Load "extmod"
    Load "fbdevhw"
    Load "glx"
    Load "record"
    Load "freetype"
    Load "type1"
    Load "dri"
EndSection

Section "InputDevice"

# Specify which keyboard LEDs can be user-controlled
#(eg, with xset(1))
#    Option "Xleds"          "1 2 3"
# To disable the XKEYBOARD extension, uncomment
# kbDisable.
#    Option "XkbDisable"

# To customise the XKB settings to suit your
# keyboard, modify the lines below (which are the
# defaults). For example, for a non-U.S.
# keyboard, you will probably want to use:
```

Listing 9-1 The X server configuration file `/etc/X11/xorg.conf`. (*continued*)

```

#       Option  "XkbModel"  "pc102"
# If you have a US Microsoft Natural keyboard, you
# can use:
#       Option  "XkbModel"  "microsoft"
#
# Then to change the language, change the Layout
# setting.
# For example, a german layout can be obtained with:
#       Option  "XkbLayout"  "de"
# or:
#       Option  "XkbLayout"  "de"
#       Option  "XkbVariant" "nodeadkeys"
#
# If you'd like to switch the positions of your
# capslock and control keys, use:
#       Option  "XkbOptions"  "ctrl:swapcaps"
# Or if you just want both to be control, use:
#       Option  "XkbOptions"  "ctrl:nocaps"
#
#       Identifier "Keyboard0"
#       Driver     "kbd"
#       Option     "XkbModel"  "pc105"
#       Option     "XkbLayout" "us"
EndSection

Section "InputDevice"
    Identifier "Mouse0"
    Driver     "mouse"
    Option     "Protocol"  "IMPS/2"
    Option     "Device"    "/dev/input/mice"
    Option     "ZAxisMapping" "4 5"
    Option     "Emulate3Buttons" "yes"
EndSection

Section "InputDevice"

# If the normal CorePointer mouse is not a USB mouse #then this input
device can be used in AlwaysCore #mode to let you also use USB mice at
the same time.
    Identifier "DevInputMice"
    Driver     "mouse"
    Option     "Protocol"  "IMPS/2"
    Option     "Device"    "/dev/input/mice"
    Option     "ZAxisMapping" "4 5"
    Option     "Emulate3Buttons" "no"
EndSection

Section "Monitor"
    Identifier "Monitor0"

```

Listing 9-1 (continued)

```

        VendorName      "Monitor Vendor"
        ModelName        "LCD Panel 1280x1024"
        DisplaySize      376    301
        HorizSync         31.5  - 67.0
        VertRefresh       50.0  - 75.0
        Option            "dpms"
    EndSection

    Section "Device"
        Identifier        "Videocard0"
        Driver             "nv"
        VendorName        "Videocard vendor"
        BoardName          "NVIDIA GeForce 2 MX (generic)"
        VideoRam           32768
    EndSection

    Section "Screen"
        Identifier        "Screen0"
        Device             "Videocard0"
        Monitor            "Monitor0"
        DefaultDepth       24
        SubSection "Display"
            Depth          24
            Modes           "1024x768" "800x600" "640x480"
        EndSubSection
    EndSection

    Section "DRI"
        Group              0
        Mode                0666
    EndSection

```

Listing 9-1 *(continued)*

This file contains configuration information for the fonts used by the system, the keyboard type and layout, and the video card, monitor, and displays. There is a section in the file for each of these items. The sections and their uses are shown in Table 9-1.

Table 9-1 Sections Names and Their Uses

SECTION NAME	USED FOR
Files	File pathnames for fonts
Server Flags	Server Flags
Module	Dynamic module loading

(continued)

Table 9-1 (continued)

SECTION NAME	USED FOR
Input Device	Keyboards, mice, and other input devices
Device	Video card information
VideoAdapter	Not used
Monitor	Monitor description
Modes	Video mode descriptions
Screen	Display configuration
Server Layout	General description of the X server
DRI	Direct Rendering Infrastructure information

Listing 9-2 is the section listing for the screen section. Each of the sections follows a similar pattern beginning with the word `Section`, followed by the name of the device, then information or configuration details about the device, and finally ending with `EndSection`.

```
Section "Screen"
    Identifier "Screen0"
    Device     "Videocard0"
    Monitor    "Monitor0"
    DefaultDepth 24
    SubSection "Display"
        Depth    24
        Modes    "1024x768" "800x600" "640x480"
    EndSubSection
EndSection
```

Listing 9-2 A typical section of the `xorg.conf` file.

Making changes to the `xorg.conf` file is easy. For example, to change the default color depth you just need to change the `DefaultDepth` number 24 shown in Listing 9-2 to 8, 16, or 32.

The modes line specifies the screen size in pixels that the X server will try to use in order beginning with the first one. If you wanted to add another screen resolution, you would enter the information into the `Modes` line of the display subsection. For example if you wanted to add 1200×1024 , you would enter `1200x1024` to the beginning of the `Modes` line.

NOTE You can toggle through the modes by holding **Ctrl+Alt** and pressing **plus** or **minus**.

Now you know what to do to change the screen section of the `/etc/X11/xorg.conf` file. Changing the information in the other sections is similar.

NOTE Refer to the `xorg.conf` man pages for a complete description of the `xorg.conf` file. The `xorg.conf` man pages are included with every installation of Fedora Core or Enterprise Linux and are an excellent resource that provides great detail about the `xorg.conf` file. To access the `xorg.conf` man page type `man xorg.conf` at a terminal prompt.

Summary

In this chapter you learned about configuring the X server on your system. First you looked at using the graphical X configuration tool to make changes to the X server configuration file. Then you looked at the configuration file itself and how to change some settings manually. Finally, you learned that the best place to get complete details about the `xorg.conf` file is the `xorg.conf` man pages.

Configuring Printers

IN THIS CHAPTER

- Configuring Printers with the Printer Configuration Tool
- Editing the Printer Configurations
- Managing Print Jobs

In this chapter you learn how to successfully configure and manage your system printers. Included with Fedora Core and Red Hat Enterprise Linux is the Printer Configuration tool to help you configure your printing system. The Printer Configuration tool is an easy-to-use, graphical tool that will help you to set up whatever type of printer you choose.

After your printer is configured, you might want to gather information about jobs that you sent to the printer. You will also want to be able to change print job priority, see the contents of your print queue, and maybe even delete some of your scheduled print jobs. You will be able to do all these functions after going through this chapter.

Configuring Printers with the Printer Configuration Tool

Because the Printing Configuration tool is a graphical-based utility, you can start it by choosing it from the Applications menu. To start the Printer Configuration tool, follow these steps:

1. In Enterprise Linux choose Applications ⇄ System Settings ⇄ Printing.
In Fedora Core 4 choose Desktop ⇄ System Settings ⇄ Printing.

NOTE If you aren't logged in as the root user, the system prompts you for the root password before you can continue.

2. Type the root password, if necessary. The Printer Configuration tool opens, as shown in Figure 10-1.
3. Click the New button in the main Printer Configuration tool window. The window shown in Figure 10-2 appears.
4. Click the Forward button. The Queue Name screen appears, as shown in Figure 10-3.
5. Enter a unique name for the printer in the Name text field. Choose a descriptive name for your printer, and follow these parameters:
 - The printer name must begin with a letter and can't contain spaces.
 - You can use any valid characters for the remainder of the printer name. The valid characters are lowercase and uppercase letters a through z, numeral 0 through 9, – (dash), and _ (underscore).
 - If you want, you can enter a description for the printer in the Short Description field.

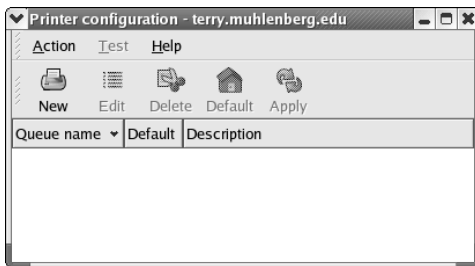


Figure 10-1 The Printer Configuration tool.

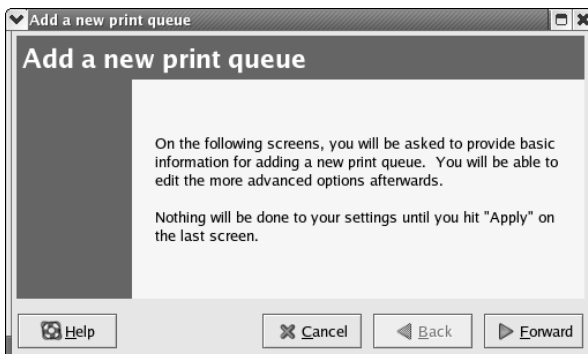


Figure 10-2 The Add a new print queue window.

6. When you finish entering a name for your printer, click Forward. The Queue Type window appears, as shown in Figure 10-4, and the Printer Configuration tool attempts to detect your printer.

The following sections detail the various possibilities available for configuring your print queue and selecting your print driver.

Configuring the Print Queue

You can configure six types of print queues. A *print queue* is a directory that holds the print jobs for the type of printer that you configure to work with the queue. The print queue is associated with the type of printer that you want to configure. At the top of the Queue Type window is a drop-down list containing the six types of print queues that you can configure. The queue type is set to Locally-Connected by default. If the printer is connected *locally* — that is, to either the parallel or the USB port on the PC, and is also recognized — it is shown in the list.

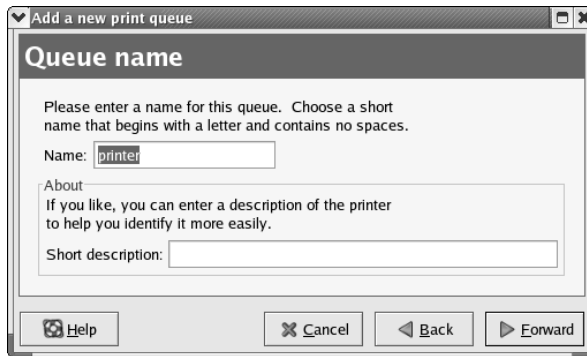


Figure 10-3 The Queue name window.

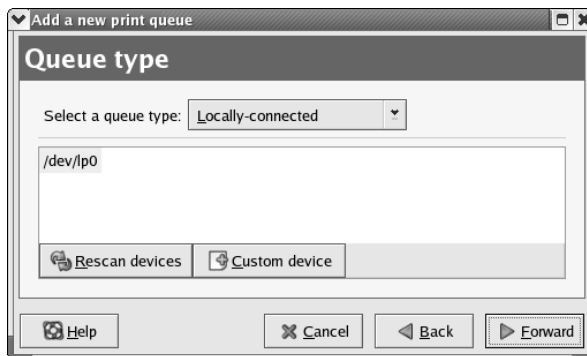


Figure 10-4 The Queue type window.

The following list shows the six types of queue that you can install; to choose one, select the type that you desire from the drop-down list.

- **Locally-Connected** — A printer attached directly to your computer through a parallel or USB port. If your printer isn't listed, click the Custom Device button, type the name of your printer, and then click OK to add it to the printer device list. A printer attached to the parallel port is usually referred to as `/dev/lp0`. A printer attached to the USB port is usually referred to as `/dev/usb/lp0`.

NOTE When you set up a locally connected printer it is set up as a local CUPS printer that only the localhost can use. If you want to use the printer on your local network you need to modify the `/etc/cups/cupsd.conf` file to allow other systems on your network to access the printer.

- **Networked CUPS (IPP)** — A printer that can be accessed over a TCP/IP network. The Common Unix Printing System (CUPS) is based on the Internet Printing Protocol (IPP), which was created in an attempt to set some standards for printing over the Internet. If you choose this type of queue, you need to enter the server name and the path to the server. Figure 10-5 shows the Networked CUPS queue dialog box.
- **Networked UNIX (LPD)** — A printer attached to a different UNIX system that can be accessed over a TCP/IP network (for example, a printer attached to another Red Hat Linux system on your network). If you choose this type of queue, you need to enter the server name and path to the server, as shown in Figure 10-6.

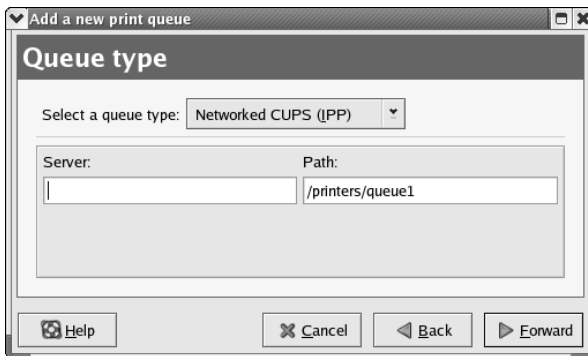


Figure 10-5 The Networked CUPS (IPP) screen is where you enter the server name and path.

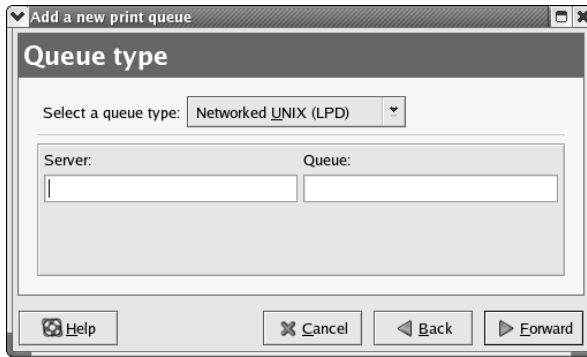


Figure 10-6 Enter the server and queue for a networked UNIX (LPD) printer.

- *Server* — The hostname or IP address of the remote machine to which the printer is attached.
- *Queue* — The remote printer queue. The default printer queue is usually `lp`. By default, the Strict RFC1179 Compliance option is not chosen. If you are having problems printing to a non-Linux `lpd` queue, choose this option to disable enhanced `lpr` printing features. LPR is an older printing protocol used by many UNIX systems.

NOTE The remote machine must be configured to allow the local machine to print on the desired queue. As root, create the file `/etc/hosts.lpd` on the remote machine to which the printer is attached. On separate lines in the file, add the IP address or hostname of each machine that should have printing privileges.

- **Networked Windows (SMB)** — A printer attached to a different system that shares a printer over an SMB network (for example, a printer attached to a Microsoft Windows machine). The Server Message Block (SMB) protocol is the native protocol that computers running Windows use to communicate with each other. See Figure 10-7.

On this screen, you see a list of shares from which you can select the networked Windows printer that you want to use. To the left of the share name is an arrow that you can click to expand the share listing and show any configured printers. Figure 10-7 shows the RHL10 share expanded and also lists three printers. Click the printer that you wish to use and then click Forward. An Authentication screen appears, as shown in Figure 10-8.

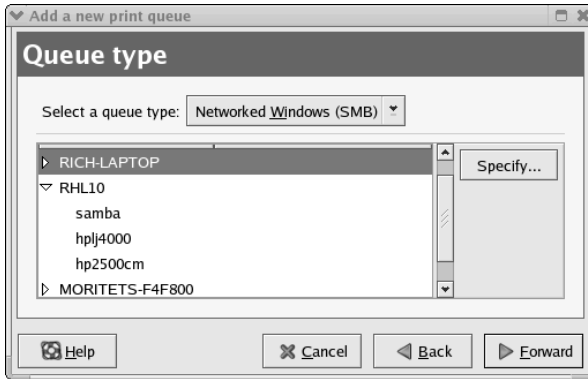


Figure 10-7 Configuring the Networked Windows (SMB) printer screen.

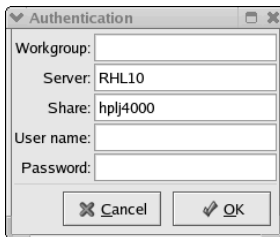


Figure 10-8 The Authentication screen for connecting to a SMB printer.

Text fields for the following options appear as shown in Figure 10-8:

- *Workgroup* — The name of your Windows workgroup needs to be entered here.
- *Server* — The name of the print server needs to be entered here.
- *Share* — This is the name of the shared printer on which you want to print. This name must be the same name defined as the Samba printer on the remote Windows print server.
- *User Name* — This is the name by which you must log in to access the printer. This user must exist on the Windows system, and the user must have permission to access the printer. The default user name is typically *guest* for Windows servers or *nobody* for Samba servers.
- *Password* — The password (if required) for the user specified in the User Name field.
- **Networked Novell (NCP)** — A printer attached to a different system that uses the Novell NetWare networking technology. After choosing this type of queue, you need to enter additional information into the Queue Type window, as shown in Figure 10-9.

You need to enter information for the following fields in Figure 10-9:

- *Server* — The host name or IP address of the NCP system to which the printer is attached.
- *Queue* — The remote queue for the printer on the NCP system.
- *User* — The name by which you must log in to access the printer.
- *Password* — The password for the user specified in the User field.
- **Networked JetDirect** — A printer connected directly to the network through HP JetDirect instead of to a computer. (See Figure 10-10.)

You need to enter the appropriate information for the following text fields:

- *Printer* — The hostname or IP address of the JetDirect printer.
- *Port* — The port on the JetDirect printer that is listening for print jobs. The default port is 9100.

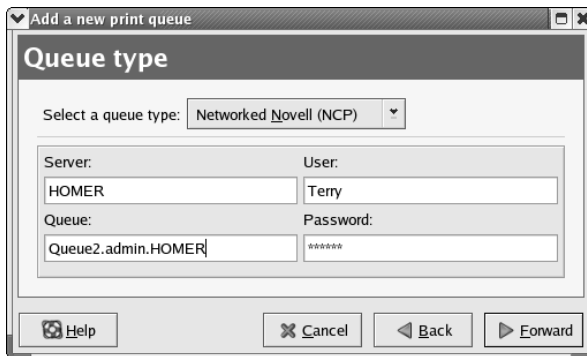


Figure 10-9 Configuring a networked Novell (NCP) printer.

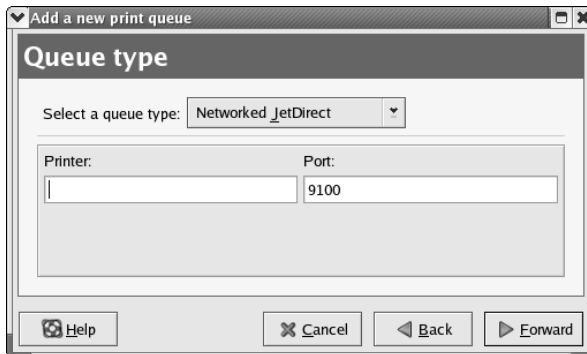


Figure 10-10 Configuring a networked JetDirect printer.

NOTE Whenever you add a new print queue or change an existing one, you must restart the printer daemon for the changes to take effect. See the upcoming “Editing the Printer Configuration” section. In case you are wondering what a *daemon* is, it means *disk and execution monitor*. It is basically a program that runs in the background, waiting for some event to occur. In this case, the printer daemon is waiting for print jobs.

Selecting the Print Driver

The next step in configuring a printer is to select the print driver. The *print driver* processes the data that you want to print into a format that the printer can understand.

1. After you select the queue type of the printer and enter the required information, click Forward to go to the Printer Model window, shown in Figure 10-11.
2. Select the driver from the list.
 - a. Click the arrow beside the manufacturer for your printer.
 - b. Find your printer from the expanded list, and then click the arrow beside the printer’s name. A list of drivers for your printer appears. The printers are listed by manufacturer.
 - c. Select one appropriate for your printer. Sometimes you might need to try several of the listed drivers to find one that works properly.

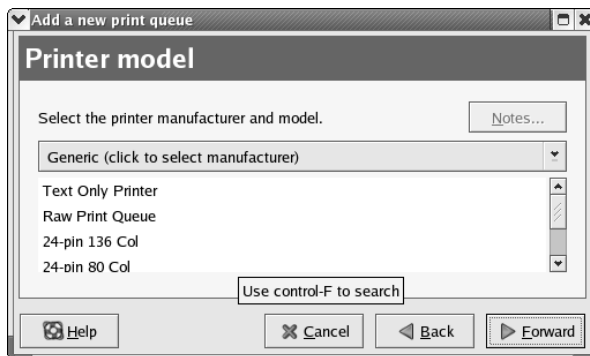


Figure 10-11 Select the printer manufacturer and model.

NOTE To read more about the print drivers, go to www.linuxprinting.org /`printer_list.cgi`. You can select a different print driver after adding a printer by starting the Printer Configuration tool, selecting the printer from the list, clicking Edit, clicking the Printer Driver tab, selecting a different print driver, and applying the changes.

3. Click Forward to go to the printer information confirmation page where you can verify your printer configuration choices.
 - a. Click Apply to add the print queue if the settings are correct.
 - b. Click Back to modify the printer configuration if necessary.
4. Click the Apply button in the main window to save your changes to the printer configuration file and restart the printer daemon (`lpd`), or click Back to modify the printer configuration if necessary. Be sure to test your configuration by printing a test page.

Editing the Printer Configuration

After adding your printer(s), you can edit settings by selecting the printer from the printer list after opening the Printer Configuration tool and then clicking the Edit button. The tabbed window shown in Figure 6-12 appears. The window contains the current values for the printer that you selected to edit. Make any changes and click OK. Then click Apply in the main Printer Configuration tool window to save the changes and restart the printer daemon.

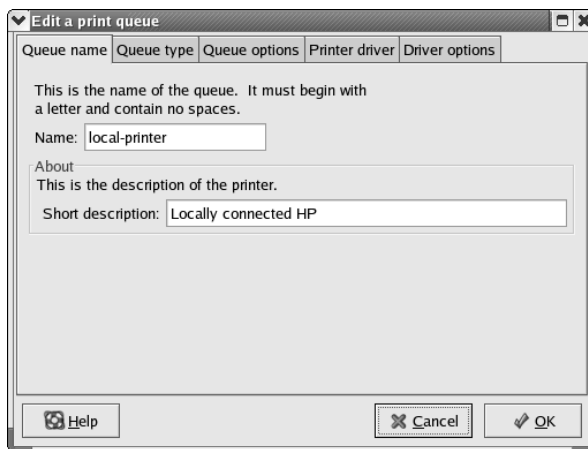


Figure 10-12 The Edit a print queue screen.

The tabs and what they hold are as follows:

- **Queue Name** — To rename a printer, change the value of Name on the Queue Name tab. Click OK to return to the main window. The name of the printer changes in the printer list. Click Apply to save the change and restart the printer daemon.
- **Queue Type** — The Queue Type tab shows the queue type that you selected when adding the printer and its settings. You can change the queue type of the printer or just change the settings. After making modifications, click OK to return to the main window. Click Apply to save the changes and restart the printer daemon. Depending on which queue type you choose, you will see different options. Refer to the section of this chapter that describes your particular printer; options unique to your printer are listed there.
- **Queue Options** — From the Queue Options tab, you can select banner pages before and after your print job. You can also set the printable area of the page. To modify filter options, highlight the option and click Edit to modify or click Delete to remove it. Click OK to accept your changes and return to the main window. Click Apply to save the change and restart the printer daemon.
- **Printer Driver** — The Printer Driver tab shows which print driver is currently being used. This is the same list that you use when you add the printer. If you change the print driver, click OK to accept your changes and return to the main window. Then click Apply to restart the printer daemon.
- **Driver Options** — The Driver Options tab displays advanced printer options. Options vary for each print driver. Common options include:
 - Select Send Form-Feed (FF) if the last page of your print job is not ejected from the printer (for example, the form feed light flashes). If selecting this option does not force the last page out of the printer, try selecting Send End-of-Transmission (EOT) instead. Some printers require both Send Form-Feed (FF) and Send End-of-Transmission (EOT) to eject the last page.
 - Select Send End-of-Transmission (EOT) if sending a form feed does not work. Refer to the preceding bullet on the Send Form-Feed (FF) option.
 - Select Assume Unknown Data Is Text if your print driver does not recognize some of the data sent to it. Select it only if you are having problems printing. If this option is selected, the print driver assumes that any data it cannot recognize is text and tries to print it as text. If you select this option and the Convert Text to PostScript option, the

print driver assumes that the unknown data is text and then converts it to PostScript.

- Select Prerender PostScript if you are trying to print characters outside of the basic ASCII character set (such as foreign language characters) that won't print correctly. If your printer doesn't support the fonts you are trying to print, try selecting this option. You should also select this option if your printer cannot handle PostScript level 3. This option converts it to PostScript level 1.
- Convert Text to PostScript is selected by default. If your printer can print plain text, try deselecting this when printing plain-text documents to decrease the time it takes to print.
- Page Size allows you to select the paper size for your printer, such as US Letter, US Legal, A3, and A4.
- Effective Filter Locale defaults to C. If you are printing Japanese characters, select ja_JP. Otherwise, accept the default of C.
- Media Source defaults to Printer default. Change this option to use paper from a different tray.

If you modify the driver options, click OK to return to the main window. Then click Apply to save the changes and restart the printer daemon.

Deleting a Printer

To delete an existing printer, select the printer and click the Delete button on the toolbar. The printer will be removed from the printer list. Click Apply to save the changes and restart the printer daemon.

Setting the Default Printer

To set the default printer, select the printer from the printer list and click the Default button on the toolbar. The default printer icon appears in the first column of the printer list beside the default printer.

Managing Print Jobs

Whenever you print a document or image, the Print Notifier automatically opens and an icon appears on the panel at the top of the desktop. See Figure 10-13 to find out what the icon looks like.

You can open the Print Notifier by clicking the panel icon to open the window shown in Figure 10-14.



Figure 10-13 The Print Notifier icon on the top panel.

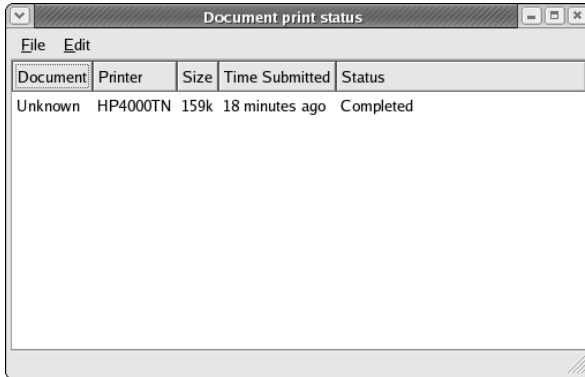


Figure 10-14 The Print Notifier window.

You can see a list of your print jobs and their status. You can choose a job by clicking it and then either right-click or choose Edit from the top menu. This will let you pause the selected job, delete the selected job, or resume printing a previously paused job.

Summary

In this chapter you learned about the Printer Configuration tool. With this tool you can configure many different types of printers for use on your system. You saw the procedure to add a printer, select a print queue, and choose the proper driver for your printer. You also learned how to modify an existing printer configuration and manage jobs that you sent to the printer.

CHAPTER 11 TCP/IP Networking

IN THIS CHAPTER

- TCP/IP Explained
- Understanding Network Classes
- Setting Up a Network Interface Card (NIC)
- Understanding Subnetting
- Working with Gateways and Routers
- Configuring Dynamic Host Configuration Protocol
- Configuring the Network Using the Network Configuration Tool
- Editing Your Network Configuration

This chapter provides an overview of the TCP/IP protocols as they apply to networking with Fedora Core and Red Hat Enterprise Linux. TCP/IP is complex, and many books have been written on this topic alone. If you want to learn more about TCP/IP, a good place to start is to use one of the Internet search engines to search for this topic on the Internet. After giving a description of TCP/IP, this chapter explains how to configure such a network in a Red Hat environment.

TCP/IP Explained

TCP/IP is an acronym for Transmission Control Protocol/Internet Protocol, and refers to a family of protocols used for computer communications. TCP and IP are just two of the separate protocols contained in the group of protocols developed by the Department of Defense, sometimes called the DoD Suite, but more commonly known as TCP/IP.

In addition to Transmission Control Protocol and Internet Protocol, this family also includes Address Resolution Protocol (ARP); Domain Name System (DNS); Internet Control Message Protocol (ICMP); User Datagram Protocol (UDP); Routing Information Protocol (RIP); Simple Mail Transfer Protocol (SMTP); Telnet, and many others. These protocols provide the necessary services for basic network functionality, and you will take a closer look at them for a better understanding of how the network works.

To be able to send and receive information on the network, each device connected to it must have an address. The address of any device on the network must be unique and have a standard, defined format by which it is known to any other device on the network. This device address consists of two parts:

- The address of the network to which the device is connected
- The address of the device itself — its node or host address

Devices that are physically connected to each other (not separated by routers) would have the same network number but different node, or host, numbers. This would be typical of an internal network at a company or university. These types of networks are now often referred to as *intranets*.

The two unique addresses I've been talking about are typically called the *network layer* addresses and the *Media Access Control (MAC)* addresses. Network Layer addresses are IP addresses that have been assigned to the device. The MAC address is built into the card by the manufacturer and refers to only the lowest-level address by which all data is transferred between devices.

Now that you know a little about addressing, you need to learn how the address, and also the data, is transmitted across the network. This transfer is accomplished by breaking the information into small pieces of data called packets or datagrams. Why is it necessary to use packets instead of just sending the entire message as one long stream of data? There are two reasons for this — sharing resources and error correction.

Let's look at the first, sharing resources. If two computers are communicating with each other, the line is busy. If these computers were sharing a large amount of data, other devices on the network would be unable to transfer their data. When long streams of data are broken into small packets, each packet is sent individually, and the other devices can send their packets between the packets of the long stream. Since each packet is uniquely addressed and has instructions on how to reassemble it, it does not matter the order that it arrives in or that it arrives in small pieces.

The second reason for breaking the data into packets is error correction. Because the data is transmitted across media that is subject to interference, the data can become corrupt. One way to deal with the corruption is to send a checksum along with the data. A checksum is a running count of the bytes sent in the message. The receiving device compares its total to the total transmitted. If these numbers are the same, the data is good; but if they are different, either

the checksum or the data itself is corrupt. The receiving device then asks the sender to resend the data. By breaking the data into small packets, each with its own checksum, it is easier to ensure that a good message arrives, and if not, only a small portion needs to be resent instead of the entire message.

In the description of packets, I mentioned unique addressing and reassembly instructions. Because packets also contain data, each is made up of two parts, the *header*, which contains the address and reassembly instructions, and the *body*, which contains the data. Keeping all this information in order is the *protocol*. The protocol is a set of rules that specifies the format of the package and how it is used.

Understanding Network Classes

As stated earlier, all addresses must have two parts, the network part and the node, or host, part. In this section you look at Ipv4 addresses. Ipv4 addresses used in TCP/IP networks are 4 bytes long, called IP addresses, and are written in standard dot notation, which means that a decimal number separated by dots (for example, 192.168.1.2). The decimal numbers must be within the numeric range of 0 to 255 to conform to the 1-byte requirement. IP addresses are divided into classes with the most significant being classes A, B, and C, depending on the value of the first byte of the address. Table 11-1 shows valid numbers for these classes.

The reason for the class division is to enable efficient use of the address numbers. If the division were the first 2 bytes to the network part, as shown in Table 11-1, and the last 2 bytes to the host part, then no network could have more than 2^{16} hosts. This would be impractical for large networks and wasteful for small networks.

There are a few ways to assign IP addresses to the devices, depending on the purpose of the network. If the network is internal, an intranet, not connected to an outside network, any class A, B, or C network number can be used. The only requirement is choosing a class that allows for the appropriate number of hosts to be connected. Although this is possible, in the real world this approach would not allow for connecting to the Internet.

Table 11-1 Network Classes and Their IP Number Range

CLASS	FIRST BYTE
Class A	0–127
Class B	128–191
Class C	192–233

A more realistic approach would be to register with one of the domain registration services and request an officially assigned network number. An organization called the InterNIC maintains a database of all assigned network numbers to ensure that each assignment is unique. After obtaining a network number, the host numbers may be assigned as required. Nearly all IP devices require manual configuration; you will look at assigning IP addresses later when you actually set up your own network. You have now seen that each device has a unique network and node address, which is called an IP address. Earlier, this was described as the Network Layer address. You also read about the Media Access Control, or MAC, address. The MAC address was defined as the lowest level at which communication occurs. On an Ethernet network, this address is also called the Ethernet address. This is the address that is ultimately necessary for transmission of data. For transfer to happen, the IP address must be mapped to the Ethernet address of the device. The mechanism that makes this possible is Address Resolution Protocol, or ARP.

To determine the Ethernet address of a node on the same network, the sending device sends an ARP request to the Ethernet broadcast address. The Ethernet broadcast address is a special address to which all Ethernet cards are configured to “listen.” The ARP request, containing the sender’s IP and Ethernet addresses, as well as the IP address it is looking for, asks each device for the Ethernet address that corresponds to a particular IP address. The device whose address matches the request sends a reply to the sender’s Ethernet address. The sender is then able to send its data to the specific address it received in response to its ARP request. This process works for sending data between devices on the same network, but what about sending data to devices on different networks? For this you need a router.

Routers enable networks not physically connected to each other to communicate. A router must be connected physically to each network that wants to communicate. The sending node must be able to send its request to a router on its own network, and the receiving node must also be on a network connected to a router. The sending node sends its request to the router on its network. This router is typically called the *default gateway*, and its address must be configured in the sending node’s configuration files. You will learn how to do this later in this chapter in the “Exploring Gateways and Routers” section.

The router receives the request from the sending node and determines the best route for it to use to transmit the data. The router has an internal program, called a *routing table*, which it uses to send the data, either to another router if the other network is not directly connected, or directly to the other network. If the destination network cannot be found in the routing table, then the packet is considered undeliverable and is dropped. Typically, if the packet is dropped, the router sends an ICMP Destination Unreachable message to the sender.

Routing tables can be manually configured or acquired dynamically. *Manual configuration* means that it is necessary for whoever is setting up the router to

provide all the information about other networks and how to reach them. This method is impractical because of the size of the file required and constantly changing information.

Dynamic acquisition means that the router sends a message using the Routing Information Protocol (RIP) or Open Shortest Path First (OSPF) protocol. These dynamic protocols enable routers to share details with other routers concerning networks and their locations.

Ultimately, the purpose of everything you have looked at so far — packets, IP addresses, and routing — is to give users access to services such as printing, file sharing, and email.

You have had a brief look at the IP part of the TCP/IP family of protocols and have arrived at TCP. Transmission Control Protocol is encapsulated in IP packets and provides access to services on remote network devices. TCP is considered to be a stream-oriented reliable protocol. The transmission can be any size because it is broken down into small pieces, as you have already seen. Data that is lost is retransmitted, and out-of-order data is reordered. The sender is notified about any data that cannot be delivered. Typical TCP services are File Transfer Protocol (FTP), Telnet, and Simple Mail Transfer Protocol (SMTP).

Setting Up a Network Interface Card (NIC)

Every Fedora Core and Red Hat Enterprise Linux distribution includes networking support and tools that can be used to configure your network. In this section you'll learn how to configure a computer for connection to an internal and external network.

NOTE This section tells you how to configure a network interface card from the command line by modifying the configuration files directly. If you would rather use a graphical based configuration utility, skip to the section titled, "Configuring the Network with the Network Configuration Tool."

Even if the computer is not connected to outside networks, internal network functionality is required for some applications. This address is known as the loopback device, and its IP address is 127.0.0.1. You should check that this network interface is working before configuring your network cards. To do this, you can use the `ifconfig` utility to get some information. If you type `ifconfig` at a console prompt, you will be shown your current network interface configuration. Figure 11-1 illustrates the output of the `ifconfig` command.

TIP Make sure that the loopback (IP address 127.0.0.1) is working before you begin to configure your network cards.

```

terry@terry:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0F:B0:45:97:76
          inet addr:192.168.9.153  Bcast:192.168.9.255  Mask:255.255.255.0
          inet6 addr: fe80::20f:b0ff:fe45:9776/64 Scope:Link
          IPX/Ethernet 802.3 addr:00001009:000F80459776
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:333205 errors:0 dropped:0 overruns:0 frame:0
          TX packets:81806 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:99184314 (94.5 MiB)  TX bytes:6658215 (6.3 MiB)
          Interrupt:10 Base address:0x4000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:30317 errors:0 dropped:0 overruns:0 frame:0
          TX packets:30317 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:19564605 (18.6 MiB)  TX bytes:19564605 (18.6 MiB)

terry@terry ~$

```

Figure 11-1 The `ifconfig` utility shows the current network interface configuration.

If your loopback is configured, the `ifconfig` shows a device called `lo` with the address `127.0.0.1`. If this device and address are not shown, you can add the device by using the `ifconfig` command as follows:

```
ifconfig lo 127.0.0.1
```

You then need to use the `route` command to give the system a little more information about this interface. For this, type:

```
route add -net 127.0.0.0
```

You now have your loopback set up, and the `ifconfig` command shows the device `lo` in its listing.

Configuring the Network Card

The procedure for configuring a network card is the same as that for configuring the loopback interface. You use the same command, `ifconfig`, but this time use the name '`eth0`' for an Ethernet device. You also need to know the IP address, the net mask, and the broadcast addresses. These numbers vary, depending on the type of network being built. For an internal network that never connects to the outside world, any IP numbers can be used; however, there are IP numbers typically used with these networks. Table 11-2 shows the IP numbers that are usually used for such networks.

NOTE The network interface card (NIC), if already installed, is detected and configured during system installation. You should check to determine whether your card is already configured before following the configuration instructions in this section. Use the `ifconfig` command to see the configured network cards.

Table 11-2 Reserved Network Numbers

NETWORK CLASS	NET MASK	NETWORK ADDRESSES
A	255.0.0.0	10.0.0.0–10.255.255.255
B	255.255.0.0	172.16.0.0–172.31.255.255
C	255.255.255.0	192.168.0.0–192.168.255.255

If you are connecting to an existing network, you must have its IP address, net mask, and broadcast address. You also need to have the router and domain name server (DNS) addresses.

In this example, you configure an Ethernet interface for an internal network. You need to issue the following command:

```
ifconfig eth0 192.168.2.5 netmask 255.255.255.0 broadcast 192.168.2.255
```

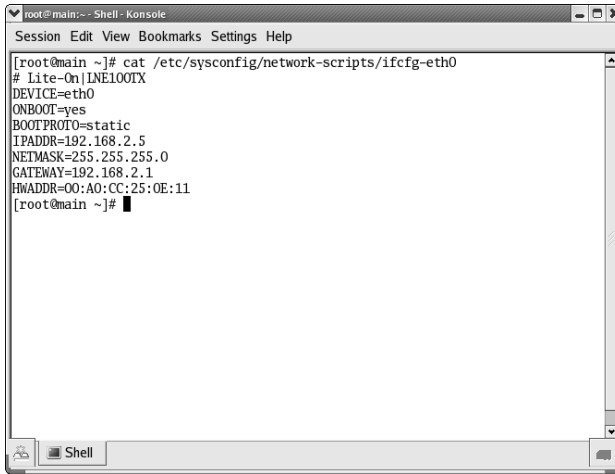
This results in the creation of device `eth0` with a network address of 192.168.2.5, a net mask of 255.255.255.0, and a broadcast address of 192.168.2.255. A file is created in `/etc/sysconfig/network-scripts` called `ifcfg-eth0`. A listing of this file, shown in Figure 11-2, shows the information that you just entered. The line `onboot=yes` tells the kernel to configure this device at system startup. The line `bootproto=static` means that the IP address was manually entered for the NIC. If you desire, you can use Dynamic Host Configuration Protocol, or DHCP, to obtain the required IP information for your NIC.

NOTE If you want to use DHCP to obtain the required IP information for your NIC, see the section “Configuring Dynamic Host Configuration Protocol” later in this chapter.

Configuring an Internal Network

Now you have a network device configured for one computer. To add additional computers to your network, you need to repeat this process on the other computers you want to add. The only change is that you need to assign a different IP address. For example, the second computer on your network could have the address 192.168.2.6, the third could have 192.168.2.7, and so on.

NOTE This section does not cover the physical requirements for building a network – cabling, hubs, and so forth. A good source for this information is *Network Plus* by David Groth (ISBN 0-7821-4014-9, published by Sybex).



```
root@main:~# cat /etc/sysconfig/network-scripts/ifcfg-eth0
# Lite-On|LNE100TX
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=static
IPADDR=192.168.2.5
NETMASK=255.255.255.0
GATEWAY=192.168.2.1
HWADDR=00:A0:CC:25:0E:11
root@main ~]#
```

Figure 11-2 The configuration file for the network device eth0.

In addition to configuring the network cards on each of the computers in the network, three files on each computer need to be modified. These files are all located in the `/etc` directory:

- `/etc/nsswitch.conf`
- `/etc/hosts`
- `/etc/resolv.conf`
- `/etc/sysconfig/network`

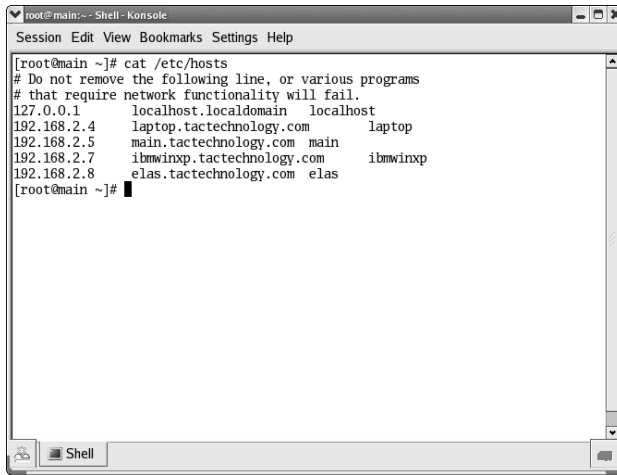
The `/etc/nsswitch.conf` file contains configuration information for the name resolver and should contain the following line:

```
hosts: files dns
```

This configuration tells the name resolver to check the `/etc/hosts` file before attempting to query a name server and to return all valid addresses for a host found in the `/etc/hosts` file instead of just the first.

The `/etc/hosts` file could contain the names of all the computers on the local network, or an outside network. For a small network, maintaining this file is not difficult, but for a large network, like the Internet, keeping the file up to date is often impractical. Figure 11-3 shows my home network, containing several computers. The first address represents the current system, and the other addresses represent other computers on the network.

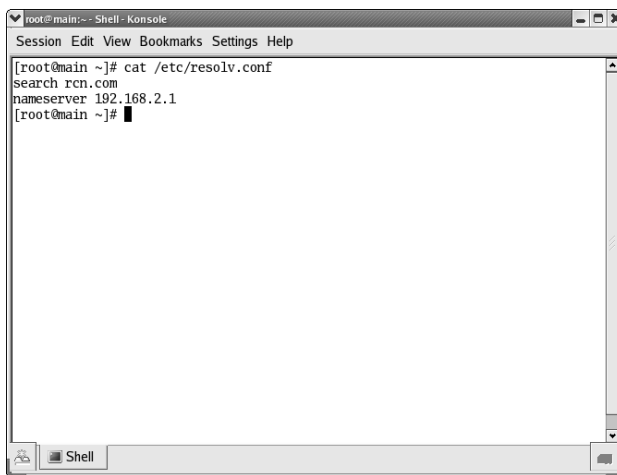
The `/etc/resolv.conf` file provides information about name servers employed to resolve hostnames. Figure 11-4 shows a typical `resolv.conf` file listing.

A terminal window titled "root@main:~ - Shell - Konsole" with a menu bar (Session, Edit, View, Bookmarks, Settings, Help). The terminal shows the command `cat /etc/hosts` and its output. The output lists several IP addresses and their corresponding hostnames, including `127.0.0.1 localhost.localdomain localhost`, `192.168.2.4 laptop.tactechology.com laptop`, `192.168.2.5 main.tactechology.com main`, `192.168.2.7 ibmwinxp.tactechology.com ibmwinxp`, and `192.168.2.8 elas.tactechology.com elas`. The prompt `[root@main ~]#` is visible at the end of the output.

```
root@main:~ - Shell - Konsole
Session Edit View Bookmarks Settings Help

[root@main ~]# cat /etc/hosts
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1      localhost.localdomain  localhost
192.168.2.4    laptop.tactechology.com  laptop
192.168.2.5    main.tactechology.com   main
192.168.2.7    ibmwinxp.tactechology.com ibmwinxp
192.168.2.8    elas.tactechology.com   elas
[root@main ~]#
```

Figure 11-3 The `/etc/hosts` file contains a listing of the computers on a network.

A terminal window titled "root@main:~ - Shell - Konsole" with a menu bar (Session, Edit, View, Bookmarks, Settings, Help). The terminal shows the command `cat /etc/resolv.conf` and its output. The output shows the domain `search rcn.com` and the nameserver `nameserver 192.168.2.1`. The prompt `[root@main ~]#` is visible at the end of the output.

```
root@main:~ - Shell - Konsole
Session Edit View Bookmarks Settings Help

[root@main ~]# cat /etc/resolv.conf
search rcn.com
nameserver 192.168.2.1
[root@main ~]#
```

Figure 11-4 The `/etc/resolv.conf` file contains a listing of the domain and name servers on the network.

CROSS-REFERENCE Chapter 14 discusses Domain Name Servers (DNS).

The `/etc/sysconfig/network` file contains two lines, as follows:

```
NETWORKING=yes
HOSTNAME=(host and domain name of your system)
```

The first line enables networking for your system. The second line displays the hostname of your system and the name of the domain to which it belongs.

Understanding Subnetting

You have learned how easy it is to build an internal network, but now you need to learn how to connect to the outside world. A few more steps accomplish outside connection, including configuring a router, obtaining an IP address, and actually making the connection. You begin with obtaining an IP address and subnetting.

Earlier in this chapter you saw that IP addresses used on the Internet are assigned by the InterNIC. Now you will take a closer look at the makeup of IP addresses and how they can be extended through subnetting.

IP numbers are not assigned to hosts; they are assigned to network interfaces on hosts. Even though many computers on an IP network have a single network interface and a single IP number, a single computer can have more than one network interface. In this case, each interface would have its own IP number.

TIP It is also possible to assign more than one IP address to a single NIC. This is accomplished by using the `ifconfig` and `route` commands. To add another IP address, 192.168.1.4, to `eth0` issue these commands:

```
ifconfig eth0:1 192.168.1.4
route add -host 192.168.1.4 dev eth0
```

The first command binds the IP address to the virtual interface `eth0:1`, and the second command adds a route for the address to the actual device `eth0`.

Another method for adding a second IP address to a single NIC is to create an alias file. The configuration file for device `eth0` is located in `/etc/sysconfig/network-scripts/ifcfg-eth0`. Copy this file to another file called `/ifcfg-eth0:1` in the same directory. Open the newly copied file and change the line that reads:

```
DEVICE=eth0
```

to read:

```
DEVICE=eth0:1
```

NOTE If you create an alias for your NIC, you cannot use DHCP to obtain your IP information. You must assign static IP addresses to the devices.

Even though this is true, most people refer to host addresses when referring to an IP number. Just remember, this is simply shorthand for the IP number of this particular interface on this host. Many devices on the Internet have only a single interface and thus a single IP number.

In the current (IPv4) implementation, IP numbers consist of 4 (8-bit) bytes for a total of 32 bits of available information. This system results in large numbers, even when they are represented in decimal notation. To make them easier to read and organize, they are written in what is called dotted quad format. The numbers you saw earlier in this chapter were expressed in this format, such as the internal network IP address 192.168.1.1. Each of the four groups of numbers can range from 0 to 255. The following shows the IP number in binary notation with its decimal equivalent. If the bit is set to 1 it is counted, and if set to zero it is not counted.

$$\begin{array}{cccccccc}
 1 & + & 1 & + & 1 & + & 1 & + & 1 & + & 1 & + & 1 & + & 1 \\
 128 & + & 64 & + & 32 & + & 16 & + & 8 & + & 4 & + & 2 & + & 1 = 255
 \end{array}$$

The binary notation for 192.168.1.1 is:

11000000.10101000.00000001.00000001

The dotted quad notation from this binary is:

$$(128+64) \cdot (128+32+8) \cdot (1) \cdot (1) = 192.168.1.1$$

The leftmost bits of the IP number of a host identify the network on which the host resides; the remaining bits of the IP number identify the network interface. Exactly how many bits are used by the network ID and how many are available to identify interfaces on that network is determined by the network class. Earlier you learned that there are three classes of networks, and you saw how they are composed in Table 11-1.

Class A IP network numbers use the left quad to identify the network, leaving three quads to identify host interfaces on that network. Class A addresses always have the farthest left bit of the farthest left byte a zero, so there is a maximum of 128 class A network numbers available, with each one containing up to 33,554,430 possible interfaces.

The network numbers 0.0.0.0, known as the default route, and 127.0.0.0, the loopback network, have special meanings and cannot be used to identify networks. You saw the loopback interface when you set up your internal network. You'll look at the default route when you set up your connection to the Internet. So if you take these two network numbers out, there are only 126 available class A network numbers.

Class B IP network numbers use the two left dotted quads to identify the network, leaving two dotted quads to identify host interfaces. Class B addresses

always have the farthest left bits of the left byte set to 10. This leaves 14 bits left to specify the network address giving 32,767 available B class networks. Class B networks have a range of 128 to 191 for the first of the dotted quads, with each network containing up to 32,766 possible interfaces.

Class C IP network numbers use the left three quads to identify the network, leaving the right quad to identify host interfaces. Class C addresses always start with the farthest left three bits set to 1 1 0 or a range of 192 to 255 for the farthest left dotted quad. This means that there are 4,194,303 available Class C network numbers, each containing 254 interfaces.

IP addresses are also set aside for internal networks, as you saw in Table 11-2.

Interpreting IP Numbers

IP numbers can have three possible meanings. The first of these is an address of a network, which is the number representing all the devices that are physically connected to each other. The second is the broadcast address of the network, which is the address that enables all devices on the network to be contacted. Finally, the last meaning is an actual interface address. Look at a Class C network for an example. For a Class C network:

- 192.168.3.0 is a Class C network number.
- 192.168.3.42 is a host address on this network.
- 192.168.3.255 is the network broadcast address.

When you set up your Ethernet device, `eth0`, you used the `ifconfig` utility to pass some information that was written to the `ifcfg-eth0` file. One of these parameters was the network mask. The network mask is more properly called the *subnet mask*. However, it is generally referred to as the *network mask*, or *subnet mask*. The determining factor in subnetting is the network mask and how it is understood on a local network segment. In setting up your network card, you used a net mask of 255.255.255.0. In this case, all the network bits were set to one and the host bits were set to zero. This is the standard format for all network masks. Table 11-2 shows the network masks for the three classes of networks.

You should remember two important things about the network mask. The network mask affects only the interpretation of IP numbers on the same network segment, and the network mask is not an IP number; it is used to modify the way IP numbers are interpreted by the network.

NOTE The network mask affects only the interpretation of IP numbers on the same network segment.

A subnet enables you to use one IP address and split it up so that it can be used on several physically connected local networks. This is a tremendous advantage, as the number of IP numbers available is rapidly diminishing. You can have multiple subnetted networks connected to the outside world with just one IP address. By splitting the IP address, it can be used on sites that need multiple connections; splitting the address eliminates the problems of high traffic and difficult manageability.

The other advantages to subnetting are that different network topologies can exist on different network segments within the same organization, and overall network traffic is reduced. Subnetting also enables increased security by separating traffic into local networks. There is a limit to the number of subnets that can be created based on the number of times a given number can be divided. Tables 11-3, 11-5, and 11-6 show the possible numbers of subnets and hosts that can exist.

Before You Subnet Your Network

Before you can subnet your network, you need to make some choices and gather some information.

NOTE The information in this section explains the concepts of subnetting a network and does not give actual details about the procedure to follow should you decide to subnet your network.

First, you need to decide the number of hosts on each of your subnets so that you can determine how many IP addresses you need. Earlier in this chapter you set up an Ethernet interface using the reserved internal Class C network number 192.168.1.0. You will continue to use this number for the subnetting example.

Every IP network has two addresses that cannot be used — the network IP number itself and the broadcast address. Whenever you subnetwork the IP network you are creating additional addresses that are unusable. For each subnet you create, two addresses are unusable, the subnet's network IP address and its broadcast address. Every time you subnet you are creating these two unusable addresses, so the more subnets you have, the more IP addresses you lose. The point is that you don't subnet your network more than necessary.

TIP Don't subnet your network more than necessary.

Next, you need to determine the subnetwork mask and network numbers. The network mask for an IP network without subnets is simply a dotted quad that has all the network bits of the network number set to 1 and all the host bits set to 0.

So, for the three classes of IP networks, the standard network masks are shown in Table 11-2.

Subnetting takes one or more of the available host bits and makes them appear as network bits to the local interfaces. If you wanted to divide your Class C network into two subnetworks, you would change the first host bit to one, and you would get a net mask of 11111111.11111111.11111111.10000000, or 255.255.255.128. This would give you 126 possible IP numbers for each of your subnets. Remember that you lose two IP addresses for each subnet. If you want to have four subnetworks, you need to change the first two host bits to ones, and this would give you a net mask of 255.255.255.192. You would have 62 IP addresses available on each subnetwork. Table 11-3 shows the subnets, the subnet masks, and the available hosts for your Class C network.

Now all you need to do is assign the appropriate numbers for the network, the broadcast address, and the IP addresses for each of the interfaces and you're nearly done. Table 11-4 shows these numbers for subnetting your Class C network into two subnets.

To create subnets for Class A and B networks, you follow the same procedure as that shown for Class C networks. Table 11-5 shows the subnets for a Class A network, and Table 11-6 shows the subnets for a Class B network.

Table 11-3 Class C Subnets and Subnet Masks

NUMBER OF BITS	NUMBER OF SUBNETS	SUBNET MASK	NUMBER OF HOSTS
1*	2	255.255.255.128	126
2	4	255.255.255.192	62
3	8	255.255.255.224	30
4	16	255.255.255.240	14
5	32	255.255.255.248	6
6	64	255.255.255.252	2

Table 11-4 Creating Two Subnets for a Class C Network Address

NETWORK	NET MASK	BROADCAST	FIRST IP	LAST IP
192.168.1.0	255.255.255.128	192.168.1.127	192.168.1.1	192.168.1.126
192.168.1.128	255.255.255.128	192.168.1.255	192.168.1.129	192.168.1.254

Table 11-5 Class A Subnets and Subnet Masks

NUMBER OF BITS	NUMBER OF SUBNETS	SUBNET MASK	NUMBER OF HOSTS
2	2	255.192.0.0	4194302
3	6	255.224.0.0	2097150
4	14	255.240.0.0	1048574
5	30	255.248.0.0	524286
6	62	255.252.0.0	262142
7	126	255.254.0.0	131070
8	254	255.255.0.0	65534
9	510	255.255.128.0	32766
10	1022	255.255.192.0	16382
11	2046	255.255.224.0	8190
12	4094	255.255.240.0	4094
13	8190	255.255.248.0	2046
14	16382	255.255.252.0	1022
15	32766	255.255.254.0	510
16	65534	255.255.255.0	254
17	131070	255.255.255.128	126
18	262142	255.255.255.192	62
19	524286	255.255.255.224	30
20	1048574	255.255.255.240	14
21	2097150	255.255.255.248	6
22	4194302	255.255.255.252	2

Table 11-6 Class B Subnets and Subnet Masks

NUMBER OF BITS	NUMBER OF SUBNETS	SUBNET MASK	NUMBER OF HOSTS
2	2	255.255.192.0	16382
3	6	255.255.224.0	8190
4	14	255.255.240.0	4094

(continued)

Table 11-6 (continued)

NUMBER OF BITS	NUMBER OF SUBNETS	SUBNET MASK	NUMBER OF HOSTS
5	30	255.255.248.0	2046
6	62	255.255.252.0	1022
7	126	255.255.254.0	510
8	254	255.255.255.0	254
9	510	255.255.255.128	126
10	1022	255.255.255.192	62
11	2046	255.255.255.224	30
12	4094	255.255.255.240	14
13	8190	255.255.255.248	6
14	16382	255.255.255.252	2

Classless InterDomain Routing

Classless InterDomain Routing (CIDR) was invented several years ago to keep the Internet from running out of IP addresses. The class system of allocating IP addresses can be very wasteful. Anyone who could reasonably show a need for more than 254 host addresses was given a Class B address block of 65,533 host addresses. Even more wasteful was allocating companies and organizations Class A address blocks, which contain over 16 million host addresses! Only a tiny percentage of the allocated Class A and Class B address space has ever been actually assigned to a host computer on the Internet.

People realized that addresses could be conserved if the class system was eliminated. By accurately allocating only the amount of address space that was actually needed, the address space crisis could be avoided for many years. This solution was first proposed in 1992 as a scheme called *supernetting*. Under supernetting, the class subnet masks are extended so that a network address and subnet mask could, for example, specify multiple Class C subnets with one address. For example, if you needed about a thousand addresses, you could supernet four Class C networks together:

```

192.60.128.0 (11000000.00111100.10000000.00000000) Class C subnet address
192.60.129.0 (11000000.00111100.10000001.00000000) Class C subnet address
192.60.130.0 (11000000.00111100.10000010.00000000) Class C subnet address
192.60.131.0 (11000000.00111100.10000011.00000000) Class C subnet address

```

```
-----
192.60.128.0  (11000000.00111100.10000000.00000000)  Supernetted Subnet address
255.255.252.0 (11111111.11111111.11111100.00000000)  Subnet Mask
192.60.131.255 (11000000.00111100.10000011.11111111)  Broadcast address
```

In this example, the subnet 192.60.128.0 includes all the addresses from 192.60.128.0 to 192.60.131.255. As you can see in the binary representation of the subnet mask, the network portion of the address is 22 bits long, and the host portion is 10 bits long.

Under CIDR, the subnet mask notation is reduced to simplified shorthand. Instead of spelling out the bits of the subnet mask, the number of 1 bits that start the mask are simply listed. In the example, instead of writing the address and subnet mask as

```
192.60.128.0, Subnet Mask 255.255.252.0
```

the network address is written simply as

```
192.60.128.0/22
```

This address indicates the starting address of the network, and number of 1 bits (22) in the network portion of the address. If you look at the subnet mask in binary, you can easily see how this notation works.

```
(11111111.11111111.11111100.00000000)
```

The use of a CIDR-notated address is the same as for a class address. Class addresses can easily be written in CIDR notation (Class A = /8, Class B = /16, and Class C = /24).

It is currently almost impossible for you, as an individual or company, to be allocated your own IP address blocks. You will simply be told to get them from your ISP. The reason for this is the ever-growing size of the Internet routing table. Just five years ago, there were less than 5,000 network routes in the entire Internet. Today, there are over 100,000. Using CIDR, the biggest ISPs are allocated large chunks of address space, usually with a subnet mask of /19 or even smaller. The ISP's customers, often other, smaller ISPs, are then allocated networks from the big ISP's pool. That way, all the big ISP's customers, and their customers, are accessible via one network route on the Internet.

CIDR will probably keep the Internet happily in IP addresses for the next few years at least. After that, IPv6, with 128-bit addresses, will be needed. Under IPv6, even careless address allocation would comfortably enable a billion unique IP addresses for every person on earth! The complete details of CIDR are documented in RFC1519, which was released in September 1993.

NOTE Requests for Comment (RFCs) are documents containing information about computer networking and many areas of the Internet. If you want to learn more about RFCs, check out <ftp://ftp.rfc-editor.org/in-notes/rfc2555.txt>.

Working with Gateways and Routers

You have successfully created two subnets from your Class C network, but the individual network segments cannot communicate with each other yet. You still have to configure a path for them; you do this by using a router. Earlier in this chapter, you learned that a router is necessary for separate networks to communicate with each other. You also learned that each network must be connected to a router in order for this communication to take place. This router connected to each network is called its *gateway*.

In Linux, you can use a computer with two network interfaces to route between two or more subnets. To be able to do this you need to make sure that you enable IP forwarding. All current Linux distributions have IP forwarding compiled as a module, so all you need to do is make sure the module is loaded. You can check this by entering the following query at a command prompt:

```
cat /proc/sys/net/ipv4/ip_forward
```

If forwarding is enabled, the number 1 is displayed; if forwarding is not enabled, the number 0 is displayed.

To enable IP forwarding if it is not already enabled, type the following command:

```
echo "1" > /proc/sys/net/ipv4/ip_forward
```

Continue your setup using the two subnets you previously created with the information in Table 11-4.

Assume that a computer running Linux is acting as a router for your network. It has two network interfaces to the local LANs using the lowest available IP address in each subnetwork on its interface to that network. The network interfaces would be configured as shown in Table 11-7.

Table 11-7 Network Interface Configuration

INTERFACE	IP ADDRESS	NET MASK
Eth0	192.168.1.1	255.255.255.128
Eth1	192.168.1.129	255.255.255.128

The network routing the system would use is shown in Table 11-8.

Table 11-8 Network Routing Configuration

DESTINATION	GATEWAY	MASK	INTERFACE
192.168.1.0	192.168.1.1	255.255.255.128	eth0
192.168.1.128	192.168.1.129	255.255.255.128	eth1

You're nearly finished now, just one more step. Each computer on the subnet has to show the IP address for the interface that is its gateway to the other network. The computers on the first subnet, the 192.168.1.0 network, would have the gateway 192.168.1.1. Remember that you used the first IP address on this network for the gateway computer. The computers on the second subnet, 192.168.1.128, would use 192.168.1.129 as the gateway address. You can add this information using the route command as follows:

```
route add -net 192.168.1.0
```

and then type

```
route add default gw 192.168.1.129
```

This command sets up the route for local (internal) routing and the external route for your first subnet. You need to repeat the previous commands, substituting the appropriate numbers for the second subnet and any additional subnets. You have now successfully set up two subnets and established communication between them. Next, you'll look at automatic network address assignments.

Configuring Dynamic Host Configuration Protocol

So far, you have learned to configure a network card and assign it an IP address, subnet mask, broadcast address, and gateway. Using Dynamic Host Configuration Protocol (DHCP), you can have an IP address and the other information automatically assigned to the hosts connected to your network. This method is quite efficient and convenient for large networks with many hosts, because the process of manually configuring each host is quite time-consuming. By using DHCP, you can ensure that every host on your network has a valid IP address, subnet mask, broadcast address, and gateway, with minimum effort on your part.

While not absolutely necessary, you should have a DHCP server configured for each of your subnets. Each host on the subnet needs to be configured as a DHCP client. You may also need to configure the server that connects to your ISP as a DHCP client if your ISP dynamically assigns your IP address.

Setting Up the Server

The program that runs on the server is `dhcpd` and is included as an RPM on the Fedora Core and Red Hat Enterprise Linux installation CDs. You can install it using the Package Management tool by following these instructions.

1. On Enterprise Linux choose Applications ⇄ System Settings ⇄ Add/Remove Applications from the top panel. On Fedora Core 4 choose Desktop ⇄ System Settings ⇄ Add/Remove Applications. The screen shown in Figure 11-5 appears.
2. Scroll down the list until you see a listing for Network Servers.
3. Click the Details link for Network Servers. The screen shown in Figure 11-6 appears.
4. Click Close; then click Update, and finally click Continue.
5. Insert the requested numbered installation CD when prompted and click OK.
6. After the package is installed, click Close to exit the Package Management tool.

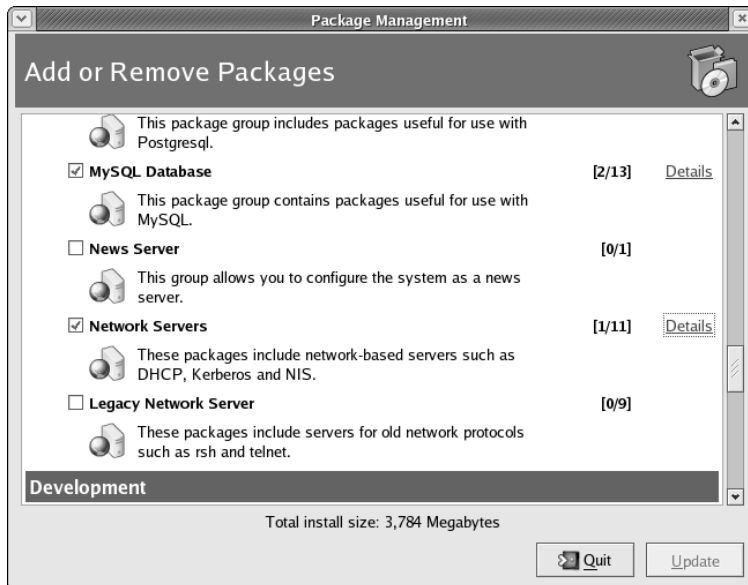


Figure 11-5 Installing a package using the Package Management tool.

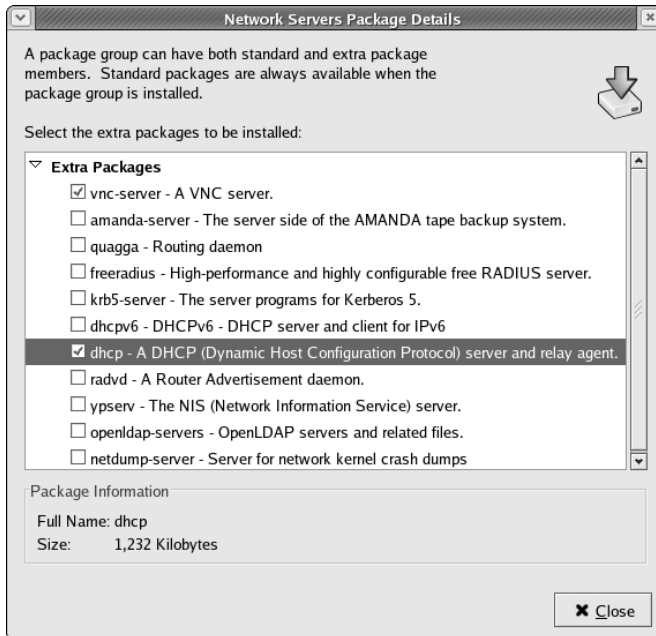


Figure 11-6 Selecting the DHCP package for installation.

In Fedora Core and Red Hat Enterprise Linux the DHCP server is controlled by the text file `/etc/dhcpd.conf`. Listing 11-1 shows the configuration file for my system. Comment lines begin with a `#` sign.

```
#(The amount of time in seconds that the host can keep the IP address.)
default-lease-time 36000;
#(The maximum time the host can keep the IP address.)
#domain name
max-lease-time 100000;
# (The domain of the DHCP server.)
#nameserver
option domain-name "tactechology.com";
option domain-name-servers 192.168.1.1;
#gateway/routers, can pass more than one:
option routers 1.2.3.4,1.2.3.5;
option routers 192.168.1.1; (IP address of routers.)
#netmask (The subnet mask of the network.)
option subnet-mask 255.255.255.0;
#broadcast address (The broadcast address of the network.)
option broadcast-address 192.168.1.255;
#specify the subnet number gets assigned in
subnet 192.168.1.0 netmask 255.255.255.0
#define which addresses can be used/assigned
range 192.168.1.1 192.168.1.126;
```

Listing 11-1 The `dhcpd.conf` file.

A sample file is created when you install the `dhcpd` package that you can use as a guide. You can modify it using a text editor. Be sure to use the proper addresses for your network.

NOTE You need to restart the DHCP server whenever you make changes to the `/etc/dhcpd.conf` file.

To start the server, run the command `service dhcpd start`. To ensure that the `dhcpd` program runs whenever the system is booted, you should run the command `chkconfig --level 35 dhcpd on`.

Configuring the DHCP Client

First, you need to be sure that your NIC is properly configured and recognized by your system. After that, it is easy to tell your system to use DHCP to obtain its IP information. Follow these steps.

1. Using your favorite text editor, open the `/etc/sysconfig/network-scripts/ifcfg-eth0` file.
2. Find the line `bootproto=static`.
3. Change `static` to **`dhcp`**.
4. Save your changes.
5. Restart the network by issuing the command **`service network restart`**, and your system will receive its IP information from the DHCP server.

Configuring the Network Using the Network Configuration Tool

Fedora Core and Red Hat Enterprise Linux provide a graphical network configuration tool that you can use to configure network interface devices installed in your system. With this tool, you can configure Crypto IP Encapsulation (CPIE), Ethernet, Integrated Services Digital Network (ISDN), modem, token ring, wireless, and xDSL. The `x` refers to different versions of Digital Subscriber Loop (DSL) devices. In this chapter, I cover the most common types of devices: Ethernet, modem, and wireless.

You can access the Network Configuration tool by using the Applications menu from the GNOME desktop. To start the Network Configuration tool in Enterprise Linux choose Applications ⇨ System Settings ⇨ Network. In Fedora Core 4 choose Desktop ⇨ System Settings ⇨ Network. The Network Configuration window, shown in Figure 11-7, appears.

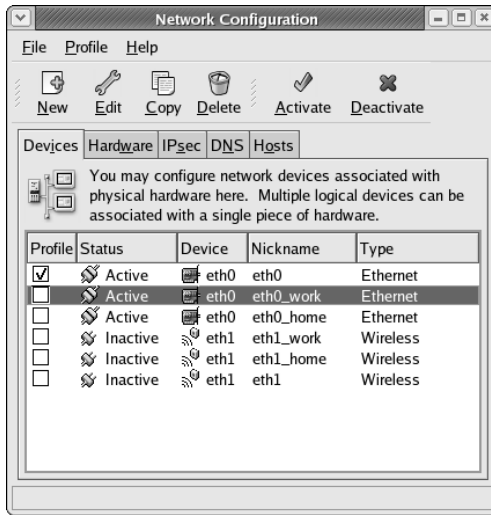


Figure 11-7 The Network Configuration tool main window.

The main Network Configuration tool window (shown in Figure 11-7) has five tabbed pages and opens to the Devices tab by default. Read on for a more detailed look at the five tabbed pages.

- **Devices** — This tab shows the network devices that are installed and configured on your PC. Network devices are associated with the actual physical hardware in the PC.

NOTE If you have a supported NIC installed on your system during installation of Red Hat Enterprise Linux, your NIC should already be listed in the Network Configuration tool. Click the Hardware tab to see information about the device. Figure 11-7 shows an Ethernet NIC with a wireless NIC already installed.

- **Hardware** — This tab shows the actual physical hardware installed in your PC.
- **IPSec** — This tab is where you can configure IPSec tunnels used for secure communications.
- **DNS** — This tab shows the system hostname, domain, and name servers used for DNS lookups. You can configure this information here.
- **Hosts** — This tab shows the PC hostname to static IP address mapping.

Adding an Ethernet Device

With the Network Configuration tool, you can easily add and configure your Ethernet device. To add an Ethernet device, do the following:

1. Click the New button from the toolbar of the Network Configuration tool main window. The Select Device Type window appears, as shown in Figure 11-8.
2. Choose Ethernet Connection from the Device Type list, and then click Forward to go to the Select Ethernet Device list.
3. If your NIC is shown in the Select Ethernet Device list, select it and then click Forward to go to the Configure Network Settings window. (See Figure 11-10.)
4. If your NIC is not listed, choose Other Ethernet Card and then click Forward to open the Select Ethernet Adapter window, as shown in Figure 11-9.
5. Select your card from the Adapter drop-down list.
6. Choose the device name from the Device drop-down list.

You should choose `eth0` for the first device, `eth1` for the second, `eth2` for the third, and so on. You can also enter the system resources that the adapter will use, if desired. Usually, this is not necessary because the OS automatically assigns resources to devices. But in the event that you need to control the resource assignments manually because of other hardware that you are using, you are able to do so.

7. Click Forward to continue to the Configure Network Settings window, as shown in Figure 11-10.

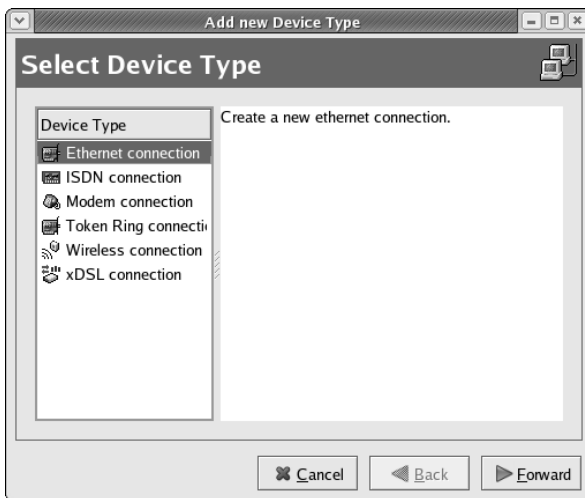


Figure 11-8 Select your network device here.

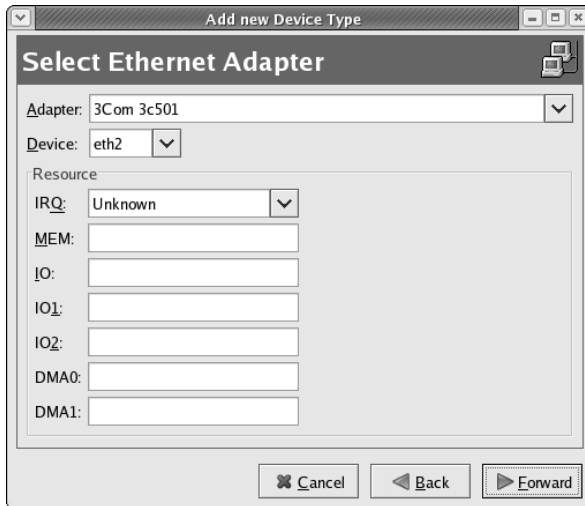


Figure 11-9 The Select Ethernet Adapter window.

8. Choose whether you want to use DHCP to obtain your IP address automatically or whether you want to enter a static IP address.

Make your choice by selecting the appropriate radio button. If you choose to set your address statically, you must enter the IP address, the network mask, and the address of the default gateway. You can also enter a hostname for your PC, if you desire.

9. Click Forward.

You see a listing of your selected information.

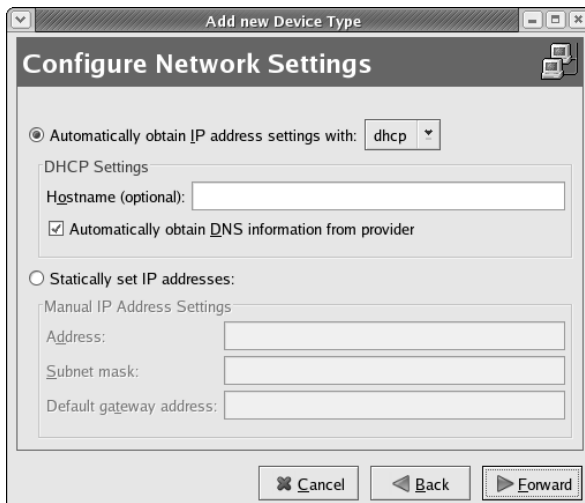


Figure 11-10 The Configure Network Settings window.

10. (Optional) If you want to make changes, click Back to return to the desired window and make changes. If you are satisfied with your choices, click Apply to create the device.

After clicking Apply, the device is created and appears in the device list. Although the device has been configured and added to the list of devices, it is inactive, as you can see from the device listing. By default, the device will start at boot time, but you can activate it immediately by highlighting it and then clicking the Activate button from the menu bar at the top of the window.

11. Choose File ⇨ Save from the menu to save your changes.

Adding a Wireless NIC

With the Network Configuration tool, you can easily add and configure your wireless NIC. To add a wireless NIC, do the following:

1. Click the New button from the toolbar of the Network Configuration tool main window. (Refer to Figure 11-7.)
The Select Device Type window appears. (Refer to Figure 11-8.)
2. Choose Wireless Connection from the Device Type list and then click Forward to go to the Select Wireless Device list.
3. If your NIC is shown in the Select Wireless Device list, select it and click Forward to go to the Configure Wireless Connection dialog box, as shown in Figure 11-11. If your NIC is not listed, go to Step 6.

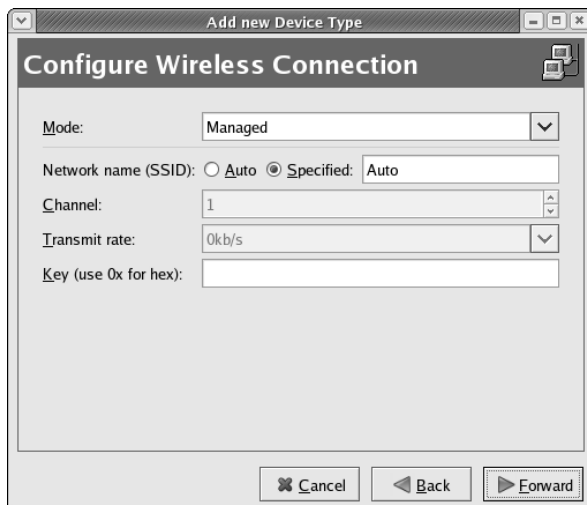


Figure 11-11 The Configure Wireless Connection dialog box.

4. In the Configure Wireless Connection dialog box, enter the appropriate information for your wireless connection as follows:
 - **Mode** — From the drop-down list, choose from:
 - Auto* to have the system automatically determine the connection type
 - Managed* to set your configuration to connect to a wireless access point
 - Ad-hoc* if you will be connecting directly to other wireless NICs
 - **Network Name (SSID)** — Either select the Specified radio button and enter the name of your network here, or select the Auto radio button to have the system determine the name.
 - **Channel** — Enter the channel that your wireless network uses.
 - **Transmit Rate** — Choose Auto or a specific transmission rate for your network.
 - **Key** — If your network uses Wired Equivalent Privacy (WEP), enter the appropriate encryption key here.

5. After you enter your network information, click Forward to go to the Configure Network Settings dialog box and continue with Step 10. (Refer to Figure 11-10.)
6. If your NIC is not listed, choose Other Wireless Card and then click Forward to open the Select Ethernet Adapter window. (Refer to Figure 11-9.)
7. Select your card from the drop-down list in the Adapter field.
8. After choosing your card, choose the device name from the drop-down list in the Device field.

You should choose `eth0` for the first device, `eth1` for the second, `eth2` for the third, and so on. You can also enter the system resources that the adapter will use, if desired. Usually, this is not necessary because the OS automatically assigns resources to devices. But in the event that you need to manually control the resource assignments because of other hardware you are using, you are able to do so.

9. Click Forward to continue to the Configure Wireless Connection dialog box. (Refer to Figure 11-11; see Step 4 for details on this dialog box.) After entering your information, click Forward to go to the Configure Network Settings window.
10. In the Configure Network Settings window, you can choose whether you want to use DHCP to obtain your IP address automatically or whether you want to enter a static IP address.

Make your choice by selecting the appropriate radio button. If you choose to set your address statically, you must enter the IP address, the network mask, and the address of the default gateway. You can also enter a hostname for your PC, if you desire.

11. After you make the appropriate entries, click Forward.

You see a listing of your selected information.

12. If you want to make changes, click Back to return to the desired window and make changes. If you are satisfied with your choices, click Apply to create the device.

After clicking Apply, the device is created and appears in the Device list.

Although the device has been configured and added to the list of devices, it is inactive, as you can see from the device listing. By default, the device starts at boot time, but you can activate it immediately by highlighting it and then clicking the Activate button from the menu bar at the top of the window.

13. Choose File ⇨ Save from the menu to save your changes.

Adding a Modem Connection

With the Network Configuration tool, you can easily add and configure your modem. To add a modem, do the following:

1. Click the New button on the toolbar.
2. Choose Modem Connection from the Device Type list, and then click Forward.

The Configuration tool searches your system to try to detect a modem.

NOTE If you have a modem installed in your system during installation your modem should already be listed in the Network Configuration tool. Click the Hardware tab to see information about the device.

If you have a modem in your hardware list, the Configuration tool uses that modem and opens the Select Modem window, as shown in Figure 11-12, with values appropriate for the modem. If no modem is found, a message appears, stating that no modem was found and prompting you to click OK. After you click OK, the Select Modem dialog box appears, but the values might not be correct for the modem that you have installed.

3. If your modem was successfully found, you can accept the default values for modem device, baud rate, flow control, and modem volume; otherwise, enter the values appropriate for your modem.
4. When you are satisfied with the settings, click Forward to go to the Select Provider window, as shown in Figure 11-13.

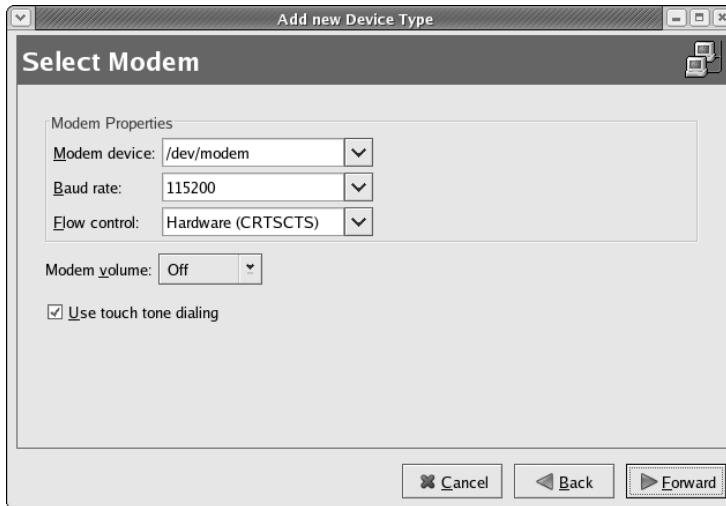


Figure 11-12 The Select Modem dialog box.

5. Here you need to enter the name of your ISP and the telephone number that you dial to connect. Enter the login name and password that were given to you by your ISP. Unless you live in any of the countries listed, you can ignore the country list on the left.
6. Click Forward to go to the IP Settings window, as shown in Figure 11-14. You can probably accept the default setting here to obtain IP addressing information automatically.

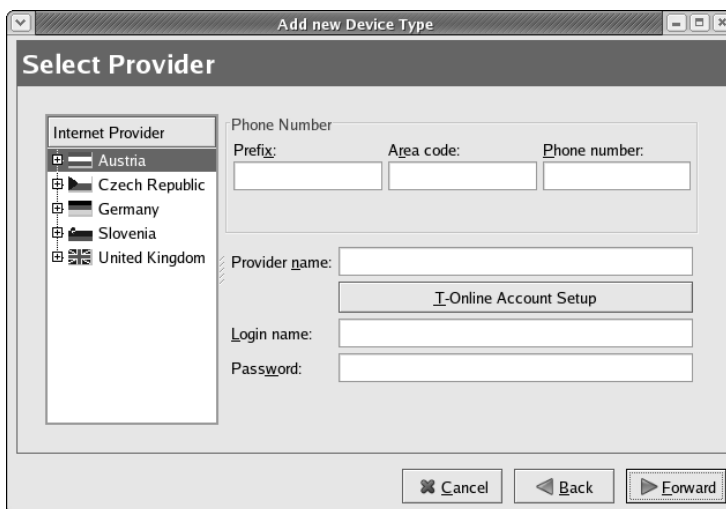


Figure 11-13 The Select Provider dialog box.

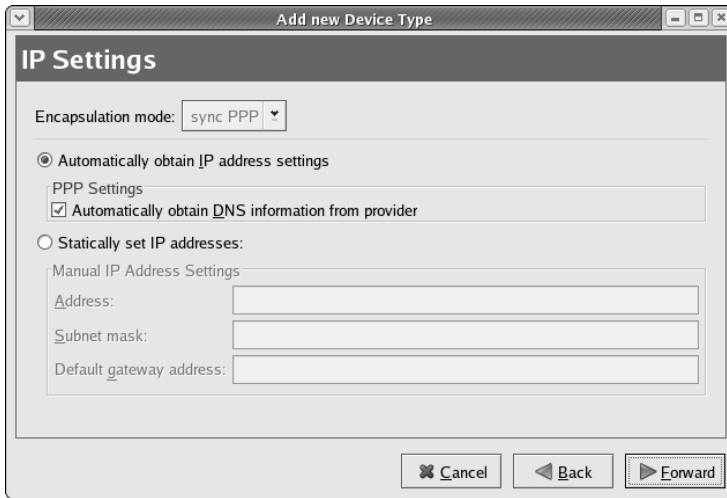


Figure 11-14 The IP Settings dialog box.

7. Click Forward to continue.

You see a listing of your selected information. If you want to make changes, click Back to return to the desired window and make changes. If you are satisfied with your choices, click Apply to create the device.

The device is created and appears in the Device list. Although the device has been configured and added to the list of devices, it is inactive, as you can see from the device listing. By default, the device starts at boot time, but you can activate it immediately by highlighting it and then clicking the Activate button from the menu bar at the top of the window.

8. Choose File ➦ Save from the menu to save your changes.

Editing Your Network Configuration

After you add and configure your network connection device, whether it is a wired NIC, wireless NIC, or modem, you usually don't need to change the configuration. You might need to modify the configuration, though, if you change to a different NIC.

Removing a NIC

Using the Network Configuration tool, you can easily make the necessary changes. Start the Network Configuration tool as follows:

1. In Enterprise Linux choose Applications ⇨ System Settings ⇨ Network.
In Fedora Core 4 choose Desktop ⇨ System Settings ⇨ Network.
2. Click the Hardware tab.
3. Highlight the device that you want to remove, and then click Delete.
4. When finished, choose File ⇨ Save to save your changes.

Changing the NIC Configuration

Using the Network Configuration tool, you can easily make the necessary changes. Start the Network Configuration tool as follows:

1. In Enterprise Linux choose Applications ⇨ System Settings ⇨ Network.
In Fedora Core 4 choose Desktop ⇨ System Settings ⇨ Network.
2. Highlight the device that you want to modify, and then click Edit (on the toolbar).

The Ethernet Device properties dialog box for the device you selected, as shown in Figure 11-15, appears.

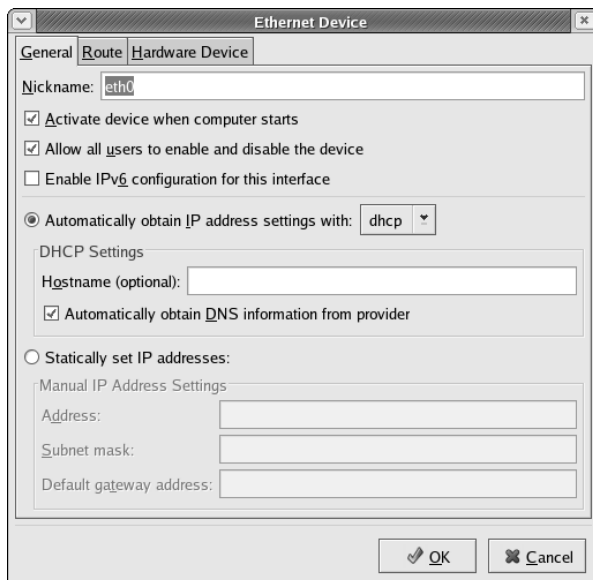


Figure 11-15 The Ethernet Device properties dialog box.

3. The three tabs available from this dialog box are used for the following:
 - **General** — Here you can enter a nickname for the device and choose whether the device is activated when the system starts. You can also choose to allow other users to be able to enable and disable the device. You can choose to obtain IP information automatically by using DHCP, or you can manually enter the IP information for the device.

NOTE In most cases, you can accept the default setting and let the system obtain IP information using DHCP. If you need to use a static IP address, you can usually get the IP information from your system administrator. If you are the system administrator, you should know what IP information to use.

- **Route** — Here you can enter static routes to other networks. You need to enter the network IP number as well as the gateway IP number. In most cases, you don't need to enter any information here if you are using DHCP.
- **Hardware Device** — This tab contains information about the hardware associated with the Ethernet device. You can assign device aliases here if you desire.

NOTE *Device aliases* are virtual devices associated with the same physical hardware, but they can be activated at the same time to have different IP addresses. They are commonly represented as the device name followed by a colon and a number (for example, `eth0:1`). They are useful if you want to have more than one IP address for a system but the system has only one network card.

If you have configured a device, such as `eth0`:

- a. Click the Add button in the Network Administration tool to create an alias for the device.
- b. Select the network device and configure the network settings.

NOTE The alias will appear in the device list with a device name, followed by a colon and the alias number.

4. After you make the changes you desire, click OK to return to the Network Configuration dialog box.
5. Choose File ⇨ Save to write your configuration changes to a file.

Managing DNS Settings

The DNS tab of the Network Configuration tool is where you configure the system's hostname, domain, name servers, and search domain. *Name servers* are used to look up other hosts on the network. To enter or change these settings, do the following:

1. In Enterprise Linux choose Applications ⇨ System Settings ⇨ Network. In Fedora Core 4 choose Desktop ⇨ System Settings ⇨ Network.
2. Click the DNS tab from the Network Configuration dialog box.
3. On the DNS tab, enter the appropriate information for your system.
4. After you finish, choose File ⇨ Save to save your changes.

NOTE The Nameservers section does not configure the system to be a name server. If the DNS server names are retrieved from DHCP (or retrieved from the ISP of a modem connection), do not add primary, secondary, or tertiary DNS servers.

Managing Hosts

On the Hosts tab of the Network Configuration tool, you can add, edit, or remove hosts to or from the `/etc/hosts` file. This file contains IP addresses and their corresponding hostnames. When your system tries to resolve a hostname to an IP address or determine the hostname for an IP address, it refers to the `/etc/hosts` file before using the name servers (if you are using the default Fedora Core or Red Hat Enterprise Linux configuration). If the IP address is listed in the `/etc/hosts` file, the name servers are not used. If your network contains computers whose IP addresses are not listed in DNS, it is recommended that you add them to the `/etc/hosts` file.

1. In Enterprise Linux choose Applications ⇨ System Settings ⇨ Network. In Fedora Core 4 choose Desktop ⇨ System Settings ⇨ Network.
2. Click the Hosts tab from the Network Configuration dialog box.
The Hosts tab that appears shows the hostname to static IP address mappings, if any.
3. Click New from the toolbar to open the Add/Edit Hosts Entry dialog box.
4. Enter the hostname and its IP address. If there is an alias for the hostname, enter it as well.

5. Click OK to add the entry to the list.
6. Choose File ⇨ Save to save your changes.

WARNING Never remove the Localhost entry from the Hosts section, or your network will not work properly.

Working with Profiles

Multiple logical network devices can be created for each physical hardware device. For example, if you have one Ethernet card in your system (`eth0`), you can create logical network devices with different nicknames and different configuration options, all associated with `eth0`. These logical network devices are different from device aliases. Logical network devices associated with the same physical device must exist in different profiles and cannot be activated simultaneously. Device aliases are also associated with the same physical hardware device, but device aliases associated with the same physical hardware can be activated at the same time.

Profiles can be used to create multiple configuration sets for different networks. A configuration set can include logical devices as well as hosts and DNS settings. After configuring the profiles, you can use the Network Administration tool to switch back and forth between them.

By default, there is one profile called Common. To create a new profile, do the following:

1. In Enterprise Linux choose Applications ⇨ System Settings ⇨ Network. In Fedora Core 4 choose Desktop ⇨ System Settings ⇨ Network.
2. Choose Profile ⇨ New from the menu.
3. Enter a unique name for the profile.
4. After creating a new profile, if all the devices are not listed for all the profiles, add them by clicking the Add button. If a device already exists for the physical device, use the Copy button to copy the existing device.

In the list of devices is a column of check boxes labeled Profile. For each profile, you can check (mark) or uncheck (clear) devices. Only the checked devices are included for the currently selected profile.

NOTE A profile cannot be activated at boot time. Only the devices in the Common profile, which are set to activate at boot time, are activated at boot time. After the system has booted, execute the following command to enable a profile (replacing `profilename` with the name of the profile):

```
redhat-config-network-cmd --profile profilename
```

Configuring IP Masquerading

So far, you have configured an internal network consisting of two subnets, and you configured a router for connectivity between the networks. Assuming that you made the connection to the Internet through your router, you need to make only a few configuration changes and every computer on your network can connect to the Internet through your one connection. This type of connection provides a service known as Network Address Translation, or NAT. To use a single Internet connection for multiple computers, you need to use *IP masquerading*. IP masquerading enables you to connect a TCP/IP network to the outside world using a single server and a single IP address.

Current Red Hat distributions make IP masquerading available as a module, so you need only load the module and enable the appropriate configuration. You already enabled IP forwarding when you configured your router, so you need to consider only one more item, a utility called *iptables*, which sets up a simple packet-filtering firewall.

CROSS-REFERENCE To learn about securing your Red Hat system for the Internet, see Chapter 31.

The *iptables* utility gives you enough protection for now. To set up masquerading, type the following commands:

```
iptables -P forward DENY
iptables -A forward -s 192.168.0.0/24 -j MASQ -d 0.0.0.0/0
```

Those commands are all you need to enable the firewall rules and to start masquerading.

Of course you want IP masquerading enabled whenever you boot the computer, so it's a good idea to make a script file that enables IP forwarding as well as the *iptables* utility. This file would ensure that forwarding and masquerading start each time the machine boots.

Be sure to include the command to start IP forwarding (shown earlier in this chapter) as well as the *iptables* commands shown here in your system startup sequence. You can place the appropriate commands in your `/etc/rc.local` file so that they will run when your system starts.

Summary

In this chapter, you learned about the TCP/IP protocol suite and how it works to enable communication across networks. Then you learned how to configure a network interface card by creating and modifying the configuration files

directly, as well as using the graphical Network Configuration tool. You used subnetting to create two internal subnetworks and configured a router so the subnetworks could communicate with each other. You set up a Dynamic Host Configuration Protocol server to assign IP addresses to the hosts on the network. You also enabled forwarding and masquerading so that every computer on your internal network could have Internet access.

The Network File System

IN THIS CHAPTER

- NFS Overview
- Planning an NFS Installation
- Configuring an NFS Server
- Configuring an NFS Client
- Using Automount Services
- Examining NFS Security

Linux Servers are often installed to provide centralized file and print services for networks. This chapter explains how to use the Network File System (NFS) to create a file server. After a short overview of NFS, you learn how to plan an NFS installation, how to configure an NFS server, and how to set up an NFS client. You'll learn how to mount remote file systems automatically, eliminating the need to mount remote file systems manually before you can access it. The final section of the chapter highlights NFS-related security issues.

NFS Overview

NFS is the most common method used to share files across Linux and UNIX networks. It is a distributed file system that enables local access to remote disks and file systems. In a properly designed and carefully implemented NFS installation, NFS's operation is totally transparent to clients using remote file systems. Provided that you have the appropriate network connection, you can access files and directories that are physically located on another system or even in a different city or country using standard Linux commands. No special

procedures, such as using a password, are necessary. NFS is a common and popular file-sharing protocol, so NFS clients are available for many non-UNIX operating systems, including the various Windows versions, MacOS, OS/2, VAX/VMS, and MVS.

Understanding NFS

NFS follows standard client/server architectural principles. The server component of NFS consists of the physical disks that contain the file systems you want to share and several daemons that make these shared file systems visible to and available for use by client systems on the network. When an NFS server is sharing a file system in this manner, it is said to be *exporting a file system*. Similarly, the shared file system is referred to as an *NFS export*. The NFS server daemons provide remote access to the exported file systems, enable file locking over the network, and, optionally, allow the server administrator to set and enforce disk quotas on the NFS exports.

On the client side of the equation, an NFS client simply mounts the exported file systems locally, just as local disks would be mounted. The mounted file system is known colloquially as an *NFS mount*.

The possible uses of NFS are quite varied. NFS is often used to provide diskless clients, such as X terminals or the slave nodes in a cluster, with their entire file system, including the kernel image and other boot files. Another common scheme is to export shared data or project-specific directories from an NFS server and to enable clients to mount these remote file systems anywhere they see fit on the local system.

Perhaps the most common use of NFS is to provide centralized storage for users' home directories. Many sites store users' home directories on a central server and use NFS to mount the home directory when users log in or boot their systems. Usually, the exported directories are mounted as `/home/username` on the local (client) systems, but the export itself can be stored anywhere on the NFS server, for example, `/exports/users/username`. Figure 12-1 illustrates both of these NFS uses.

The network shown in Figure 12-1 shows a server (suppose that its name is `diskbeast`) with two set of NFS exports, user home directories on the file system `/exports/homes` (`/exports/homes/u1`, `/exports/homes/u2`, and so on) and a project directory stored on a separate file system named `/proj`. Figure 12-1 also illustrates a number of client systems (pear, apple, mango, and so forth). Each client system mounts `/home` locally from `diskbeast`. On `diskbeast`, the exported file systems are stored in the `/exports/homes` directory. When a user logs in to a given system, that user's home directory is automatically mounted on `/home/username` on that system. So, for example,

because user `u1` has logged in on `pear`, `/exports/homes/u1` is mounted on `pear`'s file system as `/home/u1`. If `u1` then logs in on `mango`, too (not illustrated in Figure 12-1), `mango` also mounts `/home/u1`. Logging in on two systems this way is potentially dangerous because changes to files in the exported file system made from one login session might adversely affect the other login session. Despite the potential for such unintended consequences, it is also very convenient for such changes to be immediately visible.

Figure 12-1 also shows that three users, `u5`, `u6`, and `u7`, have mounted the project-specific file system, `/proj`, in various locations on their local file systems. Specifically, user `u5` has mounted it as `/work/proj` on `kiwi` (that is, `kiwi:/work/proj` in `host:/mount/dir` form) `u6` as `lime:/projects`, and `u7` as `peach:/home/work`.

NFS can be used in almost any situation requiring transparent local access to remote file systems. In fact, you can use NFS and NIS (Chapter 13 covers NIS in depth) together to create a highly centralized network environment that makes it easier to administer the network, add and delete user accounts, protect and back up key data and file systems, and give users a uniform, consistent view of the network regardless of where they log in.

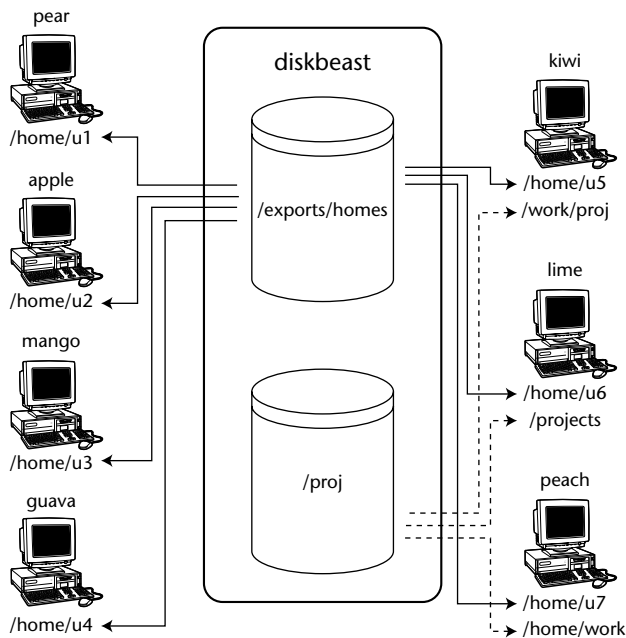


Figure 12-1 Exporting home directories and project-specific file systems.

As you will see in the sections titled “Configuring an NFS Server” and “Configuring an NFS Client,” NFS is easy to set up and maintain and pleasantly flexible. Exports can be mounted read-only or in read-write mode. Permission to mount exported file systems can be limited to a single host or to a group of hosts using either hostnames with the wildcards * and ? or using IP address ranges, or even using NIS groups, which are similar to, but not the same as, standard UNIX user groups. Other options enable strengthening or weakening of certain security options as the situation demands.

What’s New with NFSv4?

NFS version 4, which is the version available in Fedora Core and Red Hat Enterprise Linux, offers significant security and performance enhancements over older versions of the NFS protocol and adds features such as replication (the ability to duplicate a server’s exported file systems on other servers) and migration (the capability to move file systems from one NFS server to another without affecting NFS clients) that NFS has historically lacked. For example, one of the (justified) knocks against NFS has been that it transmits authentication data as clear text. NFSv4 incorporates RPCSEC-GSS (the SecureRPC protocol using the Generic Security Service API) security, which makes it possible to encrypt the data stream transmitted between NFS clients and servers. Another security feature added to NFSv4 is support for access control lists, or ACLs. ACLs build on the traditional Linux UID- and GID-based file and directory access by giving users and administrators the ability to set more finely grained restrictions on who can read, write, and/or execute a given file.

In terms of backward compatibility, NFSv4 isn’t, at least not completely. Specifically, an NFSv4 client might not be able to mount an NFSv2 export. It has been our experience that mounting an NFSv2 export on an NFSv4 client requires the use of the NFS-specific mount option `nfsvers=2`. Going the other direction, mounting an NFSv4 export on an NFSv2 does not require special handling. NFSv4 and NFSv3 interoperability is no problem. See the section titled “Configuring an NFS Client” for more details about interoperability between NFS versions.

In terms of performance enhancements, NFSv4 makes fuller use of client-side caching, which reduces the frequency with which clients must communicate with an NFS server. By decreasing the number of server round trips, overall performance increases. In addition, NFSv4 was specifically designed (or enhanced) to provide reasonable performance over the Internet, even on slow, low-bandwidth connections or in high latency situations (such as when someone on your LAN is downloading the entire *Lord of the Rings* trilogy). However, despite the improved client-side caching, NFS is still a stateless protocol. Clients maintain no information about available servers across reboots, and the client-side cache is likewise lost on reboot. In addition, if a server reboots or

becomes unavailable when a client has pending (uncommitted) file changes, these changes will be lost if the server does not come back up fairly soon.

Complementing the new version's greater Internet-friendliness, NFSv4 also supports Unicode (UTF-8) filenames, making cross-platform and intercharacter set file sharing more seamless and more international.

When applicable, this chapter will discuss using NFSv4 features, include examples of NFSv4 clients and servers, and warn you of potential problems you might encounter when using NFSv4.

NOTE For more information about NFSv4 in Fedora Core and Red Hat Enterprise Linux, see Van Emery's excellent article, "Learning NFSv4 with Fedora Core 2," on the Web at www.vanemery.com/Linux/NFSv4/NFSv4-no-rpcsec.html. The NFSv4 open-source reference implementation is driven by the Center for Information Technology Integration (CITI) at the University of Michigan, which maintains an information-rich Web site at www.citi.umich.edu/projects/nfsv4.

NFS Advantages and Disadvantages

Clearly, the biggest advantage NFS provides is centralized control, maintenance, and administration. It is much easier, for example, to back up a file system stored on a single server than it is to back up directories scattered across a network, on systems that are geographically dispersed, and that might or might not be accessible when the backup is made. Similarly, NFS makes it trivial to provide access to shared disk space, or limit access to sensitive data. When NFS and NIS are used together, changes to systemwide configuration files, such as authentication files or network configuration information, can be quickly and automatically propagated across the network without requiring system administrators to physically visit each machine or requiring users to take any special action.

NFS can also conserve disk space and prevent duplication of resources. Read-only file systems and file systems that change infrequently, such as `/usr`, can be exported as read-only NFS mounts. Likewise, upgrading applications employed by users throughout a network simply becomes a matter of installing the new application and changing the exported file system to point at the new application.

End users also benefit from NFS. When NFS is combined with NIS, users can log in from any system, even remotely, and still have access to their home directories and see a uniform view of shared data. Users can protect important or sensitive data or information that would be impossible or time-consuming to re-create by storing it on an NFS mounted file system that is regularly backed up.

NFS has its shortcomings, of course, primarily in terms of performance and security. As a distributed, network-based file system, NFS is sensitive to network congestion. Heavy network traffic slows down NFS performance. Similarly, heavy disk activity on the NFS server adversely affects NFS's performance. In the face of network congestion or extreme disk activity, NFS clients run more slowly because file I/O takes longer. The performance enhancements incorporated in NFSv4 have increased NFS's stability and reliability on high latency and heavily congested networks, but it should be clear that unless you are on a high-speed network, such as Gigabit Ethernet or Myrinet, NFS will not be as fast as a local disk.

If an exported file system is not available when a client attempts to mount it, the client system can hang, although this can be mitigated using a specific mount option that you will read about in the section titled "Configuring an NFS Client." Another shortcoming of NFS is that an exported file system represents a single point of failure. If the disk or system exporting vital data or application becomes unavailable for any reason, such as a disk crash or server failure, no one can access that resource.

NFS suffers from potential security problems because its design assumes a trusted network, not a hostile environment in which systems are constantly being probed and attacked. The primary weakness of most NFS implementations based on protocol versions 1, 2, and 3 is that they are based on standard (unencrypted) remote procedure calls (RPC). RPC is one of the most common targets of exploit attempts. As a result, sensitive information should never be exported from or mounted on systems directly exposed to the Internet, that is, one that is on or outside a firewall. While `RPCSEC_GSS` makes NFSv4 more secure and *perhaps* safer to use on Internet-facing systems, evaluate such usage carefully and perform testing before deploying even a version 4–based NFS system across the Internet. *Never* use NFS versions 3 and earlier on systems that front the Internet; clear-text protocols are trivial for anyone with a packet sniffer to intercept and interpret.

NOTE An NFS client using NFS servers inside a protected network can safely be exposed to the Internet because traffic between client and server travels across the protected network. What we are discouraging is accessing an NFSv3 (or earlier) export across the Internet.

Quite aside from encryption and even inside a firewall, providing all users access to all files might pose greater risks than user convenience and administrative simplicity justify. Care must be taken when configuring NFS exports to limit access to the appropriate users and also to limit what those users are permitted to do with the data. Moreover, NFS has quirks that can prove disastrous

for unwary or inexperienced administrators. For example, when the root user on a client system mounts an NFS export, you do not want the root user on the client to have root privileges on the exported file system. By default, NFS prevents this, a procedure called *root squashing*, but a careless administrator might override it.

Planning an NFS Installation

Planning an NFS installation is a grand-sounding phrase that boils down to thoughtful design followed by careful implementation. Of these two steps, design is the more important because it ensures that the implementation is transparent to end users and trivial to the administrator. The implementation is remarkably straightforward. This section highlights the server configuration process and discusses the key design issues to consider.

“Thoughtful design” consists of deciding what file systems to export to which users and selecting a naming convention and mounting scheme that maintains network transparency. When you are designing your NFS installation, you need to:

- Select the file systems to export
- Establish which users (or hosts) are permitted to mount the exported file systems
- Identify the automounting or manual mounting scheme that clients will use to access exported file systems
- Choose a naming convention and mounting scheme that maintains network transparency and ease of use

With the design in place, implementation is a matter of configuring the exports and starting the appropriate daemons. Testing ensures that the naming convention and mounting scheme works as designed and identifies potential performance bottlenecks. Monitoring is an ongoing process to ensure that exported file systems continue to be available, network security and the network security policy remain uncompromised, and that heavy usage does not adversely affect overall performance.

A few general rules exist to guide the design process. You need to take into account site-specific needs, such as which file systems to export, the amount of data that will be shared, the design of the underlying network, what other network services you need to provide, and the number and type of servers and clients. The following tips and suggestions for designing an NFS server and its exports will simplify administrative tasks and reduce user confusion:

- Good candidates for NFS exports include any file system that is shared among a large number of users, such as `/home`, workgroup project directories, shared data directories, such as `/usr/share`, the system mail spool (`/var/spool/mail`), and file systems that contain shared application binaries and data. File systems that are relatively static, such as `/usr`, are also good candidates for NFS exports because there is no need to replicate the same static data and binaries across multiple machines.

TIP A single NFS server can export binaries for multiple platforms by exporting system-specific subdirectories. So, for example, you can export a subdirectory of Linux binaries from a Solaris NFS server with no difficulty. The point to emphasize here is that NFS can be used in heterogeneous environments as seamlessly as it can be used in homogeneous network installations.

- Use `/home/username` to mount home directories. This is one of the most fundamental directory idioms in the Linux world, so disregarding it not only antagonizes users but also breaks a lot of software that presumes user home directories live in `/home`. On the server, you have more leeway about where to situate the exports. Recall from Figure 12-1, for example, that diskbeast stored user home directories in `/exports/home`.
- Few networks are static, particularly network file systems, so design NFS servers with growth in mind. For example, avoid the temptation to drop all third-party software onto a single exported file system. Over time, file systems usually grow to the point that they need to be subdivided, leading to administrative headaches when client mounts must be updated to reflect a new set of exports. Spread third-party applications across multiple NFS exports and export each application and its associated data separately.
- If the previous tip will result in a large number of NFS mounts for clients, it might be wiser to create logical volume sets on the NFS server. By using logical volumes underneath the exported file systems, you can increase disk space on the exported file systems as it is needed without having to take the server down or take needed exports offline.
- At large sites, distribute multiple NFS exports across multiple disks so that a single disk failure will limit the impact to the affected application. Better still, to minimize downtime on singleton servers, use RAID for redundancy and logical volumes for flexibility. If you have the capacity, use NFSv4's replication facilities to ensure that exported file systems remain available even if the primary NFS server goes up in smoke.

- Similarly, overall disk and network performance improves if you distribute exported file systems across multiple servers rather than concentrate them on a single server. If it is not possible to use multiple servers, at least try to situate NFS exports on separate physical disks and/or on separate disk controllers. Doing so reduces disk I/O contention.

When identifying the file systems to export, keep in mind a key restriction on which file systems can be exported and how they can be exported. You can export only local file systems and their subdirectories. To express this restriction in another way, you cannot export a file system that is itself already an NFS mount. For example, if a client system named `userbeast` mounts `/home` from a server named `homebeast`, `userbeast` cannot reexport `/home`. Clients wishing to mount `/home` must do so directly from `homebeast`.

Configuring an NFS Server

This section shows you how to configure an NFS server, identifies the key files and commands you use to implement, maintain, and monitor the NFS server, and illustrates the server configuration process using a typical NFS setup.

On Fedora Core and Red Hat Enterprise Linux systems, the `/etc/exports` file is the main NFS configuration file. It lists the file systems the server exports, the systems permitted to mount the exported file systems, and the mount options for each export. NFS also maintains status information about existing exports and the client systems that have mounted those exports in `/var/lib/nfs/rmtab` and `/var/lib/nfs/xtab`.

In addition to these configuration and status files, all of the daemons, commands, initialization scripts, and configuration files in the following list are part of NFS. Don't panic because the list is so long, though; you have to concern yourself with only a few of them to have a fully functioning and properly configured NFS installation. Notice that approximately half of the supporting files are part of NFSv4 — presumably the price one pays for added features.

- Daemons
 - `rpc.gssd` (new in NFSv4)
 - `rpc.idmapd` (new in NFSv4)
 - `rpc.lockd`
 - `rpc.mountd`
 - `rpc.nfsd`
 - `rpc.portmap`

- `rpc.rquotad`
- `rpc.statd`
- `rpc.svcgssd` (new in NFSv4)
- Configuration files (in `/etc`)
 - `exports`
 - `gssapi_mech.conf` (new in NFSv4)
 - `idmapd.conf` (new in NFSv4)
- Initialization scripts (in `/etc/rc.d/init.d`)
 - `nfs`
 - `rpcgssd` (new in NFSv4)
 - `rpcidmapd` (new in NFSv4)
 - `rpcsvcgssd` (new in NFSv4)
- Commands
 - `exportfs`
 - `nfsstat`
 - `showmount`
 - `rpcinfo`

NFS Server Configuration and Status Files

The server configuration file is `/etc/exports`, which contains a list of file systems to export, the clients permitted to mount them, and the export options that apply to client mounts. Each line in `/etc/exports` has the following format:

```
dir [host](options) [...]
```

`dir` specifies a directory or file system to export, `host` specifies one or more hosts permitted to mount `dir`, and `options` specifies one or more mount options. If you omit `host`, the listed options apply to every possible client system, likely not something you want to do. If you omit `options`, the default mount options (described shortly) will be applied. Do not insert a space between the hostname and the opening parenthesis that contains the export options; a space between the hostname and the opening parenthesis of the option list has four (probably unintended) consequences:

1. *Any* NFS client can mount the export.
2. You'll see an abundance of error messages in `/var/log/messages`.

3. The list options will be applied to *all* clients, not just the client(s) identified by the host specification.
4. The client(s) identified by the host specification will have the default mount options applied, not the mount options specified by options.

host can be specified as a single name, an NIS netgroup, a subnet using address/net mask form, or a group of hostnames using the wildcard characters ? and *. Multiple host(options) entries, separated by whitespace, are also accepted, enabling you to specify different export options for a single dir depending on the client.

TIP The exports manual (man) page recommends not using the wildcard characters * and ? with IP addresses because they don't work except by accident when reverse DNS lookups fail. We've used the wildcard characters without incident on systems we administer, but, as always, your mileage may vary.

When specified as a single name, host can be any name that DNS or the resolver library can resolve to an IP address. If host is an NIS netgroup, it is specified as @groupname. The address/net mask form enables you to specify all hosts on an IP network or subnet. In this case the net mask can be specified in dotted quad format (/255.255.252.0, for example) or as a mask length (such as /22). As a special case, you can restrict access to an export to only those clients using RPCSEC_GSS security by using the client specification gss/krb5. If you use this type of client specification, you cannot also specify an IP address. You may also specify the host using the wildcards * and ?.

Consider the following sample /etc/exports file:

```
/usr/local      *.example.com(ro)
/usr/devtools   192.168.1.0/24(ro)
/home           192.168.0.0/255.255.255.0(rw)
/projects       @dev(rw)
/var/spool/mail 192.168.0.1(rw)
/opt/kde        gss/krb5(ro)
```

The first line permits all hosts with a name of the format somehost.example.com to mount /usr/local as a read-only directory. The second line uses the address/net mask form in which the net mask is specified in Classless Inter-Domain Routing (CIDR) format. In the CIDR format, the net mask is given as the number of bits (/24, in this example) used to determine the network address. A CIDR address of 192.168.1.0/24 allows any host with an IP address in the range 192.168.1.1 to 192.168.1.254 (192.168.1.0 is excluded because it is the network address; 192.168.1.255 is excluded because it is the broadcast address) to mount /usr/devtools read-only. The third line permits any host

with an IP address in the range 192.168.0.1 to 192.168.0.254 to mount `/home` in read-write mode. This entry uses the address/net mask form in which the net mask is specified in dotted quad format. The fourth line permits any member of the NIS netgroup named `dev` to mount `/projects` (again, in read-write mode). The fifth line permits only the host whose IP address is 192.168.0.1 to mount `/var/mail`. The final line allows any host using `RPCSEC_GSS` security to mount `/opt/kde` in read-only mode.

TIP If you have trouble remembering how to calculate IP address ranges using the address/net mask format, use the excellent `ipcalc` utility created by **Krischan Jodies**. You can download it from his Web site (jodies.de/ipcalc/) or from the Web site supporting this book, wiley.com/go/redhat-admin3e.

The export options, listed in parentheses after the host specification, determine the characteristics of the exported file system. Table 12-1 lists valid values for options.

Table 12-1 Nfs Export Options

OPTION	DESCRIPTION
<code>all_squash</code>	Maps all requests from all UIDs or GIDs to the UID or GID, respectively, of the anonymous user.
<code>anongid=gid</code>	Sets the GID of the anonymous account to <code>gid</code> .
<code>anonuid=uid</code>	Sets the UID of the anonymous account to <code>uid</code> .
<code>async</code>	Allows the server to cache disk writes to improve performance.
<code>fsid=n</code>	Forces NFS's internal file system identification (FSID) number to be <code>n</code> .
<code>hide</code>	Hides an exported file system that is a subdirectory of another exported file system.
<code>insecure</code>	Permits client requests to originate from unprivileged ports (those numbered 1024 and higher).
<code>insecure_locks</code>	Disables the need for authentication before activating lock operations (synonym for <code>no_auth_nlm</code>).
<code>mp[=path]</code>	Exports the file system specified by <code>path</code> only if the corresponding mount point is mounted (synonym for <code>mountpoint[=path]</code>).
<code>no_all_squash</code>	Disables <code>all_squash</code> .
<code>no_root_squash</code>	Disables <code>root_squash</code> .

Table 12-1 (continued)

OPTION	DESCRIPTION
<code>no_subtree_check</code>	Disables <code>subtree_check</code> .
<code>no_wdelay</code>	Disables <code>wdelay</code> (must be used with the <code>sync</code> option).
<code>nohide</code>	Does not hide an exported file system that is a subdirectory of another exported file system.
<code>ro</code>	Exports the file system read-only, disabling any operation that changes the file system.
<code>root_squash</code>	Maps all requests from a user ID (UID) or group ID (GID) of 0 to the UID or GID, respectively, of the anonymous user (-2 in Red Hat Linux).
<code>rw</code>	Exports the file system read-write, permitting operations that change the file system.
<code>secure</code>	Requires client requests to originate from a secure (privileged) port, that is, one numbered less than 1024.
<code>secure_locks</code>	Requires that clients requesting lock operations be properly authenticated before activating the lock (synonym for <code>auth_nlm</code>).
<code>subtree_check</code>	If only part of a file system, such as a subdirectory, is exported, subtree checking makes sure that file requests apply to files in the exported portion of the file system.
<code>sync</code>	Forces the server to perform a disk write before notifying the client that the request is complete.
<code>wdelay</code>	Instructs the server to delay a disk write if it believes another related disk write may be requested soon or if one is in progress, improving overall performance.

TIP Recent versions of NFS (actually, of the NFS utilities) default to exporting directories using the `sync` option. This is a change from past practice, in which directories were exported and mounted using the `async` option. This change was made because defaulting to `async` violated the NFS protocol specification.

The various squash options, and the `anonuid` and `anongid` options require additional explanation. `root_squash` prevents the root user on an NFS client from having root privileges on an NFS server via the exported file system. The Linux security model ordinarily grants root full access to the file systems on a host. However, in an NFS environment, exported file systems are shared resources that are properly “owned” by the root user of the NFS *server*, not by

the root users of the client systems that mount them. The `root_squash` option remaps the root UID and GID (0) on the client system to a less privileged UID and GID, -2. Remapping the root UID and GID prevents NFS clients from inappropriately taking ownership of NFS exports by. The `no_root_squash` option disables this behavior, but should not be used because doing so poses significant security risks. Consider the implications, for example, of giving a client system root access to the file system containing sensitive payroll information.

The `all_squash` option has a similar effect to `root_squash`, except that it applies to all users, not just the root user. The default is `no_all_squash`, however, because most users that access files on NFS exported file systems are already merely mortal users, that is, they have unprivileged UIDs and GIDs, so they do not have the power of the root account. Use the `anonuid` and `anongid` options to specify the UID and GID of the anonymous user. The default UID and GID of the anonymous user is -2, which should be adequate in most cases.

`subtree_check` and `no_subtree_check` also deserve some elaboration. When a subdirectory of file system is exported but the entire file system is not exported, the NFS server must verify that the accessed file resides in the exported portion of the file system. This verification, called a *subtree check*, is programmatically nontrivial to implement and can negatively impact NFS performance. To facilitate subtree checking, the server stores file location information in the file handles given to clients when they request a file.

In most cases, storing file location information in the file handle poses no problem. However, doing so becomes potentially troublesome when an NFS client is accessing a file that is renamed or moved *while the file is open*. Moving or renaming the file invalidates the location information stored in the file handle, so the next client I/O request on that file causes an error. Disabling the subtree check using `no_subtree_check` prevents this problem because the location information is not stored in the file handle when subtree checking is disabled. As an added benefit, disabling subtree checking improves performance because it removes the additional overhead involved in the check. The benefit is especially significant on exported file systems that are highly dynamic, such as `/home`.

Unfortunately, disabling subtree checking also poses a security risk. The subtree check routine ensures that files to which only root has access can be accessed only if the file system is exported with `no_root_squash`, even if the file's permissions permit broader access.

The manual page for `/etc/exports` recommends using `no_subtree_check` for `/home` because `/home` file systems normally experiences a high level of file renaming, moving, and deletion. It also recommends leaving subtree checking enabled (the default) for file systems that are exported read-only; file systems that are largely static (such as `/usr` or `/var`); and file systems from which only subdirectories and not the entire file system, are exported.

The `hide` and `nohide` options mimic the behavior of NFS on SGI's IRIX. By default, if an exported directory is a subdirectory of another exported directory, the exported subdirectory will be hidden unless both the parent and child exports are explicitly mounted. The rationale for this feature is that some NFS client implementations cannot deal with what appears to be two different files having the same inode. In addition, directory hiding simplifies client- and server-side caching. You can disable directory hiding by specifying `nohide`.

The final interesting mount option is `mp`. If set, the NFS server will not export a file system unless that file system is actually mounted on the server. The reasoning behind this option is that a disk or file system containing an NFS export might not mount successfully at boot time or might crash at run-time. This measure prevents NFS clients from mounting unavailable exports.

Here is a modified version of the `/etc/exports` file presented earlier:

```
/usr/local      *.example.com(mp,ro,secure)
/usr/devtools  192.168.1.0/24(mp,ro,secure)
/home          192.168.0.0/255.255.255.0(mp,rw,secure,no_subtree_check)
/projects      @dev(mp,rw,secure,anonuid=600,anongid=600,sync,no_wdelay)
/var/mail      192.168.0.1(mp,rw,insecure,no_subtree_check)
/opt/kde       gss/krb5(mp,ro,async)
```

The hosts have not changed, but additional export options have been added. All file systems use the `mp` option to make sure that only mounted file systems are available for export. `/usr/local`, `/usr/devtools`, `/home`, and `/project` can be accessed only from clients using secure ports (the `secure` option), but the server accepts requests destined for `/var/mail` from any port because the `insecure` option is specified. For `/projects`, the anonymous user is mapped to the UID and GID 600, as indicated by the `anonuid=600` and `anongid=600` options. The wrinkle in this case is that only members of the NIS netgroup `dev` will have their UIDs and GIDs mapped because they are the only NFS clients permitted to mount `/projects`.

`/home` and `/var/mail` are exported using the `no_subtree_check` option because they see a high volume of file renaming, moving, and deletion. Finally, the `sync` and `no_wdelay` options disable write caching and delayed writes to the `/project` file system. The rationale for using `sync` and `no_wdelay` is that the impact of data loss would be significant in the event the server crashes. However, forcing disk writes in this manner also imposes a performance penalty because the NFS server's normal disk caching and buffering heuristics cannot be applied.

If you intend to use NFSv4-specific features, you need to be familiar with the `RPCSEC_GSS` configuration files, `/etc/gssapi_mech.conf` and `/etc/idmapd.conf`. `idmapd.conf` is the configuration file for NFSv4's `idmapd` daemon. `idmapd` works on the behalf of both NFS servers and clients to translate NFSv4 IDs to user and group IDs and vice versa; `idmapd.conf` controls

idmapd's runtime behavior. The default configuration (with comments and blank lines removed) should resemble Listing 12-1.

```
[General]
Verbosity = 0
Pipefs-Directory = /var/lib/nfs/rpc_pipefs
Domain = localdomain
[Mapping]
Nobody-User = nobody
Nobody-Group = nobody
[Translation]
Method = nsswitch
```

Listing 12-1 Default idmapd configuration.

In the [General] section, the Verbosity option controls the amount of log information that idmapd generates; Pipefs-directory tell idmapd where to find the RPC pipe file system it should use (idmapd communicates with the kernel using the pipefs virtual file system); Domain identifies the default domain. If Domain isn't specified, it defaults to the server's fully qualified domain name (FQDN) less the hostname. For example, if the FQDN is coondog.example.com, the Domain parameter would be example.com; if the FQDN is mail.admin.example.com, the Domain parameter would be the subdomain admin.example.com. The Domain setting is probably the only change you will need to make to idmapd's configuration.

The [Mapping] section identifies the user and group names that correspond to the nobody user and group that NFS server should use. The option Method = nsswitch, finally, tells idmapd how to perform the name resolution. In this case, names are resolved using the name service switch (NSS) features of glibc.

The /etc/gssapi_mech.conf file controls the GSS daemon (rpc.svcgssd). You won't need to modify this file. As provided in Fedora Core and RHEL, gssapi_mech.conf lists the specific function call to use to initialize a given GSS library. Programs (in this case, NFS) need this information if they intend to use secure RPC.

Two additional files store status information about NFS exports, /var/lib/nfs/rmtab and /var/lib/nfs/etab. /var/lib/nfs/rmtab is the table that lists each NFS export that is mounted by an NFS client. The daemon rpc.mountd (described in the section "NFS Server Daemons") is responsible for servicing requests to mount NFS exports. Each time the rpc.mountd daemon receives a mount request, it adds an entry to /var/lib/nfs/rmtab. Conversely, when mountd receives a request to unmount an exported file system, it removes the corresponding entry from /var/lib/nfs/rmtab. The following short listing shows the contents of /var/lib/nfs/rmtab on an NFS

server that exports `/home` in read-write mode and `/usr/local` in read-only mode. In this case, the host with IP address 192.168.0.4 has mounted both exports:

```
$ cat /var/lib/nfs/rmtab
192.168.0.4:/home:0x00000001
192.168.0.4:/usr/local:0x00000001
```

Fields in `rmtab` are colon-delimited, so it has three fields: the host, the exported file system, and the mount options specified in `/etc/exports`. Rather than try to decipher the hexadecimal options field, though, you can read the mount options directly from `/var/lib/nfs/etab`. The `exportfs` command, discussed in the subsection titled “NFS Server Scripts and Commands,” maintains `/var/lib/nfs/etab`. `etab` contains the table of currently exported file systems. The following listing shows the contents of `/var/lib/nfs/etab` for the server exporting the `/usr/local` and `/home` file systems shown in the previous listing (the output wraps because of page width constraints).

```
$ cat /var/lib/nfs/etab
/usr/local
192.168.0.4(ro,sync,wdelay,hide,secure,root_squash,no_all_squash,
subtree_check,secure_locks,mapping=identity,anonuid=-2,anongid=-2)
/home
192.168.0.2(rw,sync,wdelay,hide,secure,root_squash,no_all_squash,
subtree_check,secure_locks,mapping=identity,anonuid=-2,anongid=-2)
```

As you can see in the listing, the format of the `etab` file resembles that of `/etc/exports`. Notice, however, that `etab` lists the default values for options not specified in `/etc/exports` in addition to the options specifically listed.

NOTE Most Linux systems use `/var/lib/nfs/etab` to store the table of currently exported file systems. The manual page for the `exportfs` command, however, states that `/var/lib/nfs/xtab` contains the table of current exports. We do not have an explanation for this — it’s just a fact of life that the manual page and actual usage differ.

The last two configuration files to discuss, `/etc/hosts.allow` and `/etc/hosts.deny`, are not, strictly speaking, part of the NFS server. Rather, `/etc/hosts.allow` and `/etc/hosts.deny` are access control files used by the TCP Wrappers system; you can configure an NFS server without them and the server will function perfectly (to the degree, at least, that *anything* ever functions perfectly). However, using TCP Wrappers’ access control features helps enhance both the overall security of the server and the security of the NFS subsystem.

The TCP Wrappers package is covered in detail in Chapter 19. Rather than preempt that discussion here, we suggest how to modify these files, briefly explain the rationale, and suggest you refer to Chapter 19 to understand the modifications in detail.

First, add the following entries to `/etc/hosts.deny`:

```
portmap:ALL
lockd:ALL
mountd:ALL
rquotad:ALL
statd:ALL
```

These entries deny access to NFS services to all hosts not explicitly permitted access in `/etc/hosts.allow`. Accordingly, the next step is to add entries to `/etc/hosts.allow` to permit access to NFS services to specific hosts. As you will learn in Chapter 19, entries in `/etc/hosts.allow` take the form:

```
daemon:host_list [host_list]
```

TIP The NFS HOWTO (<http://nfs.sourceforge.net/nfs-howto/server.html#CONFIG>) discourages use of the `ALL:ALL` syntax in `/etc/hosts.deny`, using this rationale: “While [denying access to all services] is more secure behavior, it may also get you in trouble when you are installing new services, you forget you put it there, and you can’t figure out for the life of you why they won’t work.”

We respectfully disagree. The stronger security enabled by the `ALL:ALL` construct in `/etc/hosts.deny` far outweighs any inconvenience it might pose when configuring new services.

`daemon` is a daemon such as `portmap` or `lockd`, and `host_list` is a list of one or more hosts specified as hostnames, IP addresses, IP address patterns using wildcards, or address/net mask pairs. For example, the following entry permits all hosts in the `example.com` domain to access the `portmap` service:

```
portmap:.example.com
```

The next entry permits access to all hosts on the subnetworks `192.168.0.0` and `192.168.1.0`:

```
portmap:192.168.0. 192.168.1.
```

You need to add entries for each host or host group permitted NFS access for each of the five daemons listed in `/etc/hosts.deny`. So, for example, to

permit access to all hosts in the `example.com` domain, add the following entries to `/etc/host.allow`:

```
portmap:.example.com
lockd  :.example.com
mountd :.example.com
rquotad:.example.com
statd  :.example.com
```

Therefore, a name of the form `.domain.dom` matches all hosts, including hosts in subdomains like `.subdom.domain.dom`.

NFS Server Daemons

Providing NFS services requires the services of six daemons: `/sbin/portmap`, `/usr/sbin/rpc.mountd`, `/usr/sbin/rpc.nfsd`, `/sbin/rpc.statd`, `/sbin/rpc.lockd`, and, if necessary, `/usr/sbin/rpc.rquotad`. They are generally referred to as `portmap`, `mountd`, `nfssd`, `statd`, `lockd`, and `rquotad`, respectively. If you intend to take advantage of NFSv4's enhancements, you'll also need to know about `rpc.gssd`, `rpc.idmapd`, and `rpc.svcgssd`. For convenience's sake, we'll refer to these daemons using the shorthand expressions `gssd`, `idmapd`, and `svcgssd`. Table 12-2 briefly describes each daemon's purpose.

Table 12-2 Nfs Server Daemons

DAEMON	FUNCTION
<code>gssd</code>	Creates security contexts on RPC clients for exchanging RPC information using SecureRPC (RPCSEC) using GSS
<code>idmapd</code>	Maps local user and group names to NFSv4 IDs (and vice versa)
<code>lockd</code>	Starts the kernel's NFS lock manager
<code>mountd</code>	Processes NFS client mount requests
<code>nfsd</code>	Provides all NFS services except file locking and quota management
<code>portmap</code>	Enables NFS clients to discover the NFS services available on a given NFS server
<code>rquotad</code>	Provides file system quota information NFS exports to NFS clients using file system quotas
<code>statd</code>	Implements NFS lock recovery when an NFS server system crashes
<code>svcgssd</code>	Creates security contexts on RPC servers for exchanging RPC information using SecureRPC (RPCSEC) using GSS

The NFS server daemons should be started in the following order to work properly:

1. portmap
2. nfsd
3. mountd
4. statd
5. rquotad (if necessary)
6. idmapd
7. svcgssd

The start order is handled for you automatically at boot time if you have enabled NFS services using Service Configuration Tool (`/usr/bin/system-config-services`).

Notice that the list omits `lockd`. `nfsd` starts it on an as-needed basis, so you should rarely, if ever, need to invoke it manually. Fortunately, the Red Hat Linux initialization script for NFS, `/etc/rc.d/init.d/nfs`, takes care of starting up the NFS server daemons for you. Should the need arise, however, you can start NFS yourself by executing the handy `service` utility script directly:

```
# service nfs start
Starting NFS services:           [ OK ]
Starting NFS quotas:           [ OK ]
Starting NFS daemon:           [ OK ]
Starting NFS mountd:           [ OK ]
```

You can also use:

```
# /etc/rc.d/init.d/nfs start
Starting NFS services:           [ OK ]
Starting NFS quotas:           [ OK ]
Starting NFS daemon:           [ OK ]
Starting NFS mountd:           [ OK ]
```

By default, the startup script starts eight copies of `nfsd` to enable the server to process multiple requests simultaneously. To change this value, edit `/etc/sysconfig/nfs` and add an entry resembling the following (you need to be root to edit this file):

```
RPCNFSDCOUNT=n
```

Replace *n* with the number of `nfsd` processes you want to start. Busy servers with many active connections might benefit from doubling or tripling this

number. If file system quotas for exported file systems have *not* been enabled on the NFS server, it is unnecessary to start the quota manager, `rquotad`, but be aware that the initialization script starts `rquotad` whether quotas have been enabled or not.

TIP If `/etc/sysconfig/nfs` does not exist, you can create it using your favorite text editor. In a pinch, you can use the following command to create it with the `RPCNFSDCOUNT` setting mentioned in the text:

```
# cat > /etc/sysconfig/nfs
RPCNFSDCOUNT=16
^d
```

^d is the end-of-file mark, generated by pressing the Control key and d simultaneously.

NFS Server Scripts and Commands

Three initialization scripts control the required NFS server daemons, `/etc/rc.d/init.d/portmap`, `/etc/rc.d/init.d/nfs`, and `/etc/rc.d/init.d/nfslock`. The `exportfs` command enables you to manipulate the list of current exports on the fly without needing to edit `/etc/exports`. The `showmount` command provides information about clients and the file systems they have mounted. The `nfsstat` command displays detailed information about the status of the NFS subsystem.

The `portmap` script starts the `portmap` daemon, frequently referred to as *the portmapper*. All programs that use RPC, such as NIS and NFS, rely on the information the portmapper provides. The portmapper starts automatically at boot time, so you rarely need to worry about it, but it is good to know you can control it manually. Like most startup scripts, it requires a single argument, such as `start`, `stop`, `restart`, or `status`. As you can probably guess, the `start` and `stop` arguments start and stop the portmapper, `restart` restarts it (by calling the script with the `start` and `stop` arguments, as it happens), and `status` indicates whether the portmapper is running, showing the portmapper's PID if it is running.

The primary NFS startup script is `/etc/rc.d/init.d/nfs`. Like the portmapper, it requires a single argument, `start`, `stop`, `status`, `restart`, or `reload`. `start` and `stop` start and stop the NFS server, respectively. The `restart` argument stops and starts the server processes in a single command and can be used after changing the contents of `/etc/exports`. However, it is not necessary to reinitialize the NFS subsystem by bouncing the server daemons in this way. Rather, use the script's `reload` argument, which causes `exportfs`, discussed shortly, to reread `/etc/exports` and to reexport the

file systems listed there. Both `restart` and `reload` also update the timestamp on the NFS lock file (`/var/lock/subsys/nfs`) used by the initialization script. The `status` argument displays the PIDs of the `mountd`, `nfsd`, and `rquotad` daemons. For example:

```
$ service nfs status
rpc.mountd (pid 4358) is running...
nfsd (pid 1241 1240 1239 1238 1235 1234 1233 1232) is running...
rpc.rquotad (pid 1221) is running...
```

The output of the command confirms that the three daemons are running and shows the PIDs for each instance of each daemon. All users are permitted to invoke the NFS initialization script with the `status` argument, but all the other arguments (`start`, `stop`, `restart`, and `reload`) require root privileges.

NFS services also require the file-locking daemons `lockd` and `statd`. As explained earlier, `nfsd` starts `lockd` itself, but you still must start `statd` separately. You can use an initialization script for this purpose, `/etc/rc.d/init.d/nfslock`. It accepts almost the same arguments as `/etc/rc.d/init.d/nfs` does, with the exception of the `reload` argument (because `statd` does not require a configuration file).

To tie everything together, if you ever need to start the NFS server manually, the proper invocation sequence is to start the portmapper first, followed by NFS, followed by the NFS lock manager, that is:

```
# service portmap start
# service nfs start
# service nfslock start
```

Conversely, to shut down the server, reverse the start procedure:

```
# service nfslock stop
# service nfs stop
# service portmap stop
```

Because other programs and servers may require the portmapper's service, we suggest that you let it run unless you drop the system to run level 1 to perform maintenance.

You can also find out what NFS daemons are running using the `rpcinfo` command with the `-p` option. `rpcinfo` is a general-purpose program that displays information about programs that use the RPC protocol, of which NFS is one. The `-p` option queries the portmapper and displays a list of all registered RPC programs. The following listing shows the output of `rpcinfo -p` on a fairly quiescent NFS server:

```
$ /usr/sbin/rpcinfo -p
program vers proto  port
100000      2    tcp    111  portmapper
100000      2    udp    111  portmapper
100011      1    udp    961  rquotad
100011      2    udp    961  rquotad
100011      1    tcp    964  rquotad
100011      2    tcp    964  rquotad
100003      2    udp    2049 nfs
100003      3    udp    2049 nfs
100003      4    udp    2049 nfs
100003      2    tcp    2049 nfs
100003      3    tcp    2049 nfs
100003      4    tcp    2049 nfs
100021      1    udp    32770 nlockmgr
100021      3    udp    32770 nlockmgr
100021      4    udp    32770 nlockmgr
100021      1    tcp    35605 nlockmgr
100021      3    tcp    35605 nlockmgr
100021      4    tcp    35605 nlockmgr
100005      1    udp    32772 mountd
100005      1    tcp    32825 mountd
100005      2    udp    32772 mountd
100005      2    tcp    32825 mountd
100005      3    udp    32772 mountd
100005      3    tcp    32825 mountd
```

`rpcinfo`'s output shows the RPC program's ID number, version number, the network protocol it is using, the port number it is using, and an alias name for the program number. The program number and name (first and fifth columns) are taken from the file `/etc/rpc`, which maps program numbers to program names and also lists aliases for program names. At a bare minimum, to have a functioning NFS server, `rpcinfo` should list entries for `portmapper`, `nfs`, and `mountd`.

The `exportfs` command enables you to manipulate the list of available exports, in some cases without editing `/etc/exports`. It also maintains the list of currently exported file systems in `/var/lib/nfs/etab` and the kernel's internal table of exported file systems. In fact, the NFS initialization script discussed earlier in this subsection uses `exportfs` extensively. For example, the `exportfs -a` command initializes `/var/lib/nfs/etab`, synchronizing it with the contents of `/etc/exports`. To add a new export to `etab` and to the kernel's internal table of NFS exports *without* editing `/etc/exports`, use the following syntax:

```
exportfs -o opts host:dir
```


`opts`, `host`, and `dir` use the same syntax as that described for `/etc/exports` earlier in the chapter. Consider the following command:

```
# exportfs -o async,rw 192.168.0.3:/var/spool/mail
```

This command exports `/var/spool/mail` with the `async` and `rw` options to the host whose IP address is `192.168.0.3`. This invocation is exactly equivalent to the following entry in `/etc/exports`:

```
/var/spool/mail 192.168.0.3(async,rw)
```

A bare `exportfs` call lists all currently exported file systems; adding the `-v` option lists currently exported file systems with their mount options.

```
# exportfs -v
/usr/local      192.168.0.4(ro,wdelay,root_squash)
/home           192.168.0.4(rw,wdelay,root_squash)
```

To remove an exported file system, use the `-u` option with `exportfs`. For example, the following command unexports the `/home` file system shown in the previous example.

```
# exportfs -v -u 192.168.0.4:/home
unexporting 192.168.0.4:/home
```

The `showmount` command queries the mount daemon, `mountd`, about the status of the NFS server. Its syntax is:

```
showmount [-adehv] [host]
```

Invoked with no options, `showmount` displays a list of all clients that have mounted file systems from the current host. Specify `host` to query the mount daemon on that host, where `host` can be a resolvable DNS hostname or, as in the following example, an IP address:

```
# showmount 192.168.0.1
Hosts on 192.168.0.1:
192.168.0.0/24
192.168.0.1
```

Table 12-3 describes the effects of `showmount`'s options.

Table 12-3 Options for the showmount Command

OPTION	DESCRIPTION
-a	Displays client hostnames and mounted directories in host:directory format
-d	Displays only the directories clients have mounted
-e	Displays the NFS server's list of exported file systems
-h	Displays a short usage summary
--no-headers	Disables displaying descriptive headings for showmount's output
-v	Displays showmount's version number

The following examples show the output of showmount executed on an NFS server that has exported /media to the client named bubba.example.com, which has an IP address of 192.168.0.2, using the following entry in /etc/exports:

```
/media 192.168.0.0/24(rw)
```

The first command uses the -a option for the most comprehensive output, the second uses the -d option to show only the mounted directories, and the third example uses -e to show the server's export list.

```
# showmount -a
All mount points on bubba.example.com:
192.168.0.0/24:/media
192.168.0.1:192.168.0.0/24
# showmount -d
Directories on bubba.example.com:
/media
# showmount -e
Export list for bubba.example.com:
/media 192.168.0.0/24
```

The showmount command is most useful on potential NFS clients because they can identify the directories an NFS server is exporting. By the same token, however, this poses a security risk because, in the absence of entries in /etc/hosts.deny that forbid access to the portmapper, *any* host can obtain this information from an NFS server.

Using Secure NFS

Although NFSv4 is installed, the default installation does not use NFSv4's security enhancements by default. You need to set this up manually. To do so, use the following procedure:

1. Enable secure NFS by adding the following line to `/etc/sysconfig/nfs`:

`SECURE_NFS=no`

`/etc/sysconfig/nfs` does not exist on Fedora Core 4 and RHEL 4 systems. To use Kerberos 5 or other strong encryption mechanism with NFSv4, you should set this variable to `yes`.
2. Edit `/etc/ldapd.conf` and set the `Domain` option to your domain and change the `Nobody-User` and `Nobody-Group` options to `nobody`:

```
Domain = example.com
[Mapping]
Nobody-User = nobody
Nobody-Group = nobody
```

You might not have to make this change because `ldapd.conf` is usually configured to use the `nobody` user and group by default.

3. Restart the portmapper and NFS using the `service` utility:

```
# service portmap restart
# service nfs condrestart
```

You do not need to start the GSS client and server daemons, `rpcgssd` and `rpcsvcgssd`, respectively, unless you wish to use Kerberos 5 or another strong encryption mechanism (in which case there is additional setup to perform that this chapter does not address).

Once the daemons are running, you can configure your server as described in the next section. You'll learn how to mount the exports in the section titled "Configuring an NFS Client."

Example NFS Server

This section illustrates a simple but representative NFS server configuration. It exports two file systems, `/home` and `/media`. Here are the corresponding entries in `/etc/exports`:

```
/home 192.168.0.0/24(rw,async,no_subtree_check)
/media 192.168.0.0/24(ro)
```

With the exports configured, start (or restart) the daemons (the portmapper is already running) using the initialization scripts:

```
# service nfs start
Starting NFS services:                [ OK ]
Starting NFS quotas:                  [ OK ]
Starting NFS mountd:                  [ OK ]
Starting NFS daemon:                  [ OK ]
# service nfslock start
Starting NFS file locking services:
Starting NFS statd:                   [ OK ]
```

Next, use `rpcinfo -p` to make sure the necessary daemons are running, then finish up with `showmount -a` (or `exportfs -v`) to list the server's NFS exports:

```
# rpcinfo -p
program vers proto  port
 100000   2    tcp    111  portmapper
 100000   2    udp    111  portmapper
 100011   1    udp    958  rquotad
 100011   2    udp    958  rquotad
 100011   1    tcp    961  rquotad
 100011   2    tcp    961  rquotad
 100003   2    udp   2049  nfs
 100003   3    udp   2049  nfs
 100003   4    udp   2049  nfs
 100003   2    tcp   2049  nfs
 100003   3    tcp   2049  nfs
 100003   4    tcp   2049  nfs
 100021   1    udp  37737  nlockmgr
 100021   3    udp  37737  nlockmgr
 100021   4    udp  37737  nlockmgr
 100021   1    tcp  35981  nlockmgr
 100021   3    tcp  35981  nlockmgr
 100021   4    tcp  35981  nlockmgr
 100005   1    udp    974  mountd
 100005   1    tcp    977  mountd
 100005   2    udp    974  mountd
 100005   2    tcp    977  mountd
 100005   3    udp    974  mountd
 100005   3    tcp    977  mountd
# showmount -e
Export list for bubba.example.com:
/home 192.168.0.0/24
/media 192.168.0.0/24
```

The final step in preparing an NFS server is to ensure that NFS services are started at boot time. You can use the Services Configuration Tool (Red Hat ⇨ System Settings ⇨ Server Settings ⇨ Services on Fedora Core and Applications ⇨ System Settings ⇨ Server Settings ⇨ Services on RHEL); `system-config-services` at the command line, or the `chkconfig` command-line services administration tool. Using `chkconfig`, execute the following commands:

```
# chkconfig --level 0123456 nfs off
# chkconfig --level 0123456 nfslock off
# chkconfig --level 345 nfs on
# chkconfig --level 345 nfslock on
```

The first two commands disable the `nfs` and `nfslock` initialization scripts for all run levels. The second two commands reenable them for run levels 3, 4, and 5. After you have confirmed that the NFS daemons are running and that the exports are available, you are ready to configure one or more NFS clients. First, however, for the graphically addicted (or the command-line-challenged), we'll show you how to use Red Hat Linux's graphical tool for administering NFS exports, the NFS Server Configuration Tool.

Using the NFS Server Configuration Tool

If you prefer to use graphical tools for system administration, Red Hat Linux includes the NFS Server Configuration tool. It edits the `/etc/exports` file directly, so you can use the graphical tool and edit the configuration file directly using a text editor interchangeably. To start the NFS Server Configuration tool, select Red Hat ⇨ System Settings ⇨ Server Settings ⇨ NFS on Fedora Core or Applications ⇨ System Settings ⇨ Server Settings ⇨ NFS on RHEL. You can also start the tool by executing the command `system-config-nfs` (as root) in a terminal window. Figure 12-2 shows the NFS Server Configuration tool.

To add a new export, click the Add button, which opens the Add NFS Share dialog box (see Figure 12-3). On the Basic tab, type the name of the directory you want to export in the Directory text box or use the Browse button to locate the directory to export. Use the Host(s) text box to indicate which hosts are allowed to mount this directory. Click the Read-only radio button (selected by default) or the Read/Write radio button to indicate the basic access permissions for this export.

Figure 12-3, for example, shows that `/home` will be exported read-write to all hosts with an IP address in the range `192.168.0.0/24`. Notice that you can use the same syntax for specifying IP addresses in this NFS Server Configuration tool that you can if you edit `/etc/exports` directly.

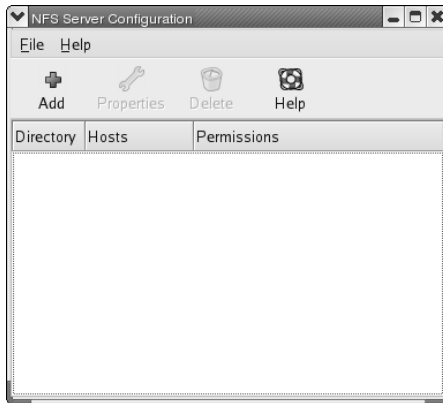


Figure 12-2 The NFS Server Configuration dialog box.

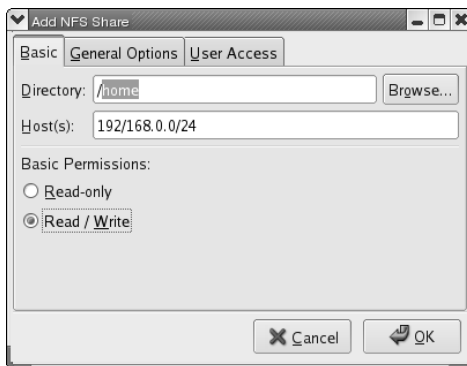


Figure 12-3 The Add NFS Share dialog box.

To modify the mount options for your new NFS export, click the General Options tab. On this tab, click the check boxes to enable the corresponding mount option. The possible mount options include:

- **Allow connections from ports 1024 and higher** — This option corresponds to the `insecure` option listed in Table 12-1.
- **Allow insecure file locking** — This option corresponds to the `insecure_locks` option listed in Table 12-1.
- **Disable subtree checking** — This option corresponds to the `no_subtree_check` option listed in Table 12-1.
- **Sync write operations on request** — This option (enabled by default) corresponds to the `sync` option listed in Table 12-1.
- **Force sync of write operations immediately** — This option is only available if Sync write operations on request is enabled and corresponds to the `no_wdelay` option listed in Table 12-1.

- **Hide filesystems beneath** — This option corresponds to the `hide` option listed in Table 12-1.
- **Export only if mounted** — This option corresponds to the `mp[=path]` option listed in Table 12-1. Selecting this option is equivalent to specify the `mp` mount option with out the optional `path` mount point.
- **Optional mount point** — This option corresponds to the `path` portion of the `mp[=path]` option listed in Table 12-1. You can type the mount point, if you want to specify on, the text box or use the Browse button to select the mount point graphically.
- **Set explicit Filesystem ID** — This option corresponds to the `fsid=n` option listed in Table 12-1. Enter the actual FSID value in the text box.

Figure 12-4 shows the General Options tab. We have disabled subtree checking for `/home` and left the required `sync` option (Sync write operations on request) enabled.

The User Access tab, shown in Figure 12-5, implements the UID/GID remapping and root-squashing options described earlier in this chapter. Select the Treat remote root user as local root user check box if you want the equivalent of `no_root_squash`. To remap all UIDs and GIDs to the UID and GID of the anonymous user (the `all_squash` option from Table 12-1), select the Treat all client users as anonymous users check box. As you might guess, if you want to specify the anonymous UID or GID, click the corresponding check boxes to enable these options and then type the desired value in the matching text boxes. In Figure 12-5, all clients will be remapped to the anonymous user.

Figure 12-5 shows the User Access Tab as it appears in Fedora Core; it looks slightly different in RHEL.

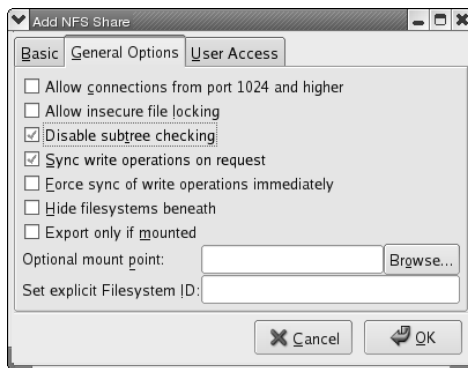


Figure 12-4 The General Options tab.

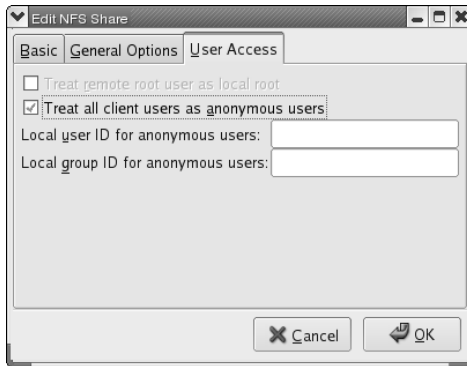


Figure 12-5 The User Access tab.

When you have finished configuring your new NFS export, click the OK button to close the Add NFS Share dialog box. After a short pause, the new NFS share appears in this list of NFS exports, as shown in Figure 12-6. If you want to change the characteristics of an NFS share, select the share you want to modify and click the Properties button on the toolbar. This will open the Edit NFS Share dialog box, which has the same interface as the Add NFS Share dialog box.

Similarly, if you want to remove an NFS share, select the export you want to cancel and click the Delete button. To close the NFS Server Configuration tool, type **Ctrl+Q** or click File → Quit on the menu bar.

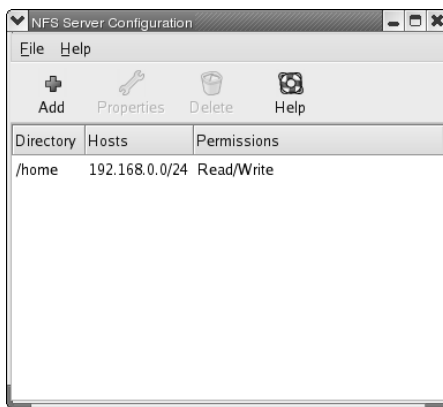


Figure 12-6 Adding an NFS share.

Configuring an NFS Client

Configuring client systems to mount NFS exports is simpler than configuring the NFS server itself. This section of the chapter provides a brief overview of client configuration, identifies the key files and commands involved in configuring and mounting NFS exported file systems, and shows you how to configure a client to access the NFS exports configured in the previous section.

Configuring a client system to use NFS involves making sure that the portmapper and the NFS file locking daemons `statd` and `lockd` are available, adding entries to the client's `/etc/fstab` for the NFS exports, and mounting the exports using the `mount` command.

As explained at the beginning of the chapter, a mounted NFS exported file system is functionally equivalent to a local file system. Thus, as you might expect, you can use the `mount` command at the command line to mount NFS exports manually, just as you might mount a local file system. Similarly, to mount NFS exports at boot time, you just add entries to the file system mount table, `/etc/fstab`. As you will see in the section titled “Using Automount Services” at the end of this chapter, you can even mount NFS file systems automatically when they are first used, without having to mount them manually. The service that provides this feature is called, yup, you guessed it, the *automounter*. More on the automounter in a moment.

As a networked file system, NFS is sensitive to network conditions, so the NFS client daemons accept a few options, passed via the `mount` command, address NFS's sensitivities and peculiarities. Table 12-4 lists the major NFS-specific options that `mount` accepts. For a complete list and discussion of all NFS-specific options, see the NFS manual page (`man nfs`).

Table 12-4 NFS-Specific Mount Options

OPTION	DESCRIPTION
<code>bg</code>	Enables mount attempts to run in the background if the first mount attempt times out (disable with <code>nobg</code>).
<code>fg</code>	Causes mount attempts to run in the foreground if the first mount attempt times out, the default behavior (disable with <code>nofg</code>).
<code>hard</code>	Enables failed NFS file operations to continue retrying after reporting “server not responding” on the system, the default behavior (disable with <code>nohard</code>).
<code>intr</code>	Allow signals (such as <code>Ctrl+C</code>) to interrupt a failed NFS file operation if the file system is mounted with the <code>hard</code> option (disable with <code>nointr</code>). Has no effect unless the <code>hard</code> option is also specified or if <code>soft</code> or <code>nohard</code> is specified.

Table 12-4 (continued)

OPTION	DESCRIPTION
<code>lock</code>	Enables NFS locking and starts the <code>statd</code> and <code>lockd</code> daemons (disable with <code>nolock</code>).
<code>mounthost=name</code>	Sets the name of the server running <code>mountd</code> to <code>name</code> .
<code>mountport=n</code>	Sets the <code>mountd</code> server port to connect to <code>n</code> (no default).
<code>nfsvers=n</code>	Specify the NFS protocol version to use, where <code>n</code> is 1, 2, 3, or 4.
<code>port=n</code>	Sets the NFS server port to which to connect to <code>n</code> (the default is 2049).
<code>posix</code>	Mount the export using POSIX semantics so that the <code>POSIX pathconf</code> command will work properly.
<code>retry=n</code>	Sets the time to retry a mount operation before giving up to <code>n</code> minutes (the default is 10,000).
<code>rsize=n</code>	Sets the NFS read buffer size to <code>n</code> bytes (the default is 1024); for NFSv4, the default value is 8192.
<code>soft</code>	Allows an NFS file operation to fail and terminate (disable with <code>no soft</code>).
<code>tcp</code>	Mount the NFS file system using the TCP protocol (disable with <code>notcp</code>).
<code>timeo=n</code>	Sets the RPC transmission timeout to <code>n</code> tenths of a second (the default is 7). Especially useful with the <code>soft</code> mount option.
<code>udp</code>	Mount the NFS file system using the UDP protocol, the default behavior (disable with <code>noupd</code>).
<code>wsize=n</code>	Sets the NFS write buffer size to <code>n</code> bytes (the default is 1024); for NFSv4, the default value is 8192.

The options you are most likely to use are `rsize`, `wsize`, `hard`, `intr`, and `nolock`. Increasing the default size of the NFS read and write buffers improves NFS's performance. The suggested value is 8192 bytes, that is, `rsize=8192` and `wsize=8192`, but you might find that you get better performance with larger or smaller values. The `nolock` option can also improve performance because it eliminates the overhead of file locking calls, but not all servers support file locking over NFS. If an NFS file operation fails, you can use a keyboard interrupt, usually `Ctrl+C`, to interrupt the operation if the exported file system was mounted with *both* the `intr` and `hard` options. This prevents NFS clients from hanging.

Like an NFS server, an NFS client needs the portmapper running in order to process and route RPC calls and returns from the server to the appropriate port and programs. Accordingly, make sure that the portmapper is running on the client system using the `portmap` initialization script:

```
# service portmap status
```

If the output says `portmap` is stopped (it shouldn't be), start the portmapper:

```
# service portmap start
```

To use NFS file locking, both an NFS server and any NFS clients need to run `statd` and `lockd`. As explained in the section on configuring an NFS server, the simplest way to accomplish this is to use the initialization script, `/etc/rc.d/init.d/nfslock`. Presumably, you have already started `nfslock` on the server, so all that remains is to start it on the client system:

```
# service nfslock start
```

Once you have configured the mount table and started the requisite daemons, all you need to do is mount the file systems. You learned about the `mount` command used to mount file systems in a previous chapter, so this section shows only the `mount` invocations needed to mount NFS file systems. During initial configuration and testing, it is easiest to mount and unmount NFS export at the command line. For example, to mount `/home` from the server configured at the end of the previous section, execute the following command as root:

```
# mount -t nfs bubba:/home /home
```

You can, if you wish, specify client mount options using `mount`'s `-o` argument, as shown in the following example.

```
# mount -t nfs bubba:/home /home -o rsize=8292,wsiz=8192,hard,intr,nolock
```

After satisfying yourself that the configuration works properly, you probably want to mount the exports at boot time. Fortunately, Fedora Core and RHEL make this easy because the initialization script `/etc/rc.d/init.d/netfs`, which runs at boot time, automatically mounts all networked file systems not configured with the `noauto` option, including NFS file systems. It does this by parsing `/etc/fstab` looking for file systems of type `nfs`, `nfs4` (described in the next section), `smbfs` (Samba) `cifs` (Common Internet Filesystem) or `ncpfs` (Netware) and mounting those file systems.

TIP If you are connecting an NFSv4 client to an NFSv2 server, you must use the mount option `nfsvers=2` or the mount attempt will fail. Use `nfsvers=1` if you are connecting to an NFSv1 server. We learned this the hard way while trying to mount an export from an ancient server running Red Hat Linux 6.2 (we told you it was ancient). We kept getting an error indicating the server was down when we knew it wasn't. Finally, we logged into the server, discovered it was running a very old distribution and were able to mount the export. While we're somewhat embarrassed to be running such an old version of Red Hat, we're also quite pleased to report that it has been running so well for so long that we forgot just how old it was.

Configuring an NFSv4 Client

The introduction of NFSv4 into the kernel added some NFSv4-specific behavior of which you need to be aware and changed some of the mount options. This section covers NFSv4-specific features and begins with the mount options that have changed in terms of their meaning or behavior. Table 12-5 lists the new or changed mount options.

The two new options listed in Table 12-5 are `clientaddr` and `proto`. Version 3 of NFS introduced NFS over TCP, which improved NFS's reliability over the older UDP-based implementation. Under NFSv3, you would use the mount option `tcp` or `udp` to specify to the client whether you wanted it to use TCP or UDP to communicate with the server. NFSv4 replaces `tcp` and `udp` with a single option, `proto=` that accepts two arguments: `tcp` or `udp`. In case it isn't clear, the NFSv3 option `tcp` is equivalent to the NFSv4 option `proto=tcp`. Figuring out the `udp` option is left as an exercise for the reader.

Table 12-5 NFSv4-Specific Mount Options

OPTION	DESCRIPTION
<code>clientaddr=n</code>	Causes a client on a multi-homed system to use the IP address specified by <code>n</code> to communicate with an NFSv4 server.
<code>proto=type</code>	Tells the client to use the network protocol specified by <code>type</code> , which can be <code>tcp</code> or <code>udp</code> (the default is <code>udp</code>); this option replaces the <code>tcp</code> and <code>udp</code> options from earlier versions of NFS.
<code>rsiz=n</code>	Sets the read buffer size to <code>n</code> bytes (the default for NFSv4 is 8192); the maximum value is 32678.
<code>sec=mode</code>	Set the security model to <code>mode</code> , which can be <code>sys</code> , <code>krb5</code> , <code>krb5i</code> , or <code>krb5p</code> .
<code>wsiz=n</code>	Sets the write buffer size to <code>n</code> bytes (the default for NFSv4 is 8192); the maximum value is 32678.

The semantics for the `rsiz` and `wsiz` options have changed with NFSv4. The default buffer size is for NFSv4 is 8192 bytes, but it can grow to as large as 32,678 bytes, which should result in a noticeable performance improvement, especially when you are transferring large files. The buffer setting is only a suggestion, however, because the client and server negotiate the buffer size to select an optimal value according to network conditions.

Strictly speaking, the `sec` option for selecting the security model NFS uses isn't new with NFSv4. It existed in NFSv3, but now that NFSv4 has added strong encryption to the core NFS protocol, using this option is worthwhile. As shown in Table 12-5, legal values for the `sec` option are `sys`, `krb5`, `krb5i`, and `krb5p`. `sys`, the default security model, uses standard Linux UIDs and GIDs to authenticate NFS transactions. `krb5` uses Kerberos 5 to authenticate users but takes no special measures to validate NFS transactions; `krb5i` (Kerberos 5 with integrity checking) uses Kerberos 5 to authenticate users and checksums to enforce the data integrity on NFS transactions; `krb5p` (Kerberos 5 with privacy checking) uses Kerberos 5 to authenticate users and encryption to protect NFS transactions against packet sniffing. You can use the various Kerberos-enabled security models only if the NFS server supports *both* NFSv4 *and* the requested security model.

Example NFS Client

The example in this section demonstrates how to mount `/home` and `/usr/local` from the NFS server configured earlier in the chapter.

1. Clients that want to use both exports need to have the following entries in `/etc/fstab`:

```
bubba:/usr/local /usr/local nfs
rsiz=8192,wsiz=8192,hard,intr,nolock 0 0
bubba:/home /home nfs
rsiz=8192,wsiz=8192,hard,intr,nolock 0 0
```

The hostname used on the left side of the colon, `bubba`, must resolve to an IP address either using DNS or an entry in the `/etc/hosts` file. We don't recommend using an IP address because, in a well-run system, IP addresses can change, whereas a hostname won't. If DNS is properly configured and maintained, the hostname will always point to the proper system regardless of what that system's IP address is at any given time.

2. If it isn't already running, start the portmapper using the following command:

```
# service portmap start
Starting portmapper: [ OK ]
```

3. Mount the exports using one of the following commands:

```
# mount -a -t nfs
```

or

```
# mount /home /usr/local
```

or

```
# service netfs start
```

The first command mounts all (-a) file systems of type nfs (-t nfs). The second command mounts only the file systems /home and /usr/local (for this command to work, the file systems you want to mount must be listed in /etc/fstab). The third command uses the service command to mount all network file systems using by invoking the netfs service. Verify that the mounts completed successfully by attempting to access files on each file system. If everything works as designed, you are ready to go.

If all the preceding seems unnecessarily tedious, it only seems that way because it is more involved to *explain* how to set up an NFS client than it is actually to do it. Once you've done it a couple of times, you'll be able to dazzle your friends and impress your coworkers with your wizardly mastery of NFS. You can really wow them after reading the next section, which shows you how to avoid the tedium by using the automounter to mount file systems automatically the first time you use them.

Using Automount Services

The easiest way for client systems to mount NFS exports is to use autofs, which automatically mounts file systems not already mounted when the file system is first accessed. autofs uses the automount daemon to mount and unmount file systems that automount has been configured to control. Although slightly more involved to configure than the other methods for mounting NFS file systems, autofs setup has to be done only once. In the next chapter, you'll even learn how to distribute automounter configuration files from a central server, obviating the need to touch client systems manually at all.

autofs uses a set of map files to control automounting. A master map file, /etc/auto.master, associates mount points with secondary map files. The secondary map files, in turn, control the file systems mounted under the corresponding mount points. For example, consider the following /etc/auto.master autofs configuration file:

```
/home /etc/auto.home
/var /etc/auto.var --timeout 600
```

This file associates the secondary map file `/etc/auto.home` with the mount point `/home` and the map file `/etc/auto.var` with the `/var` mount point. Thus, `/etc/auto.home` defines the file systems mounted under `/home`, and `/etc/auto.var` defines the file systems mounted under `/var`.

Each entry in `/etc/auto.master`, what we'll refer to as the *master map file*, consists of at least two and possibly three fields. The first field is the mount point. The second field identifies the full path to the secondary map file that controls the map point. The third field, which is optional, consists of options that control the behavior of the automount daemon.

In the example master map file, the automount option for the `/var` mount point is `--timeout 600`, which means that after 600 seconds (10 minutes) of inactivity, the `/var` mount point will be unmounted automatically. If a timeout value is not specified, it defaults to 300 seconds (5 minutes).

The secondary map file defines the mount options that apply to file systems mounted under the corresponding directory. Each line in a secondary map file has the general form:

```
localdir [-[options]] remotefs
```

localdir refers to the directory beneath the mount point where the NFS mount will be mounted. *remotefs* specifies the host and pathname of the NFS mount. *remotefs* is specified using the *host:/path/name* format described in the previous section. *options*, if specified, is a comma-separated list of mount options. These options are the same options you would use with the `mount` command.

Given the entry `/home /etc/auto.home` in the master map file, consider the following entries in `/etc/auto.home`:

```
kurt -rw,soft,intr,rsiz=8192,wsiz=8192 luther:/home/kurt
terry luther:/home/terry
```

In the first line, *localdir* is `kurt`, *options* is `-rw,soft,intr,rsiz=8192,wsiz=8192`, and *remotefs* is `luther:/home/kurt`. This means that the NFS export `/home/kurt` on the system named `luther` will be mounted in `/home/kurt` in read-write mode, as a soft mount, with read and write buffer sizes of 8192 bytes. A key point to keep in mind is that if `/home/kurt` exists on the local system, its contents will be temporarily replaced by the contents of the NFS mount `/home/kurt`. In fact, it is probably best if the directory specified by *localdir* does not exist because `autofs` dynamically creates it when it is first accessed.

The second line of the example `auto.home` file specifies *localdir* as `terry`, no *options*, and *remotefs* as the NFS exported directory `/home/terry` exported from the system named `luther`. In this case, then, `/home/terry` on `luther` will be mounted as `/home/terry` on the NFS client using the default NFS

mount options. Again, `/home/terry` should not exist on the local system, but the base directory, `/home`, should exist.

Suppose that you want to use `autofs` to mount a shared projects directory named `/proj` on client systems on the `/projects` mount point. On the NFS server (named `diskbeast` in this case), you would export the `/proj` as described in the section “Configuring an NFS Server.” On each client that will mount this export, create an `/etc/auto.master` file that resembles the following:

```
/projects /etc/auto.projects --timeout 1800
```

This entry tells the automount daemon to consult the secondary map file `/etc/auto.projects` for all mounts located under `/projects`. After 1800 seconds without file system activity in `/projects`, `autofs` will automatically unmount it.

NOTE If the `autofs` RPM is installed, Fedora Core and RHEL systems provide a default `/etc/auto.master` map file. All of the entries are commented out using the `#` sign, so you can edit the existing file if you wish.

Next, create the following `/etc/auto.projects` file on each client that will use `diskbeast`’s export:

```
code -rw,soft,rsize=8192,wsiz=8192 diskbeast:/proj
```

This entry mounts `/proj` from `mailbeast` as `/projects/code` on the client system. The mount options indicate that the directory will be read/write, that it will be a soft mount, and that the read and write block sizes are 8192 bytes. Recall from Table 12-4 that a soft mount means that the kernel can time out the mount operation after a period of time specified by the `timeo=n` option, where `n` is defined in tenths of a second.

Finally, as the root user, start the `autofs` service:

```
# /sbin/service autofs start
Starting automount: [ OK ]
```

After starting the `autofs` service, you can use the `status` option to verify that the automount daemon is working:

```
# /sbin/service autofs status
Configured Mount Points:
-----
/usr/sbin/automount --timeout 600 /projects file /etc/auto.projects

Active Mount Points:
-----
/usr/sbin/automount --timeout 600 /projects file /etc/auto.projects
```


As you can see under the heading Active Mount Points, the `/projects` mount point is active. You can verify this by changing to the `/projects/code` directory and executing an `ls` command:

```
# cd /projects/code
# ls
3c501.c          atp.c           fmv18x.c        net_init.c       smc9194.c
3c503.c          au1000_eth.c    gmac.c          ni5010.c         smc-mca.c
3c505.c          auto_irq.c      gt96100eth.c    ni52.c           smc-
ultra32.c
3c507.c          bagetlance.c    hamachi.c       ni65.c           smc-ultra.c
3c509.c          bmac.c          hp100.c         ns83820.c        sonic.c
```

You can also see the automount daemon at work by using the `mount` command:

```
# mount -t autofs
automount(pid11081) on /projects type autofs
(rw,fd=4,pgrp=11081,minproto=2,maxproto=4)
# mount -t nfs
diskbeast:/proj on /projects/code type nfs
(rw,soft,rsz=8192,wsz=8192,nfsvers=2,addr=192.168.0.1)
```

Using `mount`'s `-t` option limits the output to file systems of the specified type, `autofs` for automounted file systems, and `nfs` for NFS file systems.

The first output line shows that automount is managing the `/projects` file system; the second line shows that the automount-managed file system has mounted the NFS file system `diskbeast:/proj` on `/projects` using the options specified in `/etc/auto.projects`.

To stop the automounter, use the service script's `stop` argument:

```
# /sbin/service autofs stop
Stopping automount: [ OK ]
```

One of the handiest features of the `autofs` service is that changes made to the secondary map files go into effect almost immediately. The next time that a directory or file system managed by `autofs` is accessed, the automounter rereads the secondary map files. So, changes to the secondary map files do not require any special treatment. However, if you modify the master map file, you have to reload the configuration file using the following command:

```
/sbin/service autofs reload
```

Examining NFS Security

As explained at the beginning of the chapter, NFS protocol versions 3 and older have some inherent security problems that make it unsuitable for use across the Internet and potentially unsafe for use even in a trusted network. This section identifies key security issues of NFS in general and the security risks specific to an NFS server and to NFS clients and suggests remedies that minimize your network's exposure to these security risks. Be forewarned, however, that *no* list of security tips, however comprehensive, makes your site completely secure. Nor will plugging possible NFS security holes address other potential exploits.

General NFS Security Issues

One NFS weakness, in general terms, is the `/etc/exports` file. If a cracker is able to spoof or take over a *trusted address*, an address listed in `/etc/exports`, your exported NFS mounts are accessible. Another NFS weak spot is normal Linux file system access controls that take over once a client has mounted an NFS export: Once an NFS export has been mounted, normal user and group permissions on the files take over access control.

The first line of defense against these two weaknesses is to use host access control as described earlier in the chapter to limit access to services on your system, particularly the portmapper, which has long been a target of exploit attempts. Similarly, you should add entries in `/etc/hosts.deny` `lockd`, `statd`, `mountd`, and `rquotad`.

More generally, judicious use of IP packet firewalls, using `netfilter`, dramatically increases NFS server security. `netfilter` is stronger than NFS daemon-level security or even TCP Wrappers because it restricts access to your server at the packet level. Although `netfilter` is described in detail in Chapter 34, this section gives you a few tips on how to configure a `netfilter` firewall that plays nicely with NFS.

First, you need to know the ports and services NFS uses so that you know where to apply the packet filters. Table 12-6 lists the ports and protocols each NFS daemon (on both the client and server side) use.

Table 12-6 NFS Ports and Network Protocols

SERVICE	PORT	PROTOCOLS
portmap	111	TCP, UDP
nfsd	2049	TCP, UDP
mountd	variable	TCP, UDP

(continued)

Table 12-6 (continued)

SERVICE	PORT	PROTOCOLS
lockd	variable	TCP, UDP
statd	variable	TCP, UDP
rquotad	variable	UDP

NOTE Before NFSv4, NFS over TCP was experimental on the server side, so most administrators used UDP on the server. However, TCP is quite stable on NFS clients. Nevertheless, using packet filters for both protocols on both the client and the server does no harm. NFSv4's server-side TCP code is much more stable than NFSv3, so it is safe for deployment in a production environment.

Note that `mountd`, `lockd`, `statd`, and `rquotad` do not bind to any specific port; that is, they use a port number assigned randomly by the portmapper (which is one of portmapper's purposes in the first place). The best way to address this variability is to assign each daemon a specific port using the portmapper's `-p` option and then to apply the packet filter to that port.

Regardless of how you configure your firewall, you must have the following rule:

```
iptables -A INPUT -f -j ACCEPT
```

This rule accepts all packet fragments except the first one (which is treated as a normal packet) because NFS does not work correctly unless you let fragmented packets through the firewall. Be sure to read Chapter 34 carefully to configure your NFS server's firewall properly.

Server Security Considerations

On the server, always use the `root_squash` option in `/etc/exports`. NFS helps you in this regard because root squashing is the default, so you should not disable it (with `no_root_squash`) unless you have an extremely compelling reason to do so, such as needing to provide boot files to diskless clients. With root squashing in place, the server substitutes the UID of the anonymous user for root's UID/GID (0), meaning that a *client's* root account cannot change files that only the *server's* root account can change.

The implication of root squashing might be unclear, so we'll make it explicit: all critical binaries and files should be owned by *root*, not *bin*, *wheel*, *adm* or another non-root account. The only account that an NFS client's root user cannot access is the server's root account, so critical files owned by root are much less exposed than if they are owned by other accounts.

It gets better, though. Consider the situation in which a user has root access on a system. In this case, exporting file systems using the `all_squash` option might be worth considering. A user with root access on a client can usually `su` to any user, and that UID will be used over NFS. Without `all_squash`, a compromised client can at least view and, if the file system is mounted read-write, update files owned by any user besides root if `root_squash` is enabled. This security hole is closed if the `all_squash` option is used.

NFS also helps you maintain a secure server through the `secure` mount option; because this mount option is one of the default options `mountd` applies to all exports unless explicitly disabled using the `insecure` option. Ports 1–1024 are reserved for root’s use; merely mortal user accounts cannot bind these ports. Thus, ports 1–1024 are sometimes referred to as *privileged* or *secure ports*. The `secure` option prevents a malevolent nonroot user from initiating a spoofed NFS dialog on an unprivileged port and using it as a launch point for exploit attempts.

Client Security Considerations

On the client, disable SUID (set UID) root programs on NFS mounts using the `nosuid` option. The `nosuid` mount option prevents a server’s root account from creating an SUID root program on an exported file system, logging in to the client as a normal user, and then using the UID root program to become root on the client. In some cases, you might also disable binaries on mounted file systems using the `noexec` option, but this effort almost always proves to be impractical or even counterproductive because one of the benefits of NFS is sharing file systems, such as `/usr` or `/usr/local`, that contain scripts or programs that need to be executed.

TIP You might not want to use `nosuid` if you are sharing system binary directories, because many things in `/bin` and `/usr/bin` will break if they are not SUID.

NFS versions 3 and 4 support NFS file locking. Accordingly, NFS clients must run `statd` and `lockd` in order for NFS file locks to function correctly. `statd` and `lockd`, in turn, depend on the portmapper, so consider applying the same precautions for `portmap`, `statd`, and `lockd` on NFS clients that were suggested for the NFS server.

In summary, using TCP wrappers, the `secure`, `root_squash`, and `nosuid` options, and sturdy packet filters can increase the overall security of your NFS setup. However, NFS is a complex, nontrivial subsystem, so it is entirely conceivable that new bugs and exploits will be discovered.

Summary

In this chapter, you learned to configure NFS, the Network File System. First, you found a general overview of NFS, its typical uses, and its advantages and disadvantages. Next, you found out how to configure an NFS server, you identified key files and commands to use, and you saw the process with a typical real-world example. With the server configured and functioning, you then learned how to configure a client system to access NFS exported file systems, again using key configuration files and commands and simulating the procedure with a representative example. You also learned how to address NFS performance problems and how to troubleshoot some common NFS errors. The chapter's final section identified potential security problems with NFS and suggested ways to mitigate the threat.

The Network Information System

IN THIS CHAPTER

- Understanding NIS
- Planning an NIS Installation
- Configuring an NIS Server
- Configuring an NIS Client
- Using NIS and NFS Together

A common challenge facing administrators charged with maintaining a network of Linux machines is sharing information across the network while maintaining that information centrally. The Network Information Service (NIS) is one solution to such a challenge. This chapter describes NIS and explains how to configure an NIS server and an NIS client. You'll also learn how to integrate NIS and NFS, which can significantly simplify administering a large or geographically dispersed network.

Understanding NIS

NIS distributes information that needs to be shared throughout a Linux network to all machines that participate in the NIS domain. Originally developed by Sun Microsystems, NIS was first known as Yellow Pages (YP), so many NIS-related commands begin with the letters `yp`, such as `ypserv`, `ypbind`, and `yppasswd`. Unfortunately for Sun, the phrase “Yellow Pages” was (and is) a registered trademark of British Telecom in the United Kingdom, so Sun changed the name of their Yellow Pages services to Network Information Service. Despite the name change, however, the NIS suite of utilities retained the `yp` prefixes because administrators had become accustomed to them.

The information most commonly shared using NIS consists of user authentication information, such as `/etc/passwd` and `/etc/group`. If users' password entries are accessible to all login hosts via NIS, any user can log in on any login host on the network that is running an NIS client. However, sharing authentication information is far from the only use for NIS; any information that needs to be distributed across a network and that can or should be centrally administered is a viable candidate for sharing via NIS. For instance, you can use NIS to distribute a company telephone directory or a listing of accounting codes. One of the examples in this chapter shows you how to distribute NFS automounter configuration files using NIS, which eliminates the need to edit automount files individually on each NFS client system.

Like NFS, NIS uses a standard client-server architecture that can be arrayed in one of several possible configurations. Each NIS domain must have at least one NIS server. An *NIS server* is a centrally administered repository for information shared across the network using NIS. *NIS clients* are programs that use NIS to query designated servers for information that is stored in the servers' databases, which are known as *maps*. NIS maps are stored in DBM format, a binary file format derived from simple ASCII text files. For example, the files `/etc/passwd` and `/etc/group` can be converted directly to DBM databases using an ASCII-to-DBM conversion program named `makedbm`.

NOTE Do not be confused by the use of the word *database*. As used in this chapter, *database* refers to a centralized store of information, not to refer to relational database management systems (RDBMSs) such as Oracle, PostgreSQL, or MySQL.

NIS servers can be further subdivided into master and slave servers. A *master server* maintains the authoritative copies of the NIS maps. A *slave server* maintains copies of the maps, which it receives from the master. If the maps on the master server change, the slaves receive updated copies. Slave servers receive copies of the DBM databases, not the ASCII source files. The `yppush` program notifies slave servers of changes to the NIS maps, and then the slaves automatically retrieve the updated maps in order to synchronize their databases with the master. The purpose of slave servers is to provide redundancy. On a busy network, slave servers can reduce the load on the master server. More importantly, if the master server becomes unavailable for some reason, slave servers can function as backup servers until the master is again available.

NIS revolves around the notion of a domain. An *NIS domain* is a unique name that refers to any group of systems that use the same NIS maps. NIS domains function as system management tools providing a convenient method to organize groups of systems that need to access the same information set into

a logical unit. NIS does not impose any physical restrictions on the make-up of a domain. While an NIS domain *might* consist of hosts on a single subnet or contain all of the systems in a single office or building, it doesn't necessarily need to. At one place where Kurt worked, an NIS domain for the engineering group included hosts in both the United States and Germany.

Likewise, do not confuse an NIS domain with an Internet or DNS domain. A DNS name (more specifically, a fully qualified domain name, or FQDN) is the official name that uniquely identifies a system to the Internet domain name system. In fact, although doing so is common practice, most NIS experts recommend not naming an NIS domain with the same name used in a DNS name or FQDN because such a naming convention is potentially confusing and makes it trivially easy for crackers to guess the name of your NIS domain. So, if your domain name is `possumholler.com`, avoid the temptation to name your NIS domain `possumholler`.

Before you proceed, make sure you have the NIS programs installed. For a complete installation, you need the following three packages:

- `ypbind`
- `ypserv`
- `yp-tools`

Planning an NIS Installation

Four NIS topologies are commonly used:

- A single domain with a master server, no slave servers, and one or more clients. (See Figure 13-1.)
- A single domain with a master server, one or more slave servers, and one or more clients. (See Figure 13-2.)
- Multiple domains, each with its own master server, no slave servers, and one or more clients. (See Figure 13-3.)
- Multiple domains, each with its own master server, one or more slave servers, and one or more clients. (See Figure 13-4.)

The single domain configurations are the most widely used. Figure 13-1 illustrates a typical single-domain/single-server configuration. The NIS domain name is `admin`. A single master server, `admin-master`, responds to all queries from NIS clients (`client-1`, `client-2`, and `client-3`) and is the sole source of information for the domain.

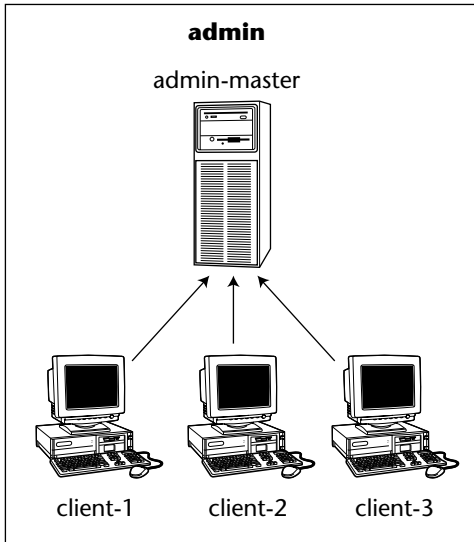


Figure 13-1 A single-domain/single-server NIS configuration.

Figure 13-2 shows the single-domain/multiple-server configuration. The admin domain shown in Figure 13-1 has a slave server, `admin-slave`, in addition to the master server, `admin-master`. In the modified configuration, `client-1` and `client-2` continue to query the master server, but `client-3` communicates with the slave server when performing NIS queries. `client-3` has not been configured specifically to communicate with `admin-slave`. Rather, `client-3` sends out NIS broadcast messages to any listening server in its domain and accepts replies from any server authoritative for that domain — the server that “wins” is the server that replies first, whether it is a master or a slave.

At large sites or in complicated networks, you might find it necessary to create multiple NIS domains. Figures 13-3 and 13-4 illustrate such configurations. Figure 13-3 shows two domains, `admin` and `devel`, each with its own master server, `admin-master` and `devel-master`. Clients in the `admin` domain (`client-1`, `client-2`, and `client-3`) communicate only with the `admin-master` server, and clients in the `devel` domain (`client-4`, `client-5`, and `client-6`) communicate only with `devel-master`.

Figure 13-4 illustrates the same setup as Figure 13-3, except that each domain has a slave server, `admin-slave` and `devel-slave`, and some of the clients in each domain communicate with the slave servers rather than with the master. As in the single-server example, any given client communicates with the server for its domain that responds the fastest to a broadcast query.

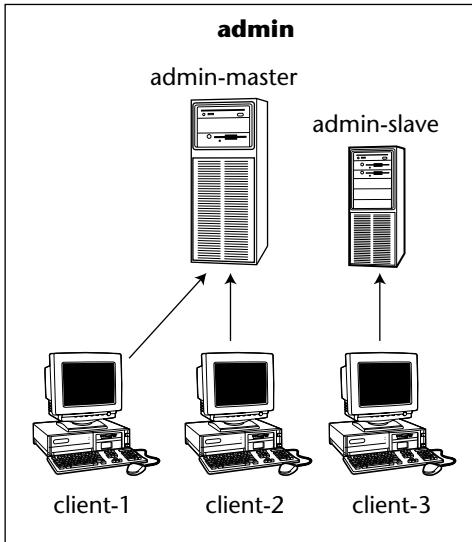


Figure 13-2 A single-domain/multiple-server NIS configuration.

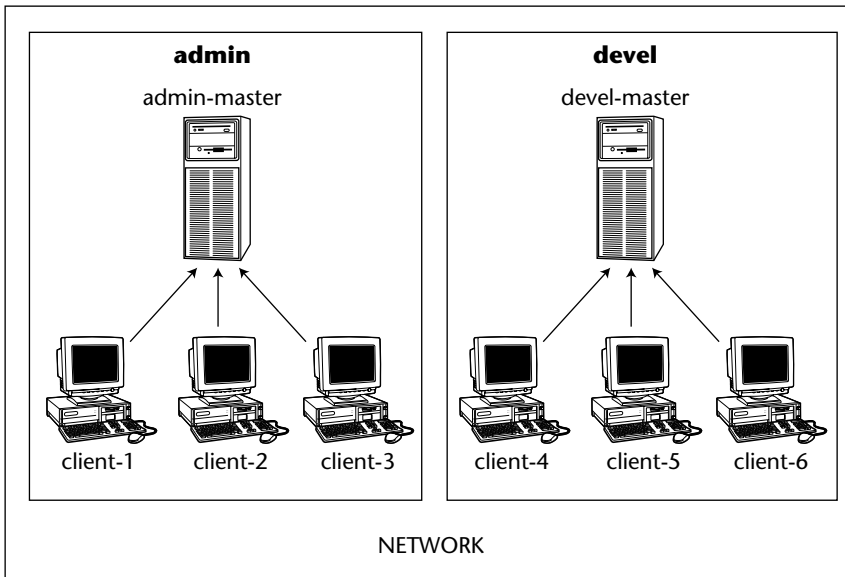


Figure 13-3 The multiple-domain/single-server NIS configuration.

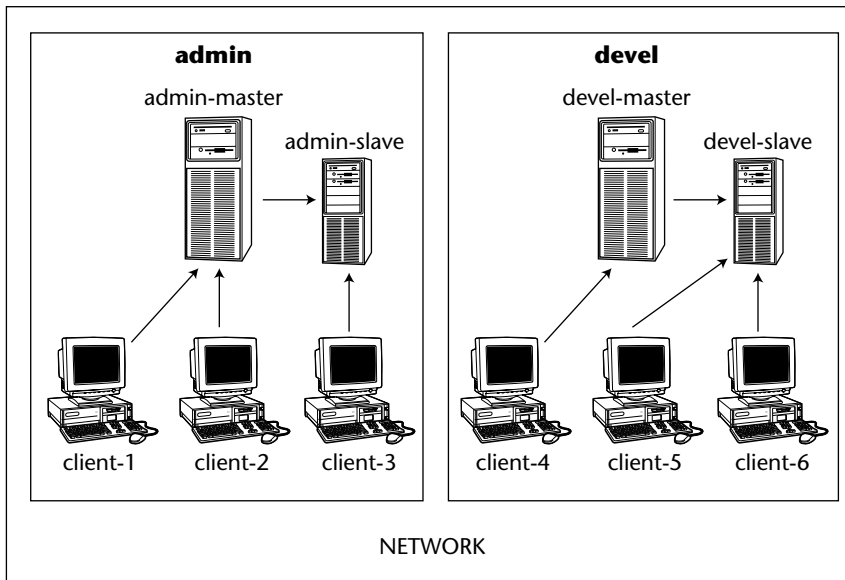


Figure 13-4 The multiple-domain/multiple-server NIS configuration.

CAUTION A singleton server (one whose function is not duplicated or replicated elsewhere in the network) that relies upon NIS for key data potentially represents a single point of failure. If your network or organization relies on the high availability of your network, NIS might not be an acceptable solution for information sharing unless you configure at least one slave server to provide redundancy and fail-over support for the master server.

A complete NIS setup involves configuring at least one NIS server and one or more NIS clients. If your Linux system is going to be part of a network with existing NIS servers, you only need to install and configure an NIS client programs, `ybind`, `ywhich`, `yycat`, `yypoll`, and `yymatch`. The most important program on an NIS client is the NIS client daemon, `ybind`. `ybind` is usually started from the system's startup procedure. As soon as `ybind` is running your system has become an NIS client.

On the other hand, if your system is going to be part of a network that does not already have NIS servers in place, you need to configure a master server and possibly one or more slave servers in addition to any NIS clients. Creating an NIS server involves configuring the `yyserv` client daemon and identifying the information that you want to distribute using NIS.

Configuring an NIS Server

The simplest NIS configuration consists of a single NIS server and one or more clients. In this case, NIS server configuration involves the following steps:

1. Setting the NIS domain name.
2. Initializing the NIS maps.
3. Editing the configuration files.
4. Starting the server daemon, `ypserv`.
5. Starting the NIS password daemon.
6. Starting the NIS transfer daemon if you use slave servers.
7. Modifying the startup process to start the NIS daemons when the system reboots.

If your NIS configuration also utilizes slave servers, you need to perform configuration steps on the slave servers. This section shows you how to create an NIS master server and a slave server.

NOTE For more information about NIS configuration, see the NIS How-To at the **Linux Documentation Project**, linuxdoc.org/HOWTO/NIS-HOWTO/index.html, and the NIS Web pages at www.linux-nis.org.

Key Files and Commands

Table 13-1 lists the commands, daemons, and configuration files used to configure and maintain an NIS server.

Table 13-1 NIS Server Configuration Commands and Files

COMMAND	DESCRIPTION
<code>/etc/ypserv.conf</code>	Stores runtime configuration options and special host access directives
<code>nisdomainname</code>	Sets a system's NIS domain name
<code>/var/yp/securenets</code>	Lists hosts permitted to access the NIS maps
<code>ypinit</code>	Builds and installs the NIS databases
<code>yppasswdd</code>	Processes user password changes in an NIS environment

(continued)

Table 13-1 (continued)

COMMAND	DESCRIPTION
<code>yppush</code>	Propagates updated NIS maps to slave servers
<code>ypserv</code>	Handles the primary NIS server duties
<code>ypxfrd</code>	Speeds up the transfer of large NIS maps from master to slave servers

The initial step in configuring an NIS client is to set the NIS domain name. When first configuring the NIS server and for testing purposes, the quickest way to set an NIS domain name is to use the `nisdomainname` command:

```
# nisdomainname nisdomain
```

Replace `nisdomain` with the name of your NIS domain. Next, reissue the `nisdomainname` command with no arguments to confirm that the NIS domain name was successfully set. Setting the NIS domain name in this way is not a permanent change and will not survive a system reboot. You learn later in this section how to set the NIS domain name permanently.

NOTE You can also use the `domainname` command to get and set a system's NIS domain name. In fact, you can use one of a number of similarly named commands to do so. See the `domainname` man page for more information.

After you set the NIS domain name, configure the NIS server daemon, `ypserv`. The key configuration files are `/var/yp/securenets` and `/etc/ypserv.conf`. `/var/yp/securenets` lists the hosts permitted to access the NIS maps on this server. `/etc/ypserv.conf` contains runtime configuration options and client access specifications that control `ypserv` and the NIS transfer daemon, `ypxfrd`.

The most important configuration file is `/var/yp/securenets`. As a rule, RPC, on which NIS is based, happily replies to any client that asks for information. Obviously, you don't want to share your password database, just for example, with any host that asks for it. So, `securenets` makes it possible to restrict access to NIS maps based on a requester's IP address. The `securenets` file contains net mask and network number pairs that define the lists of hosts permitted to access your NIS server maps. `/var/yp/securenets` contains one pair per line of the form `m.m.m.m n.n.n.n`, where `m.m.m.m` is a net mask, and `n.n.n.n` is network number. A host match occurs when the IP address matches the network number and mask. For example, consider a `/var/yp/securenets` with these entries:

```
255.255.255.255 127.0.0.1
255.255.255.0 192.168.0.0
```

The first line indicates that localhost (IP address 127.0.0.1) is permitted to access the NIS server. The second line specifies that any host with an IP address in the range 192.168.0.1 to 192.168.0.254 is permitted to access the NIS server. All other hosts are denied access.

The second configuration file, `/etc/ypserv.conf`, is used by both `ypserv`, the primary NIS server daemon, and `ypxfrd`, the NIS transfer daemon. `ypserv.conf` contains two types of runtime configuration directives. The first type of directive is known as an *option line* and is only used by `ypserv`. The second configuration directive type is called an *access rule*. Access rules, used by `ypserv` and `ypxfrd`, determine which hosts may use NIS services and under what conditions. The following listing shows the default values in `/etc/ypserv.conf`:

```
dns: no
files: 30
slp: no
slp_timeout: 3600
xfr_check_port: yes
*: *: shadow.byname: port
*: *: passwd.adjunct.byname: port
```

Entries in the file appear one per line. Option lines and access rules are made up of colon-separated fields. The first five entries in the example are option lines. The last two entries are access rules.

Option lines have the following format:

```
option:value
```

option can be one of `files`, `trusted_master`, `slp`, `slp_timeout`, or `xfr_check_port`.

- **files:n** — Sets the number of maps that `ypserv` should cache. Setting *n* to 0 disables map caching.
- **trusted_master:server** — Informs a slave server of the name of the server to accept as master. For instance, `trusted_master:nisbeast.example.com` tells the slave server to accept new or updated maps from the master NIS server `nisbeast.example.com`. By default, no trusted master server is set.
- **slp:{yes|no|domain}** — Indicates whether `ypserv` should use the Service Location Protocol (SLP) and register itself as an SLP server. The default is `no`. If `domain` is set, `ypserv` registers itself as an NIS domain server for a specific group of NIS domains. The sidebar “A Hundred-Word Tour of SLP” describes the Service Location Protocol.

- **slp_timeout** — Defines the interval after which `ypserv` reregisters itself as an SLP server. This option is currently ignored.
- **xfr_check_port** — Controls whether `ypserv` runs on a port numbered less than 1024, a so-called privileged port. The default is `yes`.

As you can see in the default configuration file, the `dns` option is `no`. The absence of `xfr_check_port` from the configuration file means that `ypserv` uses a privileged port.

Access rules have a slightly more complicated format:

```
host:domain:map:security
```

- **host** — Specifies the IP address to match. Wildcards and address/net mask address forms are allowed. For example, the *host* values `192.168.0.` and `192.168.0.0/255.255.255.0` both refer to all IP addresses between `192.168.0.1` and `192.168.0.254`.
- **domain** — Indicates the NIS domain to which the rule applies.
- **map** — Identifies the name of a map to match (or `*` for all maps).
- **security** — Indicates the type of security to use. Legal values are `none`, `port`, or `deny`. A value of `none` enables access to *map* for *host*. `port` enables access if the connection comes from a privileged port (one numbered less than 1024). `deny` denies the matching host access to this map.

Access rules are tried in order, and all rules are evaluated. If no rule matches a connecting host, access to the corresponding map is enabled.

As usual with any RPC-based service, before starting the server, make sure that the portmapper daemon, `portmap`, is running. NIS requires the portmapper because NIS uses remote procedure calls (RPC). To see if the portmapper is running, you can use the portmapper's initialization script, `/etc/rc.d/init.d/portmap`, or the `rpcinfo` command. If the portmapper is not running, you can easily start it. Using the initialization script, the command to execute and its output is:

```
# service portmap status
portmap (pid 559) is running...
```

The output shows the process ID (PID) of the portmapper. On the other hand, if the portmapper is not running, the output of the command looks like the following:

```
# service portmap status
portmap is stopped
```

A HUNDRED-WORD TOUR OF SLP

SLP, the Service Location Protocol, provides a mechanism for networked applications to discover the presence, runtime behavior, and location of services available on a network (think Windows' Network Neighborhood). The implementation used on Linux systems is OpenSLP, available on the Web at www.openslp.org. Ordinarily, to find and use network services, such as networked printers, one has to provide an address, name, or other parameters to locate and use the service. SLP eliminates this necessity by providing a query-response method to find out what services are available, the server(s) providing them, and the configuration details necessary for clients to access them.

To start the portmapper, execute the following command:

```
# service portmap start
Starting portmapper: [ OK ]
```

You can also use the `rpcinfo` command shown next to see if the portmapper is running:

```
# rpcinfo -u localhost portmapper
program 100000 version 2 ready and waiting
# rpcinfo -t localhost portmapper
program 100000 version 2 ready and waiting
```

The first command uses `rpcinfo`'s `-u` option to see if portmapper is listening on a UDP port on the host named `localhost`. The second command uses the `-t` option to perform the same query on for a TCP port. If you do not see output indicating that the portmapper is running, use the initialization script shown previously to start the portmapper. Once you have the portmapper started, start the NIS server using the command:

```
# service ypserv start
Starting YP server services: [ OK ]
```

Next, use the following `rpcinfo` invocation to make sure that the server is running:

```
# rpcinfo -u localhost ypserv
program 100004 version 1 ready and waiting
program 100004 version 2 ready and waiting
```

After the NIS server is running, you have to create something for it to serve, which means you need to create the NIS databases on the machine acting as the NIS server. The command for doing so is `ypinit`. `ypinit` builds a complete

set of NIS maps for your system and places them in a subdirectory of `/var/yp` named after the NIS domain. As suggested earlier in this chapter, you should have only one master server per NIS domain. All databases are built from scratch, either from information available to the program at runtime or from the ASCII database files in `/etc`. These files include (but aren't limited to):

- `/etc/passwd`
- `/etc/group`
- `/etc/hosts`
- `/etc/networks`
- `/etc/services`
- `/etc/protocols`
- `/etc/netgroup`
- `/etc/rpc`

CAUTION Long-time NIS users might recall needing to add the NIS entry `+:0:0::` to the bottom of `/etc/passwd` to indicate the end of local entries. However, this entry spreads an opening for an NIS-based security exploit to all of your NIS clients. For obvious reasons, we won't describe the nature of the exploit here, but you can read all about it the SecurityFocus Web site at www.securityfocus.com.

To create the NIS databases, execute the command `/usr/lib/yp/ypinit -m`.

```
# /usr/lib/yp/ypinit -m
```

The `ypinit` command uses the `-m` option to indicate that it is creating maps for the master server.

TIP Before initializing the NIS databases, you might want to make backup copies of the source text files shown in the preceding list.

If you also configure one or more slave NIS servers, you need to make sure that they can successfully communicate with the master server. From each slave server, make sure that the command `ypwhich -m` works, which means that the slave servers must also be configured as NIS clients, as described in the section “Configuring an NIS Client,” later in this chapter. After configuring the slave server as described in that section, execute the command:

```
# /usr/lib/yp/ypinit -s master
```

The `-s` option instructs `ypinit` to create a slave server using the databases from the master server specified by `master`. An NIS database on a slave server is set up by copying an existing database from a running server. `master` can also be a server on which the NIS maps are current and stable. Your NIS server is now up and running.

Starting the NIS Password Daemon

Because NIS is used to share authentication information, a method must exist for users to change this information on the NIS server and then to propagate the updated information out to other NIS clients and any slave servers that might be present. Similarly, when new users are added or existing users are deleted, clients and slaves must be notified of the change. The daemon that handles this requirement is `ypasswdd`. `ypasswdd` handles password changes and updating other NIS information that depends on user passwords. `ypasswdd` runs only on the NIS master server. Starting the NIS password daemon is a simple matter of executing its initialization script with the `start` argument, as shown in the following example:

```
# service ypasswdd start
Starting YP passwd service: [ OK ]
```

Keep in mind that NIS users are not permitted, by default, to change their full name or their login shell. You can enable NIS users to change their login information by starting `ypasswdd` using the `-e chfn` option to allow name changes or the `-e chsh` option to allow login shell changes. You can make these changes by editing `/etc/sysconfig/ypasswdd` and modifying the line that begins `YPPASSWDD_ARGS=` so that it looks like the following:

```
YPPASSWDD_ARGS="-e chfn -e chsh"
```

Starting the Server Transfer Daemon

`ypxfrd`, which you need to use only if you run NIS slave servers, speeds up the transfer of large NIS maps from master to slave servers. Ordinarily, when a slave server receives a message from a master server that there is a new map, the slave starts `ypxfr` to transfer the new map. `ypxfr` reads the contents of a map from the master server one entry at a time. The transfer daemon, `ypxfrd`, speeds up the transfer process by enabling slave servers to copy the master's maps rather than building their own from scratch. As with the password daemon, you run the transfer daemon on the master server only. To start the transfer daemon, execute the command:

```
# /sbin/service ypxfrd start
```

If you need to update a map, run `make` in the `/var/yp` directory on the NIS master. This command updates a map if the source file is newer and propagates the new map out to the slave servers, if present.

Starting the NIS Servers at Boot Time

After you have configured your NIS server, you should make the system changes persistent, which means permanently storing the NIS domain name in the network configuration and ensuring that the required daemons (`ypserv`, `yppasswdd`, and, if you use slave servers, `ypxfrd`) start and stop when the system starts and stops.

The first step is permanently saving the NIS domain name. To do this, add the following line to `/etc/sysconfig/network`:

```
NISDOMAIN=nisdomainname
```

Replace `nisdomainname` with the NIS domain name you selected when you initially configured your server. The networking initialization scripts read `/etc/sysconfig/network` at boot time and pick up the domain name to use.

The next step is to use `chkconfig` to enable services at boot time. To start the `ypserv` and `yppasswdd` services, for example, use the following commands:

```
# chkconfig --levels 0123456 ypserv off
# chkconfig --levels 345 ypserv on
# chkconfig --levels 0123456 yppasswdd off
# chkconfig --levels 345 yppasswdd off
```

If you prefer graphical tools, use the Service Configuration tool. To do so, type **system-config-services** (as root) in a terminal window or select Red Hat ⇨ System Settings ⇨ Server Settings ⇨ Services. The resulting screen should resemble Figure 13-5.

Scroll down to the bottom of the list, and place check marks in the boxes next to the services you want to start at boot time. In Figure 13-6, for example, you can see that the `yppasswdd` service has been checked, meaning that it will start at boot time. The erstwhile administrator has enabled `ypserv`, too. Click the Save button to save your changes, and then select File ⇨ Exit to close Service Configuration tool.

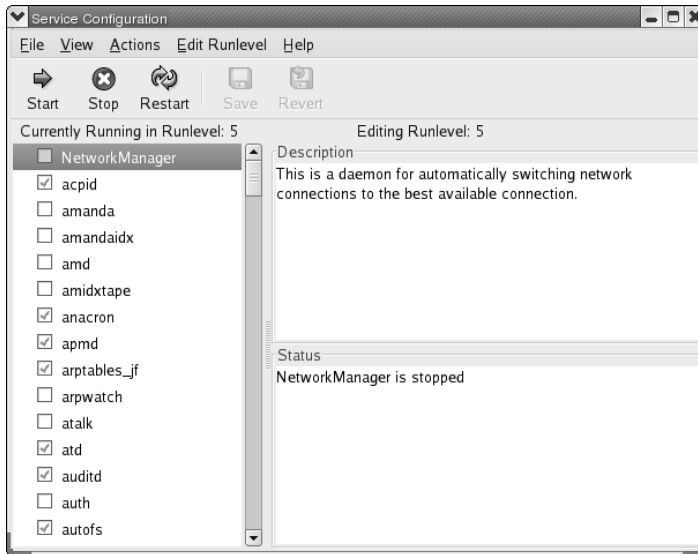


Figure 13-5 The Service Configuration tool.

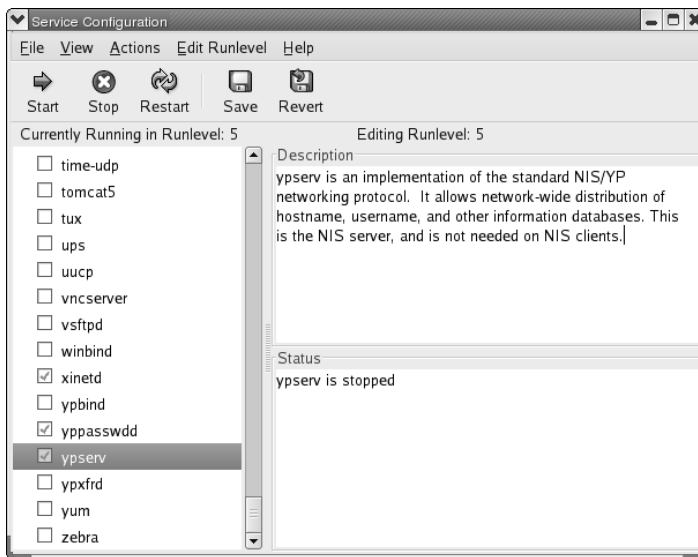


Figure 13-6 Enabling NIS at boot time.

Configuring an Example NIS Server

This section illustrates the process of setting up a simple master server. The NIS domain name is `eng`, running on the server `nisbeast.example.com`, which has an IP address `192.168.0.4`. There are no slave servers, and all hosts in the `example.com` DNS domain are permitted to access the NIS server.

1. Set the NIS domain name:

```
# nisdomainname eng
# nisdomainname
eng
```

2. Edit `/var/yp/securenets` to permit access to the NIS server for the specified hosts. The default configuration enables all hosts to have access (`0.0.0.0 0.0.0.0`), so change that line to read **255.255.255.0 192.168.0.0**. The complete file now resembles the following:

```
255.255.255.255    127.0.0.1
255.255.255.0      192.168.0.0
```

NOTE If `/var/yp/securenets` does not exist on your system, create it.

3. Make sure that the portmapper is running:

```
# rpcinfo -u nisbeast portmapper
program 100000 version 2 ready and waiting
```

4. Start the primary server daemon, `ypserv`:

```
# service ypserv start
Starting YP server services:                [ OK ]
```

5. Confirm that `ypserv` is running:

```
# rpcinfo -u nisbeast ypserv
program 100004 version 1 ready and waiting
program 100004 version 2 ready and waiting
```

6. Initialize the NIS maps:

```
# /usr/lib/yp/ypinit -m
At this point, we have to construct a list of the hosts which will
run NIS
servers.  nisbeast.kurtwerks.com is in the list of NIS server hosts.
Please continue to add
the names for the other hosts, one per line.  When you are done with
the
list, type a <control D>.
    next host to add:  nisbeast.kurtwerks.com
    next host to add:
The current list of NIS servers looks like this:
```

```
nisbeast.kurtwerks.com
```

```
Is this correct? [y/n: y] y
We need a few minutes to build the databases...
Building /var/yp/eng/ypservers...
Running /var/yp/Makefile...
gmake[1]: Entering directory `/var/yp/eng'
Updating passwd.byname...
Updating passwd.byuid...
Updating group.byname...
Updating group.bygid...
Updating hosts.byname...
Updating hosts.byaddr...
Updating rpc.byname...
Updating rpc.bynumber...
Updating services.byname...
Updating services.byservicename...
Updating netid.byname...
Updating protocols.bynumber...
Updating protocols.byname...
Updating mail.aliases...
gmake[1]: Leaving directory `/var/yp/eng'
```

nisbeast.kurtwerks.com has been set up as a NIS master server.

Now you can run `ypinit -s nisbeast.kurtwerks.com` on all slave servers.

After running the `ypinit` command as shown, a new directory containing the NIS map files exists, `/var/yp/nisbeast`. Storing NIS maps in domain-specific directories makes it easy for a single system to act as an NIS server for multiple NIS domains.

7. Start the password daemon, `yppasswdd`:

```
# service yppasswdd start
Starting YP passwd services: [ OK ]
```

8. Confirm that `yppasswd` is running:

```
# rpcinfo -u nisbeast yppasswd
program 100009 version 1 ready and waiting
```

9. Edit `/etc/sysconfig/network` and add the following line, commenting out or deleting any other line that begins with `NISDOMAIN`:

```
NISDOMAIN=eng
```

10. Use the `chkconfig` utility or the Service Configuration tool, as shown earlier, to configure `ypserv` and `yppasswdd` to start at boot time.

If you run slave servers, repeat Steps 7 and 8 for the transfer daemon, `ypxfrd` (that is, start `ypxfrd` and make sure that it is running). Also make sure to set `ypxfrd` to start at boot time in Step 10. Your shiny new NIS master server is now up and running and ready to answer requests from NIS clients. What's that? No clients? Read on.

Configuring an NIS Client

After you have successfully configured at least one master NIS server, you are ready to configure one or more NIS clients. The general procedure for setting up an NIS client involves the following steps:

1. Set the NIS domain name.
2. Configure and start the NIS client daemon.
3. Test the client daemon.
4. Configure the client's startup files to use NIS.

The following subsections describe these steps in detail and discuss the command and configuration file syntax. Note that there is some overlap between configuring a client and a server, so the discussion emphasizes client configuration tasks. The final subsection configures an example NIS client to illustrate the process of setting up a no-frills NIS client system that connects to the server configured at the end of the previous section.

Setting the NIS Domain Name

The initial step in configuring an NIS client is to set the NIS domain name. As explained in the previous section, execute the following command to set it:

```
# nisdomainname nisdomain
```

As before, replace `nisdomain` with the name of your NIS domain.

Configuring and Starting the Client Daemon

The NIS client daemon, `ypbind` uses a configuration file named `/etc/yp.conf` that specifies which NIS server's clients should use and how to locate them, a process known as binding the client to the server. NIS clients can use one of three methods to bind the server, and the type of entry in `/etc/yp.conf` controls the way binding takes place. The simplest entry takes the form:

```
ypserver nisserverip
```

This entry tells clients to use the server whose IP address is `nisserverip`. An example of this kind of entry might be:

```
ypserver 192.168.0.1
```

A somewhat more flexible approach enables clients to broadcast a query for the server to contact for a given NIS domain. This method saves tedious editing of client configuration files if (or, perhaps, when) the IP address of the NIS server changes. This entry takes the form shown here, where `nisdomain` is the name of the NIS domain of which the local host is a member.

```
domain nisdomain broadcast
```

An example entry for broadcast clients might resemble the following:

```
domain eng broadcast
```

Finally, if client systems are members of multiple NIS domains or if they can connect to one of several servers for the same NIS domain, the following form enables you to associate a given server with a given NIS domain:

```
domain nisdomain server nisserverip
```

This type of entry in `/etc/yp.conf` associates the NIS domain `nisdomain` with the NIS server (either master or slave) whose IP address is `nisserverip`. One example of this type of entry might be:

```
domain eng server 192.168.0.4
domain eng server 192.168.0.2
domain finance server 192.168.0.2
```

The first two lines identify two servers as the NIS servers for the `eng` NIS domain. The second and third lines indicate that the NIS server whose IP address is `192.168.0.2` serves two NIS domains, `eng`, and `finance`.

TIP If the client system can resolve hostnames to IP addresses without NIS (if, for example, the client runs a caching name server or has an entry in `/etc/hosts` for the NIS server), you can use a hostname instead of an IP address, but your best bet is to use IP addresses in `/etc/yp.conf` to minimize problems that might arise if name lookup services become inoperable for some reason.

To set up the client's NIS daemons, you can edit `/etc/yp.conf` directly or use the Authentication Configuration tool, a graphical tool for configuring user authentication. The following procedure shows you how to use the Authentication Configuration tool to configure a client system to use NIS:

1. Select Red Hat ⇨ System Settings ⇨ Authentication or type **system-config-authentication** (as root) in a terminal window to open the Authentication Configuration tool shown in Figure 13-7.
2. Check the Cache User Information check box. Setting this option causes the client to cache information retrieved from the server, making subsequent NIS lookups considerably faster.
3. Click the User Information tab.
4. Click the Enable NIS Support check box.
5. Click the Configure NIS button to open the NIS Settings dialog box (see Figure 13-8).
6. If the NIS Domain Name text box is not already filled in, type the NIS domain name.
7. Type the NIS server's IP address or name in the NIS Server text box. The NIS Settings dialog box should now resemble Figure 13-8. The NIS domain name is `eng` and the NIS server is `nisbeast.example.com`.
8. Check the Cache User Information check box to store authentication information at runtime. This will make lookups for NIS-based information faster.
9. Click OK to close the NIS Settings dialog box.
10. Click OK to close the Authentication Configuration tool.

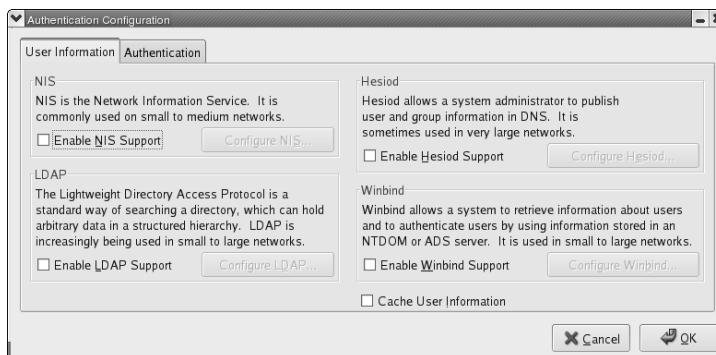


Figure 13-7 The Authentication Configuration tool.



Figure 13-8 The completed NIS Settings dialog box.

The following listing shows the edits made to `/etc/yp.conf` by the Authentication Configuration tool.

```
# /etc/yp.conf - ypbind configuration file
# Valid entries are
#
# domain NISDOMAIN server HOSTNAME
#     Use server HOSTNAME for the domain NISDOMAIN.
#
# domain NISDOMAIN broadcast
#     Use broadcast on the local net for domain NISDOMAIN
#
# domain NISDOMAIN slp
#     Query local SLP server for ypserver supporting NISDOMAIN
#
# ypserver HOSTNAME
#     Use server HOSTNAME for the local domain. The
#     IP-address of server must be listed in /etc/hosts.
#
# broadcast
#     If no server the default domain is specified or
#     none of them is reachable, try a broadcast call to
#     find a server
domain eng server nisbeast.example.com
```

NOTE If you use the server's IP address instead of its name, the IP address will appear in place of the server name.

NIS client programs, like the NIS servers, require RPC to function properly, so make sure the portmapper is running before starting the client daemon, `ypbind`. To start the client daemon, execute the following command, which invokes the `ypbind` initialization script:

```
# service ypbind start
Binding to the NIS domain:                [ OK ]
Listening for an NIS domain server.
```

After starting the NIS client daemon, use the command `rpcinfo -u localhost ypbind` to confirm that `ypbind` was able to register its service with the portmapper. The output should resemble the following:

```
# rpcinfo -u luther ypbind
program 100007 version 1 ready and waiting
program 100007 version 2 ready and waiting
```

NOTE If you skip the test procedure outlined in this section, you must at least set the domain name and create the `/var/yp` directory. Without this directory, `ypbind` does not start.

Finally, use one of the NIS client commands discussed in the section titled “Key NIS Client Files and Commands” to test whether the client and server are communicating properly. For example, use the `ypcat` command to display the contents of the NIS shared password file:

```
# ypcat passwd.byname
```

For user lookups to work properly on the client, do not add users whose authentication information will be retrieved using NIS on the client system. Instead, add a `+` sign to the end of `/etc/passwd` and `/etc/group` on your NIS clients. Experienced system administrators might use properly formatted entries for the password and group files (`+:*:0:0::` and `+:*:*`, respectively), but this isn’t necessary for NIS to work properly.

Now edit `/etc/host.conf` so that it uses NIS for hostname lookups. By default, the Fedora Core and Red Hat Enterprise Linux `host.conf` file looks like the following:

```
order hosts,bind
```

This configuration means that name service lookups first look in `/etc/hosts`, then use `bind`, the name server, to perform name lookups. Change this line so that it reads:

```
order hosts,nis,bind
```

This entry causes name lookups to query NIS after looking in `/etc/hosts` and before using the resolver library.

Last, edit `/etc/nsswitch.conf`. By default, Red Hat Linux is configured to perform standard NIS (as opposed to NIS+) lookups when user authentication and related information is requested. Among other entries, you should see lines that look like the following:

```
passwd:    files nis
shadow:    files nis
group:     files nis
hosts:     files nis
```

If you don't see these entries, add them.

Configuring the Client Startup Files

As when configuring an NIS server, you must modify some system configuration files and make sure that the client daemon starts and stops when the system starts and stops. In addition to setting the NIS domain name in `/etc/sysconfig/network` and setting the server information in `/etc/yp.conf`, you must enable `ypbind`, the NIS client, at boot time. You can use the `chkconfig` utility or the Service Configuration tool to start `ypbind` when the system boots.

Using `chkconfig`, issue the following commands:

```
# chkconfig --levels 0123456 ypbind off
# chkconfig --levels 345 ypbind on
```

To use the Service Configuration tool, start `system-config-services` as demonstrated earlier, scroll down to the bottom of the services list, and place a check mark beside the `ypbind` service. When you're done, select File ⇨ Save to save your changes, and then select File ⇨ Exit to close the Service Configuration tool.

NIS Client Commands

Table 13-2 lists the key NIS client commands and briefly describes their purpose.

The `ypcat` command enables you to view the contents of an NIS map. `ypcat` displays maps from the default server unless you request a specific NIS server using `-d nisdomain`. Similarly, to view the maps from a specific machine, use `-h hostname`, replacing `hostname` with the host in which you are interested.

Table 13-2 NIS Client Configuration Files and Commands

COMMAND	DESCRIPTION
<code>ypcat</code>	Prints the entries in an NIS database
<code>ypmatch</code>	Prints the value of one or more entries in an NIS map
<code>yppasswd</code>	Changes user passwords and information on the NIS server
<code>yppoll</code>	Displays the server and version number of an NIS map
<code>ypwhich</code>	Displays the name of the master NIS server

`ypwhich` invoked with no arguments displays the name of the default NIS server. If invoked with the `-d nisdomain` option, it queries the master NIS server for the NIS domain named `nisdomain`. You can also specify `ypwhich hostname` to query the NIS server, if any, on the machine named `hostname`. The `-x` option causes `ypwhich` to display the list of available maps.

Suppose, for example, that you want to know the list of hosts that the NIS server `nisbeast` knows about. First, use `ypwhich -x` command to see a list of map nicknames available on `nisbeast`:

```
$ ypwhich -x
Use "ethers"      for map "ethers.byname"
Use "aliases"     for map "mail.aliases"
Use "services"    for map "services.byname"
Use "protocols"   for map "protocols.bynumber"
Use "hosts"       for map "hosts.byname"
Use "networks"    for map "networks.byaddr"
Use "group"       for map "group.byname"
Use "passwd"      for map "passwd.byname"
```

This output means, for example, that the map `hosts.byname` can be accessed using the nickname or `hosts`. So, try `ypcat hosts`:

```
$ ypcat hosts
192.168.0.1      coondog.example.com coondog
192.168.0.2      hounddog.example.com hounddog
192.168.0.3      moonshine.example.com moonshine
127.0.0.1        localhost.localdomain localhost
127.0.0.1        localhost.localdomain localhost
192.168.0.4      nisbeast.example.com nisbeast
```

If you are looking for a specific piece of information, use the `ypmatch` command. For example, to find the user `bubba`'s password file entry, use the command:

```
$ ypcat passwd | grep bubba
bubba:$1$KXv8uWVw$Uk96z3r0bdHrM9gCfR.Ge0:501:501::/home/bubba:/bin/csh
```

A more elegant method is to tell `ypmatch` to do it:

```
$ ypmatch -k bubba passwd
bubba:$1$KXv8uWVw$Uk96z3r0bdHrM9gCfR.Ge0:501:501::/home/bubba:/bin/csh
```

As you can see, the output is the same, but `ypmatch` enables you to zero in on precisely the information you want without having to retrieve the entire map and filter the output. `ypmatch`'s `-k` option defines the key, or the information you want; the second argument tells `ypmatch` the map you want to

search (`passwd` in this case). To see bubba's group file entry, for example, you would specify the map group:

```
$ ypmatch -k bubba group
bubba bubba:!:501:
```

The `yppasswd` command enables users to change their NIS passwords. What's wrong with using plain vanilla `passwd`? The `passwd` command affects only the client machine. The `yppasswd` command, on the other hand, updates the NIS maps on the NIS server, which means that the updated password will be effective across the network, not just on the client machine. In fact, if you use the `passwd` command for a user that is authenticated via NIS, the password change, if it succeeds, will not be propagated to other NIS clients and will be discarded from the local machine's authentication databases the next time the NIS maps are updated.

Before you test the configuration, you need to have an NIS client configured — it's hard to test a server without a client — so we'll delay testing the server configuration until the end of the next section.

Configuring an Example NIS Client

This subsection illustrates configuring an NIS client to use the NIS services provided by the NIS server configured earlier in this chapter. As before, the NIS domain name is `eng`, running on the server `nisbeast.kurtwerks.com`, which has an IP address `192.168.0.1`.

1. Set the NIS domain name:

```
# nisdomainname eng
# nisdomainname
eng
```

2. Edit `/etc/yp.conf` to identify the default NIS server. The complete configuration file is (without comments):

```
ypserver 192.168.0.1
```

3. Make sure that the portmapper is running on the client:

```
# rpcinfo -u localhost portmapper
program 100000 version 2 ready and waiting
```

4. Start the primary client daemon, `ypbind`:

```
# service ypbind start
Binding to the NIS domain: [ OK ]
Listening for an NIS domain server.
```

WHAT ABOUT NIS+?

The Network Information Service Plus, NIS+, is a replacement for NIS that provides improved security, a more flexible naming model, and better support for large (okay, *enormous*) NIS installations. The security improvements include data encryption and secure RPC. The original NIS specification (often called *traditional NIS* to distinguish it from NIS+) suffered from the typical RPC vulnerabilities because it transmitted information over the wire as clear text, making it an easy target for packet snoopers and ne'er-do-wells. Data encryption makes snoopers' jobs more difficult. The naming scheme in NIS+ is dramatically different. The new method is very LDAP-like, organized around a tree of object nodes rather than a set of flat text files. Unfortunately, development of NIS+ for Linux has stopped. As a result, NIS+ for Linux is too immature to be considered for a production environment. If you need the additional security features and more flexible (and complicated) namespace offered by NIS+, you will want to use some combination of Kerberos and LDAP. See Chapter 34 for information about using Kerberos and LDAP for secure information sharing.

5. Confirm that `yplibd` is running:

```
# rpcinfo -u localhost ypbind
program 100007 version 1 ready and waiting
program 100007 version 2 ready and waiting
```

6. Edit `/etc/host.conf` and add NIS to the services used for hostname lookups. The completed file looks like this:

```
order hosts,nis,bind
```

7. Use the `chkconfig` utility or Service Configuration tool, as explained earlier in this chapter, to configure `ypbind` to start at boot time.

Using NIS and NFS Together

As we promised in Chapter 12, this section shows you how to use NIS and NFS to fully automate mounting NFS exports. What you'll discover, though, is that we didn't promise much. That is, the only thing you need to do to use NIS and NFS together is use NIS to distribute the automounter configuration files. The rest "just works." The technique uses NFS automounts to handle the mounting part of the process and NIS to distribute the automount files. The example shown in this section enables users to log in at any system (running an NIS client and an NFS client) and always have access to their home directory without having to make any modification on the client system except to enable NFS and NIS.

Here's the scenario:

- The NFS and NIS server is `luther.kurtwerks.com`, which has the IP address `192.168.0.4`.
- The NIS domain name is `possumholler`.
- The home directories to export reside in the file system `/export/homes`.
- The client system is `marta.kurtwerks.com`, which has the IP address `192.168.0.1`, and is running NFSv2.
- To keep the example clear, the exports will be mounted on the client system at the mount point `/net`.

And here's the procedure: The server system has already been configured as an NIS and NFS server. Likewise, the client system has already been configured as an NIS and NFS client. Accordingly, the following steps focus only on modifying these services on the server and client systems.

1. On the NFS server, add the exported file system to `/etc/exports`:

```
# cat /etc/exports
/export/homes
192.168.0.0/24(rw,no_subtree_check,secure,async,wdelay)
```

This entry exports the file system `/export/homes` to any client with an IP address in the range `192.168.0.1-192.168.0.254`. The export is configured as read-write, with subtree checking disabled, allows the server to handle NFS requests asynchronously, and permits the server to delay disk writes.

2. On the NIS server, create an automount file for `/net`. Keep in mind that this file, `auto.net`, will be used on *client* systems, not the server.

```
# /etc/auto.net
home -rw,soft,rsize=32678,wsiz=32678,nfsvers=2 luther:/export/homes
```

This entry tells the automounter to mount the file system `/export/homes` exported from `luther` under the mount point `/net/home` in read-write mode. The mount will be a soft mount, the read and write buffers can be a maximum of 32,678 bytes, and the protocol version to use is NFSv2. The latter measure is necessary because the client in the example is not running NFSv4.

3. Edit a *copy* of `/var/yp/Makefile` and make the following changes:
 - a. Beneath the line that reads (near line 93):

```
AUTO_LOCAL = $(YPSRCDIR)/auto.local
```

add the following entry:

```
AUTO_NET = $(YPSRCDIR)/auto.net
```


This entry tells the makefile where to find the `auto.net` auto-mounter file.

- b. At the end of the that reads (near line 109):

```
all:    passwd group hosts rpc service netid protocols mail \
```

insert the text `auto.net auto.master` before the terminating backslash. The modified rule should look like the following:

```
all:    passwd group hosts rpc service netid protocols mail auto.net
auto.master \
```

This change tells the makefile to include the `auto.net` and `auto.master` files as two of the files to convert to NIS maps.

- c. Add the text below to the end of the file:

```
auto.net: $(AUTO_NET) $(YPDIR)/Makefile
    @echo "Updating $@"
    -@sed -e "/^#/d" -e s/#.*$$// $(AUTO_NET) | $(DBLOAD) \
        -i $(AUTO_NET) -o $(YPMAPDIR)/$@ - $@
    -@$ (NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@
```

Each line after the first *must* be indented with a tab character! This entry (a *rule* in makefile parlance) tells make how to create an NIS map from the source `auto.net`. You don't need to understand how it works, just that it does.

4. Execute the make command in `/var/yp` to update the NIS maps:

```
# cd /var/yp
# make
gmake[1]: Entering directory `/var/yp/possumholler'
Updating netid.byname...
Updating auto.net...
Updating auto.master...
gmake[1]: Leaving directory `/var/yp/possumholler'
```

As you can see from the output, the `auto.net` file has been created. You can verify this by executing the following command:

```
# file /var/yp/possumholler/auto.net
/var/yp/possumholler/auto.net: GNU dbm 1.x or ndbm database, little
endian
```

5. On the server, start the `amd` service, which starts the server's auto-mount daemon:

```
# service amd start
Starting amd:
```

[OK]

6. On the client system, make sure that NIS and NFS client servers are running. If not, start them.
7. On the client, start the `autofs` service, which handles local automounts:

```
# service autofs start
Starting automount: [ OK ]
```

8. On the client system, make sure that the NIS client can communicate with the server:

```
# ypwhich
luther.kurtwerks.com
```

9. Ensure that the client system can “see” the `auto.net` map:

```
# ypcat auto.net
-rw,soft,rsize=32768,wsiz=32768,nfsvers=2 luther:/export/homes
```

10. Log in as user on the client system whose home directory is served by NFS and whose authentication information is provided by NIS. If everything has gone according to plan, you should be able to log in and wind up in the proper home directory.

Summary

In this chapter, you saw how to configure Fedora Core and Red Hat Enterprise Linux systems as NIS servers and clients. You first learned how to set up and test an NIS server and how to ensure that the NIS server comes up after a system reboot. You also learned how to configure an NIS client to connect to an NIS server for user-authentication information. Finally, you learned how to use NIS and NFS together, enabling NIS-authenticated users to mount their NFS-exported home directories on any NIS client system without requiring special setup at the NFS client.

Connecting to Microsoft and Novell Networks

IN THIS CHAPTER

- Installing Samba
- Configuring the Samba Server
- Creating Samba Users
- Starting the Samba Server
- Connecting to a Samba Client
- Connecting from a Windows PC to a Samba Server
- Connecting to Novell Networks

Using a program called Samba, you can emulate the Windows file-sharing protocol and connect your Fedora Core and Red Hat Enterprise network to a Windows network to share files and printers. Novell networks before version 5.0 use a native protocol known as IPX, and with Fedora Core and Red Hat Enterprise Linux you can emulate this protocol to enable file sharing and printing between Red Hat systems and Novell Netware systems. Novell versions 5.0 and later use native TCP/IP, and you can connect using that protocol. There are still many Novell installations running versions before version 5. In this chapter, you learn how to connect a Red Hat network to a Microsoft network and a Novell network running versions prior to version 5.0.

NOTE Both SMB and NCP support require kernel support, and this is active by default in both Fedora Core and Enterprise Linux kernels. If you build your own kernel, be sure to enable SMB and NCP support in the kernel.

Installing Samba

Computers running Windows 95 or greater use a protocol called Server Message Block (SMB) to communicate with each other and to share services such as file and print sharing. With Samba, the Linux PC icon appears in the Windows Network Places window, and the files on the Linux PC can be browsed using Windows Explorer. The Windows file system can be mounted on your Linux system, and you can browse the Windows files from your Linux PC.

Before you can use Samba to connect to the Windows computers, it must first be installed on the Linux PC. All current distributions of Fedora Core and Red Hat Enterprise Linux include three Samba packages: Samba, Samba-client, and Samba-common. They may not have been installed during the system installation. Even if they have been installed, you should always check for the latest versions to find out if any problems have been fixed by the latest release and to install it if necessary. To see whether Samba is installed on your system, type the following at a terminal window:

```
rpm -q samba
```

If Samba is not installed, the command returns the output stating that Samba is not installed. If Samba is installed, the RPM query returns the version number of the Samba program installed on your system.

The latest version of Samba can be obtained at Samba's Web site: samba.org. Follow the instructions at the site for downloading the RPM file for your distribution. After downloading the Samba RPM file, install it as follows ("name of file" is the version number downloaded):

```
rpm -i samba(name of file)
```

Be sure to install the Samba-common RPM, and if you want to use the Samba-client, also install the Samba-client RPM. If you are unable to download the RPM version, or if you want to compile the program yourself, download the file `samba-latest.tar.gz`. Extract the file using the following command:

```
tar -xfvz samba-latest.tar.gz
```

Change to the directory containing the extracted files (usually `/usr/src`) and type **./configure**.

Press Enter and wait for the command prompt to return. From the command prompt, type **make**.

Press Enter and wait for the command prompt to return. Finally, type **make install** from the command prompt. If all goes well, Samba is installed when the command prompt returns. Now you need to configure it.

For Samba to provide its services, the Red Hat Linux PC needs to be configured.

NOTE In this chapter, I refer to the Red Hat Linux PC as the Samba server and the Windows PC as the Samba client.

Configuring the Samba Server

Before you can use Samba to connect with your Windows PCs, it must be configured. While there are several graphical-based programs available for configuring Samba, these programs are just front ends that make changes to the Samba configuration file behind the scenes. It is much quicker and easier to edit the Samba configuration file itself. The Samba configuration file is called `smb.conf` and is located in the `/etc/samba` directory by the installation program. A sample `smb.conf` file was created during the installation that can be used for reference and modification.

The `smb.conf` file is divided into several sections, called shares, the names of which I show as bracketed subsection titles in the following discussion. Shown next is the `smb.conf` file from one of the computers I use at school. Refer to this file to see what a section looks like as it is described.

```
# This is the main Samba configuration file. You should read the
# smb.conf(5) manual page in order to understand the options listed
# here. Samba has a huge number of configurable options (perhaps too
# many!) most of which are not shown in this example
# Any line which starts with a ; (semi-colon) or a # (hash)
# is a comment and is ignored. In this example we will use a #
# for commentry and a ; for parts of the config file that you
# may wish to enable
# NOTE: Whenever you modify this file you should run the command
"testparm"
# to check that you have not made any basic syntactic errors.
===== Global Settings
=====
[global]
    log file = /var/log/samba/%m.log
    smb passwd file = /etc/samba/smbpasswd
    load printers = yes
    passwd chat = *New*password* %n\n *Retype*new*password* %n\n
    *passwd:*all*authentication*tokens*updated*successfully*
    socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192
    obey pam restrictions = yes
    encrypt passwords = yes
    passwd program = /usr/bin/passwd %u
```

```
dns proxy = no
netbios name = rhl
writeable = yes
server string = Samba Server
printing = lprng
path = /home
default = homes
unix password sync = Yes
workgroup = Tardis
printcap name = /etc/printcap
security = user
max log size = 50
pam password change = yes

[homes]
comment = Home Directories
browseable = yes
writeable = yes
create mode = 0664
directory mode = 0775
max connections = 1

[printers]
browseable = yes
printable = yes
path = /var/spool/samba
comment = All Printers
```

[global]

The first section of the `smb.conf` file is the `[global]` section. The `[global]` section contains settings that apply to the entire server and default settings that may apply to the other shares. The `[global]` section contains a list of options and values in the following format:

```
option = value
```

You have hundreds of options and values at your disposal, and you look at the most common ones here. For a complete listing of options, refer to the `smb.conf` man page. Some of the more significant options are:

- **workgroup = Tardis** — This is the name of the workgroup shown in the identification tab of the network properties box on the Windows computer.
- **smb passwd file = /etc/samba/smbpasswd** — This shows the path to the location of the Samba password file. Be sure that you include this option/value pair in your `smb.conf` file.

- **encryptpasswords = yes** — Beginning with Windows NT service pack 3 and later, passwords are encrypted. If you are connecting to any systems running these versions of Windows, you should choose encrypted passwords.
- **netbios name = RHL** — This is the name by which the Samba server is known to the Windows computer.
- **server string = Samba Server** — This is shown as a comment on the Windows PC in the network browser.
- **security = user** — This is the level of security applied to server access. Other options are *share*, *domain*, and *server*. *Share* is used to make it easier to create anonymous shares that do not require authentication, and it is useful when the NetBIOS names of the Windows computers are different from other names on the Linux computer. *Server* is used to specify the server to use if the password file is on another server in the network. *Domain* is used if the clients are added to a Windows NT domain using *smbpasswd*, and login requests are executed by a Windows NT primary or backup domain controller.
- **log file = /var/log/samba/log** — This is the location of the log file.
- **max log size = 50** — This is the maximum size in kilobytes that the file can grow to.
- **socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192** — This enables the server to be tuned for better performance. *TCP_NODELAY* is a default value; the *BUF* values set send and receive buffers.
- **dns proxy = No** — This indicates that the NetBIOS name will not be treated like a DNS name and that there is no DNS lookup.

[homes]

The next section of the `smb.conf` file, `[homes]`, is used to enable the server to give users quick access to their home directories. Refer to the `smb.conf` man page for a more complete description of how the `[homes]` section works.

- **comment = Home Directories** — A comment line.
- **browseable = yes** — Means that the directory will appear in the Windows file browser.
- **writable = yes** — Means that users can write to their directories.
- **create mode = 0664** — Sets the default file permissions for files created in the directory.

- **directory mode = 0775** — Sets the default permissions for created directories.
- **max connections = 1** — The maximum number of simultaneous connections allowed. Setting this number to 1 prevents a user from logging in to the server from more than one location. Setting this number to 2 allows a user to log in from two locations and so on. Setting this number to 0 allows an unlimited number of connections.

[printers]

This section sets the options for printing.

- **path = /var/spool/samba** — The location of the printer spool directory.
- **printable = yes** — Enables clients to send print jobs to the specified directory. This option must be set, or printing does not work.
- **browseable = yes** — Means that the printer appears in the browse list.

NOTE Be sure to have your printer properly configured for your Linux network before you attempt to set it up for use with Windows clients. You may need to enter the location of the path to the print spool for the printer you want to use in the `smb.conf` file.

The `smb.conf` file shown in the examples allows users who already have system accounts to access their home directories and to use printers. After modifying and saving the `/etc/samba/smb.conf` file, check the syntax of the file. To do this, you can use the `testparm` command as follows:

```
[root@terry terry]# testparm
Load smb config files from /etc/samba/smb.conf
Processing section "[printers]"
Processing section "[homes]"
Loaded services file OK.
Press enter to see a dump of your service definitions
```

Pressing Enter displays the contents of the configuration file. The `smb.conf` file is now ready to use.

Creating Samba Users

If you are using security-server and have assigned a domain instead of a work-group, you don't need to create Samba users since your users will be logging

into a Windows domain controller. In this example though, we will create a Samba users' password file. You can convert all of your system users to Samba users by running the following command:

```
cat /etc/passwd | mksmbpasswd.sh > /etc/samba/smbpasswd
```

This utility creates only the users' accounts, not their passwords. You need to create passwords for your users by using the `smbpasswd` command and the user's name as shown here:

```
[root@terry terry]# smbpasswd terry
New SMB password:
Retype new SMB password:
Password changed for user terry
Password changed for user terry
```

Starting the Samba Server

The last step is to start the Samba daemon. The command to start Samba is:

```
[root@terry terry]# /sbin/service smb start
Starting SMB services:          [ OK ]
Starting NMB services:         [ OK ]
```

At this point, you should have a functioning Samba server running on your system. It is configured to allow users who have accounts on your Red Hat Enterprise Linux system to access their home directories from a Windows PC. Logged-in users are also able to use the printers configured with the Red Hat system.

Connecting to a Samba Client

In this section, you learn how to connect your system to other systems running the SMB protocol. Connecting a PC running Windows 2000 or XP is covered in the section "Connecting from a Windows PC to the Samba Server," later in this chapter. You can connect your system to any computer that is running the SMB protocol, whether it is a Windows PC or another Linux system running Samba. The connection can be made from the command line using two methods. The first uses a utility called `smbclient`, and the command syntax is `smbclient //computer name/sharename`, as shown in the following example. Be sure to replace the computer name in the example with the name of your computer.

```
[root@terry terry]# smbclient //terrycollings/c
added interface ip=192.168.9.93 bcast=192.168.9.255 nmask=255.255.255.0
Got a positive name query response from 192.168.9.102 (192.168.9.102)
Password:
Domain=[Tardis] OS=[Windows 5.0] Server=[Windows 2000 LAN Manager]
smb: \>
```

The preceding example shows me logging in to my Windows PC from my Red Hat system. I was prompted for a password to log in and was given some information about the Windows system and a command prompt. You can type `help` at the command prompt to get a list of possible commands. The commands at the `smb` prompt are very similar to command-line FTP commands. To exit the connection, type **exit**.

Another way to make the files on the Samba client accessible on your Red Hat system is to mount the client file system on your file system. You can do this using the `smbmount` command. The syntax for this command is `smbmount //computer name/directory /mysystem/mount/point`, as shown in the following example:

```
[root@terry terry]# smbmount //terrycollings/c /mnt/windows
Password:
```

Next, you can change to the directory on the system where you mounted the Windows system by issuing the following command:

```
[root@terry terry]# cd /mnt/windows
```

Then you can get a directory listing by using the `ls` command:

```
[root@terry windows]# ls
arcldr.exe           MSDOS.SYS           quicktime
arcsetup.exe         Muhlnt Setup       QuickTimeInstaller.zip
AUTOEXEC.BAT        My Download Files  Recycled
boot.ini             My Music           rhsa
camtasia             NALCache           W2K.CD
CONFIG.SYS           netplus            Windows Update Setup Files
Documents and Settings Novell              WINNT
Drivers              NTDETECT.COM       WSREMOTE.ID
fgc                  ntldr              WT61CE.UWL
hiberfil.sys         p2.pdf             WT61OZ.UWL
IO.SYS               pagefile.sys       WT61UK.UWL
lconfig.aot          Program Files      WT61US.UWL
Local Muhlnt         PUTTY.RND
```

In this example, I am connecting to the same Windows PC to which I connected in the previous example. However, by using the `smbmount` command, I am mounting the Windows file system on my Red Hat file system. After

entering the password for the Windows PC and returning to the command prompt, I change to the directory that I just mounted and run the `ls` command to obtain a directory listing of the Windows PC share I mounted. I can now easily move files between the two systems using regular file system utilities.

You can put the `mount` command into a local startup script so that the directories are mounted at system boot, if you desire. Use the command as shown earlier and add an option to look for a file that contains the login username and password.

```
smbmount //terrycollings/c /mnt/windows -o credentials=/home/terry/.sambacred
```

You need to create a file as shown in the following code. I created a hidden file called `.sambacred` and in the file I placed these two lines:

```
Username = terry  
password = (password)
```

Be sure to put the information relevant to your system for the path to the file and the username and password. The reason for creating the hidden file and pointing to it from the `smbmount` command is to prevent someone from being able to find out the password that is in the file in plain text. Be sure to change the permissions on this file accordingly.

Using this method is preferable to placing the mount information in the `/etc/fstab` file. While it is possible to place the mount information in the `/etc/fstab` file, someone could find out the username and password just by looking at the `/etc/fstab` file.

To unmount the client file system, enter the following `smbumount` command and the path to the directory to unmount:

```
# smbumount /mnt/windows
```

After you press Enter, the file system will be unmounted.

Connecting from a Windows PC to the Samba Server

Now you are ready to test your connection on the Windows PC. For systems running Windows 2000 or XP, no configuration is required. On the Windows computer, double-click the My Network Places icon from the desktop. In the Network Places window, you should now see a listing for the Red Hat computer, which in this example would be `rhl10`. Double-click the `rhl10` PC icon, and you will see the shares you made available.

WHY USE SAMBA INSTEAD OF NFS?

Earlier, you set up the Network File System to enable file sharing across your network. Why didn't you just use NFS to share the Windows files? Well, you could have, but it makes more sense to use Samba to communicate with Windows computers. Windows systems don't natively support NFS; the Server Message Block (SMB) protocol is how Windows computers communicate with each other. It is the Windows native protocol for sharing files and printers. By using the same protocol, you are ensuring that file and printer sharing operate with a minimum of difficulty.

When you double-click the directory share from the rh110 PC, you are prompted for a username and password to enter the directories. That's all there is to it. Now you can share files between your Linux and Windows computers.

TIP You can use the Windows Map Network Drive command to set up a Windows system to always connect to your Samba share when you start the Windows PC.

CROSS-REFERENCE To learn how to set up a printer to use under Samba, refer to Chapter 10.

Connecting to Novell Networks

With Red Hat Enterprise Linux, you can easily configure your system to connect to Novell Netware servers. In this section, you learn how to configure your Red Hat system to be a Novell client.

Two packages need to be installed to enable communication between the Novell servers and your Fedora or Red Hat Enterprise system. The first package to load is `ipxutils`, which is Internetwork Packet Exchange protocol — the Novell protocol for networking. The second package you need to load is `ncpfs` — the Novell file system package. If you do not have these packages installed on your system, the RPMs can be found on the Red Hat Installation CDs. Install the packages before continuing.

After the packages have been installed, two modules need to be loaded before you can configure the Novell client. From a terminal command prompt, enter the following command to load the modules.

```
#/sbin/modprobe ipx ncpfs
```

NOTE The `ipx` and `ncpfs` kernel modules are now in the kernel-unsupported package, which may not be installed. You may have to install this package to obtain the necessary modules. Or, you can do a Google search for `ipx rpm` and `ncpfs rpm` for your distribution.

After the modules are loaded, you need to configure your IPX connection by entering the following command, which attempts to automatically detect and configure your IPX connection.

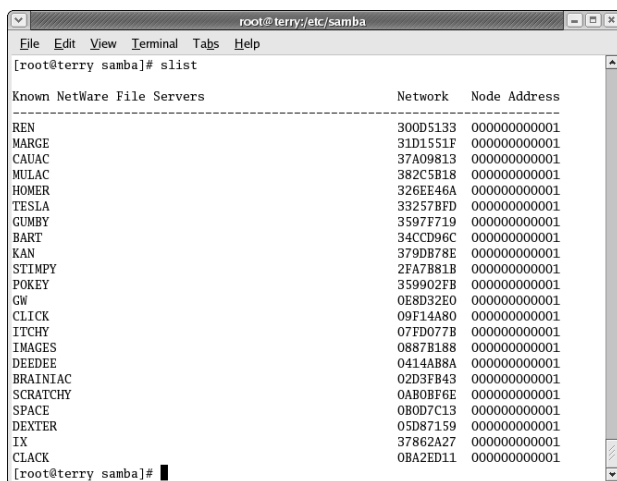
```
#/sbin/ipx_configure -auto_interface=on -auto_primary=on
```

The previous command enables the automatic configuration of the interface and detects the primary connection. You can check if your configuration is successful by issuing the `slist` command from the terminal prompt. The output from the `slist` command is shown in Figure 14-1.

If you receive a listing similar to that shown in Figure 14-1, your IPX configuration is correct, and you can attempt to log in to a server and mount its file system on your system. To do this you will use the `ncpmount` command as follows:

```
ncpmount -S (fileserver name) (fileserver directory (optional)) -U (fileserver  
login id (complete context)) -P (password) (directory mount point).
```

Be sure to replace the information in the parentheses with the information appropriate to your system.



Known NetWare File Servers	Network	Node Address
REN	300D5133	000000000001
MARGE	31D1551F	000000000001
CAUAC	37A09813	000000000001
MULAC	382C5B18	000000000001
HOMER	326EE46A	000000000001
TESLA	33257BFD	000000000001
GUMBY	3597F719	000000000001
BART	34CCD96C	000000000001
KAN	379DB78E	000000000001
STIMPY	2FA7B81B	000000000001
POKEY	359902FB	000000000001
GW	0E8D32E0	000000000001
CLICK	09F14A80	000000000001
ITCHY	07FD077B	000000000001
IMAGES	0887B188	000000000001
DEEDEE	0414AB8A	000000000001
BRAINIAC	02D3FB43	000000000001
SCRATCHY	0AB0BF6E	000000000001
SPACE	0B0D7C13	000000000001
DEXTER	05D87159	000000000001
IX	37862A27	000000000001
CLACK	0BA2ED11	000000000001

Figure 14-1 The output from the `slist` command shows the Novell servers on the network.

Suppose that you want to log in to the server BART, shown in Figure 14-1, and mount its file system on my system in the `bart` directory. You need to enter the following command:

```
# ncpmount -S BART -U terry.mediaservices.admin -P password  
/home/terry/novell/bart
```

After logging in and mounting the Novell file system on your system, a directory listing shows the contents of the server Bart. Be sure to replace the information in the example with the information appropriate for your systems.

You can unmount an NCP file system by issuing the `ncpumount` command and specifying the mount point to unmount. During system shutdown, the NCP mounts are cleanly unmounted, so it is not necessary to unmount the file system before shutting down the system.

If you do not want to go through the manual configuration process to connect your system as a Novell client each time you start your system, you should place the commands in this section in your `rc.local` file. The configuration then occurs automatically at boot time. You can even have the file system mounted at the same time, if you desire.

CROSS-REFERENCE See Chapter 10 for more information on how to configure a Novell networked printer.

Summary

In this chapter, you learned about the Server Message Block (SMB) protocol. This protocol is used by Windows computers to communicate with each other. You learned about a Linux program called Samba, which emulates the SMB protocol and can be used to connect Linux networks to Windows networks. You installed and configured Samba on a Red Hat Enterprise Linux server and configured the Windows clients. You also learned how to configure your Red Hat system as a Novell client using the NCP file system.

Configuring a Database Server

IN THIS CHAPTER

- Linux Database Servers
- Using MySQL
- Using PostgreSQL

As Linux continues to penetrate deeper into the enterprise, it is increasingly being used to house corporate data. This chapter describes installing, configuring, and testing three of the major relational database management systems (RDBMSs) on Fedora Core and Red Hat Enterprise Linux: MySQL, and PostgreSQL, the two most popular open source RDBMSs, and Oracle, the 800-pound gorilla of commercial RDBMSs. This chapter's scope is limited to helping you install these database systems and to verify the basic functionality of the installed system. In the interests of setting your expectations appropriately, you will not be a database expert after reading this chapter because we don't cover SQL, database design, query optimization, or any other technical database-specific topics. Those subjects are worth a book (each).

Linux Database Servers

Much as Linux has come to dominate the world of Web servers and cluster computing, Linux is gradually becoming the preferred OS to host corporate data and it is doing so for the same reasons that it dominates Web services and high-performance computing: Linux is fast, stable, reliable, tunable, and flexible; the source is available; and it runs on commodity hardware. Most of the major databases, including Oracle, DB/2, Informix, MySQL, and PostgreSQL,

run on it. As a result, system administrators have to contend with running database servers.

As a system administrator, you rarely have the final say or deciding vote in which database you will run on your server and be responsible for maintaining. Indeed, someone farther up the food chain than you usually dictates the decision. Nonetheless, it is your responsibility to make sure that the server is running, is accessible, and performs acceptably. In most shops, Linux system administrators are not responsible for the daily care and feeding of the database because database administrators (DBAs) handle that task. At smaller sites that don't have the luxury of a dedicated DBA, you might have more responsibilities for database maintenance, but, as a rule, you won't be writing queries, building applications, or providing technical support for the database.

Nonetheless, having passing familiarity with the services that your system or systems provide is essential. Knowing at least how to install a database server and how to make sure it is running is important because DBAs sometimes lack the ability to administer the system on which the database runs; their job is to groom the database, not the hardware and operating system on which it runs.

Regardless of the database system used, the basic procedure for bootstrapping a database server is the same:

1. Install and configure the hardware.
2. Install and configure the operating system.
3. Install and configure the database software to create a standard installation.
4. Make sure that the basic installation works.
5. Set up monitoring and backup jobs for the database.
6. Turn it over to the DBA(s).

The purpose of this chapter is to address Steps 3 and 4. The particulars involved in installing and configuring the database software varies, of course. Likewise, making sure the stock installation works requires different measures depending on the database you are using. This chapter isn't long enough to cover all of the popular database packages available, so it concentrates on the two most popular ones, MySQL and PostgreSQL. MySQL is the most popular of the three in the free software world and it certainly outstrips PostgreSQL in terms of the installed base. We chose PostgreSQL because it is part of Fedora Core and RHEL *and* because it is the database technology underneath the Red Hat Database. Together, familiarity with these two database packages should cover the majority of situations in which you have to install, configure, and maintain a database server.

Using MySQL

MySQL is the most popular open-source RDBMS in the world. It is popular enough, at least, that it is the third part of an acronym widely used to describe Web services built with free tools, LAMP, which stands for Linux, Apache, MySQL, and PHP (or Perl or Python, depending on who you ask). MySQL's popularity is result of its speed, ease of use, flexibility, and, to be sure, its cost. Most Linux distributions include MySQL as part of server-class installations. Another important element of MySQL's popularity is that it integrates smoothly and seamlessly with the other pillars of the LAMPpost, Linux, Apache, and PHP. For example, Apache's `mod_auth_mysql` module integrates MySQL authentication directly into Apache, making it possible to control access to Apache-based services using a MySQL database. Third-party Apache modules (modules not part of the standard Apache distribution) that provide MySQL support or features include the following (numbers in parentheses indicated the Apache version the module supports):

- **mod_accessCookie** — Implements Apache's `Allow From` directive using a cookie validated against a MySQL database (1.3.x)
- **mod_auth_cookie_mysql** — Provides authentication against a MySQL database using cryptographically secure cookie (1.3.x)
- **mod_auth_form** — Performs authentication using forms and a MySQL database (2.x)
- **mod_auth_sim** — Incorporates authentication and session ID management into Apache using a MySQL database (2.x)
- **mod_authn_db** — Uses a database-independent API (libdbi) to provide authentication services against a MySQL database (2.x)
- **mod_crescent** — Serves as the foundation for creating a "virtual community" using MySQL for authentication (1.3.x)
- **mod_dbd_mysql** — Provides a DBD driver module for MySQL based on `mod_dbd` (2.x)
- **mod_log_mysql** — Logs Apache requests to a MySQL database instead of or in addition to the standard flat file, `/var/log/httpd/access_log` (1.3.x, 2.x)
- **mod_my** — YAMAM (Yet Another MySQL Authentication Module) that stores authentication information in a MySQL database (1.3.x, 2.x)
- **mod_sqlinclude** — Uses a MySQL database to store the Apache configuration file, functionality intended to simplify managing large numbers of virtual (Web) hosts (1.3.x)

- **mod_vhost_mysql2** — Maintains virtual host configurations in a MySQL database (2.x)
- **mod_vhs** — Stores virtual host configuration data in a MySQL database (2.x)

NOTE Chapters 23 and 24 discuss Apache and using Apache modules in detail.

PHP, an extremely popular Web scripting language, has at least two different APIs for using MySQL in PHP-based applications. Python, a popular and easy-to-use object-oriented programming language, has a module that incorporates MySQL into Python-based programs in a standard, uniform fashion. Other programming languages also incorporate MySQL via modules, loadable libraries, or APIs. Even the `zsh` shell includes a set of predefined shell functions for using MySQL command line client programs (`mysql`, `mysqlshow`, `mysqldump`, `mysqldiff`, and `mysqladmin`).

If you're convinced that MySQL is the database you want to use, it will help to make sure it is installed. Use the `rpmquery` command to see if the packages `mysql-server`, `mysql`, and `mysql-devel` are installed. You can use the following commands to see if these packages are installed:

```
# rpmquery mysql-server
mysql-server-3.23.58-14
# rpmquery mysql
mysql-3.23.58-14
# rpmquery mysql-devel
mysql-devel-3.23.58-14
```

The versions installed on your system might be different by the time you read this. If these packages aren't installed (you'll need at least `mysql-server` and `mysql`), install them. `mysql-server` contains the MySQL server, sample configuration files, the system initialization scripts, a `logrotate` script for the server's log files, and a two directories the server uses at runtime. The `mysql` package contains the client programs, a shared library the client utilities need to interact with the server, and manual pages and language files for the client programs. The `mysql-devel` package installs the header files and libraries necessary to write MySQL programs in C, C++, and Objective-C.

NOTE Chapter 30 explains how to install RPM-based software packages.

Other MySQL-related packages that might be installed or that you might want to install include:

- **mod_auth_mysql** — Provides an Apache module that uses MySQL to control access to Web pages

- **libdbi-dbd-mysql** — Installs a device-independent database driver for use by programs using libdbi
- **php-mysql** — Contains a PHP module that enables PHP to connect to and manipulate MySQL databases
- **mysql-bench** — Includes a suite of benchmark tests and test data to use for benchmarking MySQL's performance on your system

Securing the MySQL Installation

Part of the MySQL installation process installs a script to create a database (named `mysql`) of administrative tables that handle access control and database privileges, a test database (imaginatively named `test`), an administrative user for the database (named `root`), and an anonymous user (named `anonymous`). This script is executed the first time you start the `mysqld` database daemon. Neither the `root` account nor the anonymous account is password-protected, so the first thing you want to do is create a password for the `root` account. In our opinion, naming the administrative user `root` was a poor choice because it is quite confusing in that the superuser on your system is also named `root`. This user has superuser privileges on the database, so `root` is a natural but unfortunate choice. Just so there's no confusion, *MySQL's root user is not the same as the system's root user*.

Before attempting to change any passwords, verify that the database is running. One way to do so is to become the (system) root user and use the service utility:

```
# service mysqld status
mysqld (pid 24900) is running...
```

If `mysqld`, the MySQL server daemon, isn't running, you can start it using the usual command:

```
# service mysql start
Starting MySQL: [ OK ]
```

Another way to do test the server, one that doesn't require root access, is to use the `mysqladmin` and/or `mysqlshow` commands to see whether the server is running and responding to connections. For example, the following `mysqladmin` command shows the server version:

```
$ mysqladmin version
mysqladmin Ver 8.23 Distrib 3.23.58, for redhat-linux-gnu on i386
Copyright (C) 2000 MySQL AB & MySQL Finland AB & TCX DataKonsult AB
This software comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to modify and redistribute it under the GPL license
```

```
Server version      3.23.58
Protocol version    10
Connection          Localhost via UNIX socket
UNIX socket         /var/lib/mysql/mysql.sock
Uptime:             2 min 55 sec
```

```
Threads: 1  Questions: 2  Slow queries: 0  Opens: 6  Flush tables: 1  Open tabl
es: 0  Queries per second avg: 0.011
```

The output might differ slightly, depending on the version of MySQL installed. The `mysqlshow` command can be used to get information about the server, such as what databases it is serving and the tables that exist in those databases. For example, a bare `mysqlshow` command displays the available databases:

```
$ mysqlshow
+-----+
| Databases |
+-----+
| mysql     |
| test      |
+-----+
```

You'll learn more about the MySQL client programs in the next section.

After you've established that MySQL is running, set the passwords for the MySQL root account using the `mysqladmin` commands shown in the following listing (you must be root to execute these commands):

```
# mysqladmin -u root password "sekritword"
# mysqladmin -u root -h hostname "sekritword"
```

The first command sets the password for MySQL's `root` user when it is connecting from `localhost`, to `sekritword`. The second command changes the password for the MySQL when root is connecting from the `hostname` specified by `hostname`. What's the difference? MySQL distinguishes connections by the username *and* by the host from which users are connecting. By default, MySQL assumes that a user is connecting from `localhost` (that is, the IP address `127.0.0.1`), so `username@localhost` needs a password. However, in most cases, the `localhost` also has a fully qualified domain name (that is, a `hostname`), such as `datagrunt.example.com`, and MySQL considers such a connection distinct. Accordingly, `username@hostname` (say, `username@datagrunt.example.com`) *also* needs a password.

Use the following command to see the results of your commands:

```
$ mysql -e "select host, user, password from mysql.user" -u root -p
Enter password:
```

host	user	password
localhost	root	5d2e19393cc5ef67
datagrunt.example.com	root	5d2e19393cc5ef67
localhost		
datagrunt.example.com		

You can see that the password for the root use has been set. You can also see that the password you entered has been encrypted. The `-u root` argument to the `mysql` command specifies the username; `-p` tells `mysql` to show a password prompt. You should enter the password you used when you set the password as shown earlier.

Notice that a similar set of accounts exists for a user with no name; this is the anonymous user mentioned previously. You need to decide at this point if you want to permit anonymous access to the server. If you do, you can leave the accounts without a password or you can set a password. Alternatively, you can delete the anonymous accounts entirely. Whether you leave the accounts intact is a matter of local security policy. We leave them in place and don't set passwords on them for testing applications during development and delete them on production systems. The anonymous user has limited privileges (essentially, read-only), but it isn't a good idea to have unsecured accounts on a production system.

If you choose to set a password for the anonymous accounts, use the commands shown in the following example:

```
# mysql -u root -p
Enter password:
mysql> set password for '@localhost = password('magicword');
Query OK, 0 rows affected (0.00 sec)

mysql> set password for '@hostname = password('magicword');
Query OK, 0 rows affected (0.00 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.00 sec)

mysql> quit
Bye
```

The semicolons (;) terminating each command are required. The first command starts the MySQL shell, a command interpreter for the MySQL database server, using MySQL's root account. The rest of the commands are executed from the MySQL shell, which uses the `mysql>` prompt. In your commands, replace `magicword` with the password you've chosen and `hostname` with

the fully qualified domain name of your system. The `flush privileges` instruction causes MySQL to reread the access tables and makes the new password for the anonymous account take effect. Notice that the anonymous username is specified using a pair of single quotes. This is necessary because the anonymous account doesn't, strictly speaking, have a username. The last command, `quit`, terminates the MySQL shell session and returns you to the command prompt.

TIP If you make a real mess of the instructions in this section or just want to start over, you can restore your MySQL installation to its original state by using the following procedure:

1. Stop MySQL:

```
# service mysqld stop
```

2. Delete the MySQL data directories and files in `/var/lib/mysql`:

```
# cd /var/lib/mysql
# rm -rf mysql test
```

3. Restart MySQL:

```
# service mysqld start
```

This procedure works because the `mysqld` initialization script creates the initial databases if the directory `/var/lib/mysql/mysql` doesn't exist.

If you prefer to delete the anonymous accounts entirely, use the following commands:

```
$ mysql -u root -p
Enter password:
mysql> delete from mysql.user where user = '';
Query OK, 2 rows affected (0.02 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.00 sec);

mysql> quit
Bye
```

With the root account properly secured and having made a decision about how to handle the anonymous accounts, you are ready to learn a bit more about the MySQL client programs.

Using the MySQL Client Programs

What precisely is a MySQL client program? MySQL is a standard client-server program. The MySQL server daemon, `mysqld`, is the actual database server. It listens for incoming connections and retrieves, manipulates, and returns data. It has no interface other than that provided by a client API. Programs that are written to MySQL's client API provide the user interface to the MySQL server. It is the client programs that enable you to submit queries to the database, to add and delete users, and so on. The client programs also make it easier to perform certain tasks. For example, in theory, a SQL database can be manipulated entirely using SQL statements. However, to simplify certain activities, it is often more convenient to use programs that hide SQL functionality behind a simpler interface. MySQL's client programs provide this simpler interface.

You've already seen three of the MySQL client programs in action, `mysqladmin`, `mysqlshow`, and `mysql`. `mysqladmin` is a utility that enables you to perform administrative activities, such as:

- Creating, modifying, and deleting (*dropping*, in SQL parlance) databases
- Starting and stopping the MySQL server
- Confirming the database is up
- Finding out which server threads are running
- Killing specific MySQL server threads
- Retrieving status information from a running server
- Flushing (syncing) data to disk
- Changing passwords

`mysqladmin`'s basic syntax is:

```
mysqladmin -u username -p[password] command
```

Replace `username` with the database username, such as `root`, that you want to use. The account specified in `username` must have the privileges required to perform the requested operation. If you specify just `-p`, MySQL will prompt you for `username`'s password. You can add the password after `-p`, but doing so isn't a good idea because it will appear on screen. `command` specifies the operation you want to perform. For example, to create a database named `techbooks`, the command would be:

```
mysqladmin -u username -p create techbooks
```

To delete (drop) this database, use the following command:

```
mysqladmin -u username -p drop techbooks
```


To change the password for the root user, you would use the following command:

```
mysqladmin -u username -p password 'new_password'
```

Replace `new_password` with the password you want to assign to `username` and make sure to enclose the new password in single quotes (`'`). In this case the command passed to `mysqladmin` is `password 'new_password'`; the `-p` option is *not* being given an argument of `password`.

To stop a running server, use the shutdown command, as shown in the following example:

```
mysqladmin -u username -p shutdown
```

For more details about using the `mysqladmin` command, see the `mysqladmin` main page or refer to the MySQL documentation.

`mysqlshow` is a utility that displays the structure of a MySQL database, the tables in that database, and the columns (or fields) that make up that database. It uses an option syntax similar `mysqladmin`, but takes different (and fewer) arguments:

```
mysqlshow -u username -p [database [table [column]]]
```

As before, replace `username` with the user account you want to use. If database is not specified, `mysqlshow` displays all of the available databases. If database is specified, `mysqlshow` lists the tables that exist in database. If table is also specified (it must exist in the indicated database), `mysqlshow` displays that table's columns (fields). If column is also specified (the column must exist in the specified table, which must likewise exist in the requested database), `mysqlshow` displays that column's characteristics. For example, the following command display the tables in the `mysql` database:

```
$ mysqlshow -u root -p mysql
Database: mysql
+-----+
| Tables |
+-----+
| columns_priv |
| db           |
| func         |
| host         |
| tables_priv  |
| user         |
+-----+
```

`mysql`, as already explained, is a MySQL shell or command interpreter. The commands it interprets are SQL statements. `mysql` gives you the most direct access to the MySQL's database engine, but also requires that you speak fluent SQL. You enter SQL statements at a command prompt, the interpreter passes them to the database engine, and the database engine sends the results of those SQL statements back the interpreter, which displays the results on the screen.

There are many other MySQL clients. Table 15-1 lists the ones you are most likely to use; there are others, but they are special-purpose programs that (we hope) you never need to use.

We don't have the space to go into all of MySQL's capabilities, much less provide proper guidance on using all its commands and utilities. The initial setup instructions and the short introduction to some of the MySQL client commands should, nevertheless, get you started. Fortunately, one of MySQL's strongest selling points is that it is ready to run with minimal setup after installation and that it requires very little ongoing maintenance. MySQL's simplicity makes it an ideal choice for busy system administrators who have enough to do keeping their mail servers from getting clogged up with spam and viruses without having to learn how to maintain a complicated RDBMS. As remarked at the beginning of this section, MySQL is an extremely popular database with Web programmers, precisely because it is easy to use and requires little in the way of ongoing care and feeding. If, after some period of time, you outgrow MySQL, it might be time to consider PostgreSQL, discussed in the next section.

Table 15-1 MySQL Client Programs

PROGRAM	DESCRIPTION
<code>mysql</code>	Provides an interactive command interpreter for the MySQL server
<code>mysqlaccess</code>	Adds new users to MySQL
<code>mysqladmin</code>	Performs MySQL administrative functions
<code>mysqlbinlog</code>	Displays a MySQL binary log file in a format readable by humans
<code>mysqlbug</code>	Creates and files bug reports for MySQL
<code>mysqlcheck</code>	Tests, repairs, analyzes, and optimizes MySQL databases
<code>mysqldump</code>	Backs up or restores data from or to a MySQL database
<code>mysqldumpslow</code>	Displays and summaries MySQL's query log, producing information you can use to optimize slow queries
<code>mysqlimport</code>	Imports data into MySQL tables from text files of various formats
<code>mysqlshow</code>	Displays the structure of MySQL databases, tables, and columns
<code>mysqltest</code>	Runs a database test and compares the results to previous runs

Using PostgreSQL

PostgreSQL is the second most popular free RDBMS. It provides some features not available in MySQL, so if you find you need features or functionality that MySQL lacks, PostgreSQL might be the solution you need. As with MySQL, PostgreSQL is popular with Linux users because it is free; fast; feature-rich; easy to set up, use, and maintain; and provides fuller support for the ANSI SQL99 and SQL 2003 standards than MySQL does. Like MySQL, PostgreSQL is also widely supported by and integrated into a variety of third-party applications. There are numerous Apache modules that make it possible to use PostgreSQL in Apache-based Web servers, and PHP's support for PostgreSQL is surpassed only by PHP's support for MySQL. Among scripting languages, Perl and Python have wide support for PostgreSQL, and PostgreSQL's client API makes it possible and reasonably easy to include PostgreSQL support in C and C++ applications.

Out of the box, PostgreSQL is ready to use. You'll need to make sure that it is installed of course, and there are some postinstallation tasks you need to perform to secure the database and to make sure the database is functioning and answering requests. This section will also show you, briefly, how to use some of the PostgreSQL client commands.

Why would you want to use PostgreSQL instead of MySQL? The easiest answer is that you should use PostgreSQL if it has a feature or functionality that MySQL doesn't. If you are looking for standards compliance, PostgreSQL is more compliant with SQL standards than MySQL is and supports certain types of SQL queries that MySQL doesn't. Traditionally, the biggest knock against MySQL was that it was just a glorified data file (an ISAM or *index sequential access method* file, to be precise) that supported SQL-driven data access. PostgreSQL, on the other hand, while providing persistent data storage using the file system, used to have a different in-memory layout to support SQL-driven data access. This distinction is no longer true because MySQL now provides multiple methods of persistent data storage and is no longer an ISAM-based one-trick pony.

PostgreSQL is more marketing-buzzword-compliant, too, in that it supports spatial data types and is object-relational. The spatial data types make it possible to create GIS applications using PostgreSQL. *Object-relational* means that PostgreSQL can use standard SQL access methods and relational data structures to access and manipulate object-oriented data. To provide some guidance, we have prepared a sidebar, "MySQL or PostgreSQL," that provides a side-by-side comparison of the two packages.

To return to the original question, which one should you use? We can't tell you. As a system administrator, these concerns are ordinarily peripheral to your primary job function. You maintain the system on which the database

MYSQL OR POSTGRESQL?

If you want to start an argument among in a group of people familiar with free RDBMSes, ask them which is better, PostgreSQL or MySQL. It is not this chapter's intent to start an argument, so it avoids saying which is better. There are significant differences between MySQL and PostgreSQL, though, and knowing what these differences are might help you decide which one to use. Table 15-2 lists features generally expected to exist in a RDBMS and shows whether MySQL and PostgreSQL *as shipped in Fedora Core and RHEL* support them.

As you can see in the table, PostgreSQL supports a larger set of features common in the commercial RDBMS world than MySQL. However, bigger isn't necessarily better because the richer feature set might be overkill for your needs. In addition, the versions of PostgreSQL and MySQL that ship in Fedora Core and Red Hat Enterprise Linux lag somewhat behind the current stable versions of those products. At the time this book went to press, the versions of PostgreSQL and MySQL shipping with Fedora Core and RHEL were 7.4.7 and 3.23.58, respectively, while the latest and greatest released versions were 8.0 and 4.1.9 (MySQL 5.0 had just entered an alpha release state).

For a fuller comparison of the features set of particular version PostgreSQL and MySQL, see the comparison table maintained by MySQL at <http://dev.mysql.com/tech-resources/features.html>.

runs and possibly install/upgrade the software and perform the initial configuration. It is up to information architects and database administrators (DBAs) to make decisions about which database to use and the relative merits of one database or another. Of course, not every site running Linux has the luxury of this kind of separation of duties. The system administrator of smaller sites is often also the DBA (and the network administrator, mail administrator, Webmaster, telephone technician, and brewer of the morning coffee), so it pays to be familiar with the broad outlines of database features.

Table 15-2 Database Feature Comparison

FEATURE	MYSQL	POSTGRESQL
ACID compliance	Yes	Yes
Aggregate functions	Yes	Yes
ANSI SQL compliance	Incomplete	Yes
API for custom applications	Yes	Yes
Complex queries (UNION, UNION ALL, EXCEPT)	Yes	Yes
Cross-database compatibility features	Yes	Yes

(continued)

Table 15-2 *(continued)*

FEATURE	MYSQL	POSTGRESQL
Views	No	Yes
Default column values	No	Yes
Dynamically loadable extensions	No	Yes
Extensible, user-defined data types	No	Yes
Foreign keys	Yes	Yes
Functional indexes	No	Yes
Functions	Yes	Yes
Hot stand-by	No	Yes
Index methods	Yes	Yes
Inheritance	No	Yes
Kerberos support	No	Yes
Locking granularity	Yes	Yes
ODBC support	Yes	Incomplete
Outer joins	Yes	Yes
Partial indexes	Yes	Yes
Procedural languages	Yes	Yes
Referential integrity	Yes	Yes
Replication	Yes	Yes
Rules	No	Yes
Sequences	Yes	Yes
SSL support	Yes	Yes
Stored procedures	No	Yes
Sub-selects	Incomplete	Yes
Transactions	Yes	Yes
Triggers	No	Yes
Unicode support	Yes	Yes

Assuming that you've decided that PostgreSQL is the database to use, the next two sections show you how to get the standard PostgreSQL installation working and how to use some of PostgreSQL's client utilities.

Verifying the PostgreSQL Installation

You won't get very far in this section if PostgreSQL is not installed. You can use the following commands to see if the key PostgreSQL RPMs are installed:

```
# rpmquery postgresql-server
postgresql-server-7.4.7-1.FC3.2
# rpmquery postgresql
postgresql-7.4.7-1.FC3.2
# rpmquery postgresql-libs
postgresql-libs-7.4.7-1.FC3.2
# rpmquery postgresql-devel
postgresql-7.4.7-1.FC3.2
```

The `postgresql-server` package contains the core PostgreSQL database server. It is required to create and maintain a PostgreSQL database. The `postgresql` package installs the client utilities, which you will need to do anything with the server. Similarly, the `postgresql-libs` package installs shared libraries used by all PostgreSQL clients and interfaces; you must have this package installed to be able to connect to the server and to use any other PostgreSQL package. `postgresql-devel`, another required package, provides the header files and shared libraries required to create C and C++ programs that interact with PostgreSQL databases. It also includes a C preprocessor to use against C and C++ programs that use the PostgreSQL API. If these four packages aren't installed, install them as described in Chapter 30.

Other PostgreSQL packages that might also be installed or that you might want to install include:

- **postgresql-contrib** — Includes selected contributed modules and programs not part of the standard PostgreSQL distribution
- **postgresql-docs** — Provides a rich documentation suite in both source (SGML) and rendered formats suitable for online viewing or printing
- **postgresql-jdbc** — Installs a Java database connectivity (JDBC) driver necessary to connect to PostgreSQL using Java
- **postgresql-odbc** — Installs the Open Database Connectivity (ODBC) driver necessary to connect to PostgreSQL using ODBC
- **postgresql-pl** — Contains PostgreSQL-specific procedural languages for Perl, Tcl, and Python, enabling you to use these languages to manipulate the server
- **postgresql-python** — Includes Python support, and the PL/Python procedural language for using Python with PostgreSQL

- **postgresql-tcl** — Provides Tcl (Tool command language, an embeddable scripting language) support, the PL/Tcl procedural language, and a PostgreSQL-enabled `tclsh` (a Tcl shell)
- **postgresql-test** — Contains a number of test suites for performing benchmark and regression tests against the PostgreSQL server

In addition to the packages in the preceding list, other RPMs provide PostgreSQL-related functionality that you likely won't need. To keep this section simple, we will only refer to programs and utilities provided by the four required packages.

Finalizing the PostgreSQL Installation

On a fresh PostgreSQL installation, no data structures have been created. Rather, the software has been installed, the `postgres` user and group have been created, and the data directory, `/var/lib/pgsql/data`, has been created. The steps you need to take to finalize the installation are:

1. Initialize the installation.
2. Modify access privileges.
3. Create a test database.
4. Validate connectivity to the test database.

The following sections describe each step in this process in more detail.

Initializing the Installation

Use the following procedure to initialize the installation, which consists of creating template data structures and starting the database server:

1. Become the `postgres` user using `su`. You do this in two steps, first `su`-ing to the root account and then `su`-ing to the `postgres` user account:

```
$ su - root
Password:
# su - postgres
-bash-3.00$
```

2. Set the environment variable `$PGDATA` to point to: `/var/lib/pgsql/data`.

```
$ export $PGDATA=/var/lib/pgsql/data
```

Most PostgreSQL commands read `$PGDATA` to know where to find the database. If you don't set it, you'll continually have to add an argument

like `-D /var/lib/pgsql/data` to all of the PostgreSQL commands you use. It gets tedious and is error-prone, so set the environment variable and forget about it.

3. Create the database cluster. A *database cluster* refers to the data directory and supporting files and directories stored therein, which serve as a template used to create the databases managed by a single PostgreSQL server (yes, you can have multiple PostgreSQL servers, but we aren't going to go there):

```
-bash-3.00$ initdb
```

The files belonging to this database system will be owned by user "postgres".

This user must also own the server process.

The database cluster will be initialized with locale en_US.UTF-8.

```
fixing permissions on existing directory /var/lib/pgsql/data... ok
creating directory /var/lib/pgsql/data/base... ok
creating directory /var/lib/pgsql/data/global... ok
creating directory /var/lib/pgsql/data/pg_xlog... ok
creating directory /var/lib/pgsql/data/pg_clog... ok
selecting default max_connections... 100
selecting default shared_buffers... 1000
creating configuration files... ok
creating template1 database in /var/lib/pgsql/data/base/1... ok
initializing pg_shadow... ok
enabling unlimited row size for system tables... ok
initializing pg_depend... ok
creating system views... ok
loading pg_description... ok
creating conversions... ok
setting privileges on built-in objects... ok
creating information schema... ok
vacuuming database template1... ok
copying template1 to template0... ok
```

Success. You can now start the database server using:

```
/usr/bin/postmaster -D /var/lib/pgsql/data
or
```

```
/usr/bin/pg_ctl -D /var/lib/pgsql/data -l logfile start
```

If you didn't set the value of the environment variable `$PGDATA` as recommended in Step 2, you must add `-D /var/lib/pgsql/data` to the `initdb` command line to specify the location of the database cluster.

`/var/lib/pgsql/data` is the default, but you can use any directory. The initialization process ensures that only the `postgres` user (and root, of course) has any access whatsoever to the database cluster.

4. Exit the `postgres su` session because the root user must perform the next step:

```
-bash-3.00$ exit
logout
```

5. Start the database server. You can use the commands shown at the end of Step 3, but it is easier to use the initialization script, `postgresql`, which performs the same steps and also executes some sanity checks before starting the server.

```
# service postgresql start
Starting postgresql service: [ OK ]
```

With the PostgreSQL server running, you're ready to proceed to the next part of the process, tightening up access to the server.

Modifying Access Privileges

After you have initialized the installation, you will likely want to modify the default authentication scheme. The default authentication scheme is called *trust authentication* because it permits all local users to access the server using any PostgreSQL-recognized username (including the PostgreSQL superuser account). Moreover, this access can use either UNIX-domain sockets (also known as *Berkeley sockets*) or TCP/IP. We suggest making of the following modifications to the default access policy:

- Permit local access using only UNIX-domain sockets.
- Require local users to connect to the server using their system login accounts.
- Require remote users (connecting via TCP/IP) to use SSL.
- Use strong encryption for password checking.

The file `/var/lib/pgsql/data/pg_hba.conf` controls client authentication. It contains records that have one of three formats. The first format addresses authentication of local clients, that is, clients accessing the server from same machine on which the server is running (`localhost`). The local access format has the following general form:

```
local database user auth [option]
```

`database` identifies the database to which the record applies. It can be one of `all`, which, you guessed it, applies this rule to all databases; `sameuser`, which means that the database being accessed must have the same name as the connecting user; `samegroup`, which means that the database being accessed must have the same name as the group name of the connecting user; or a comma-separated list of one or more names of specific PostgreSQL databases. `user` identifies the user to which the authentication record applies. Like `database`, `user` can be `all` (meaning all users), a username, a group name prefixed with `+`, or a comma-separated list of either user or group names.

`auth` specifies the manner in which connecting clients will be authenticated. Table 15-3 lists the possible authentication methods.

`option` applies options to the specified authentication method and will be either the name of file mapping IDENT-generated usernames to system usernames if you are using PostgreSQL's `ident` authentication method or the name of the PAM service to use if you are using PostgreSQL's `pam` authentication method.

Table 15-3 PostgreSQL Authentication Methods

METHOD	DESCRIPTION
<code>crypt</code>	Like the <code>password</code> method, but uses the <code>crypt()</code> library function to encrypt passwords for transmission across the network
<code>ident</code>	Implements authentication using the connecting user's identity as reported by the IDENT protocol (requires the <code>identd</code> daemon)
<code>krb4</code>	Uses Kerberos V4 for authentication, but available only for TCP/IP connections
<code>krb5</code>	Uses Kerberos V5 for authentication, but available only for TCP/IP connections
<code>md5</code>	Like the <code>password</code> method, but uses MD5-based encryption to foil packet sniffers
<code>pam</code>	Adds Pluggable Authentication Modules (PAM) support to the <code>password</code> method
<code>password</code>	Permits clients to connect if the supplied password, transmitted as clear text, matches the password assigned to the connecting user account
<code>reject</code>	Rejects all access
<code>trust</code>	Allows any user with a system login account to connect to the server using any PostgreSQL user account

PostgreSQL's default authentication method for local users is `trust`. The entire rule looks like the following:

```
local all all trust
```

As it is, maintainers of the PostgreSQL packages for Fedora Core and RHEL have changed this default to:

```
local all all ident sameuser
```

Changing the authentication method to `ident` for local connections means that PostgreSQL will use the IDENT protocol to determine the PostgreSQL user account. To put it another way, if the authentication method for local connections is `ident`, PostgreSQL uses the local system's IDENT server to obtain the name of the user connecting to the server. Adding the authentication option `sameuser` means that the connecting user must have a system login account.

The following rules implement three of the restrictions for local connections suggested earlier (local access only through UNIX-domain sockets, local users connect using their system login accounts, use strong encryption):

```
local all all md5
host all all 127.0.0.1 255.255.255.255 reject
```

In the first rule, the authentication method is `md5`, which requires strong encryption. The second rule rejects (the `reject` authentication method) all users connecting from the host (more about `host` rules in a moment) whose IP address is 127.0.0.1, that is, all users connecting from localhost *via a TCP/IP connection*. Records of the `local` type control connection from UNIX-domain sockets, so the second rule does not affect connection originating from the local machine. In any event, connections from the local machine are explicitly permitted by the first rule, which takes precedence over the second rule.

TIP PostgreSQL access rules do not “fall through.” Rather, rules are evaluated until a match occurs, at which point evaluation stops and the matching rule is applied. Accordingly, the order in which access rules appear in the `pg_hba.conf` is important.

Rules affecting TCP/IP connections have the following general format:

```
type database user ipaddr ipmask auth [option]
```

The database, user, auth, and option values have the same semantics as described for local connections. The type value must be one of `host`,

`hostssl`, or `hostnossl`. `host` matches any connection coming in via TCP/IP, whether it uses SSL nor not. `hostssl` matches only TCP/IP connections that use SSL. `hostnossl` matches only TCP/IP connections *not* using SSL. The `ipaddr` (an IP address) and `ipmask` (an IP net mask) options enable you to control in a very finely grained manner the remote hosts that can connect to the server. For example, the `ipaddr ipmask` pair of `127.0.0.1 255.255.255.255` refers to only the IP address `127.0.0.1`, where as the `ipaddr ipmask` pair `127.0.0.0 255.255.255.0` refers to all IP addresses between `127.0.0.1` and `127.0.0.254`. IP addresses must be specified in standard numeric (or *dotted quad*) format; host and domain names do not work.

So, to implement the recommended restrictions for clients connecting via TCP/IP (you must use SSL, use strong encryption), the following rules should suffice:

```
hostnossl all all 0.0.0.0 0.0.0.0 reject
hostssl all all 0.0.0.0 0.0.0.0 md5
```

The first rule rejects all connections from any remote host not connecting via SSL. The second rule permits all SSL-based connections for all users to all databases and uses MD5 authentication. The second rule is still too liberal if you want to restrict access to specific hosts or domain, however. To permit access to the finance database for all users on the finance subnet, which has the IP address `192.168.2.0`, the rule `hostssl finance all 192.168.2.0 255.255.255.0 md5` will do. It should replace the previous `hostssl` rule, so the rule set becomes:

```
hostnossl all all 0.0.0.0 0.0.0.0 reject
hostssl finance all 192.168.2.0 255.255.255.0 md5
```

If you want to reject all other remote connections, use add the following rule:

```
hostssl all all 0.0.0.0 0.0.0.0 reject
```

Thus, the final rule set looks like the following:

```
hostnossl all all 0.0.0.0 0.0.0.0 reject
hostssl finance all 192.168.2.0 255.255.255.0 md5
hostssl all all 0.0.0.0 0.0.0.0 reject
```

The evaluation sequence firsts rejects all TCP/IP connections not using SSL. The second rule permits any client passing the first test to connect to the finance database if the client has an IP address between `192.168.2.1` and `192.168.2.254` (inclusive). If a match occurs at this point, rule evaluation stops. Otherwise, the next rule is evaluated, which rejects all other incoming TCP/IP

connections, even if they use SSL. In practice, you will likely find that it is easiest to permit access to specific users and databases based on IP address or, in the case of local connections, login account names.

To make the access rule changes take affect, you need to reload the access control file. You can do this using the `service` utility, as shown in the following example:

```
# service postgresql reload
```

Alternatively, execute the following command as the `postgres` user:

```
-bash-3.00$ pg_ctl reload
postmaster successfully signaled
```

`pg_ctl` is a simple utility for starting, stopping, reloading, and checking the status of a running PostgreSQL database. For security purposes, only the user under which the PostgreSQL server runs (`postgres` on Fedora Core and RHEL systems) should invoke PostgreSQL command directly in this manner. After reloading the access control file, you'll want to create a test database to confirm that the server is working properly.

Creating a Test Database

So far, so good. You've initialized the database server and tightened up access to it. The next step is to create a test database so that you can validate that the server is functioning and that your access control rules work as you intended. Without going into the gruesome details, the `initdb` command you executed earlier created an initial database, named `template1`, which, as the name suggests, serves as a template or model for subsequent databases. Ordinarily, you never want to modify the template database because it is essentially cloned when a new database is created, so changes made to the template apply to all databases created from it. As you might guess, though, prudently chosen modifications to `template1` can be used to override PostgreSQL defaults that you might dislike. The task in this chapter is getting the server up and running, so we'll adroitly sidestep this issue and create a database using PostgreSQL's default settings.

PostgreSQL provides a utility named `createdb` that you can use to create a database. Its syntax is refreshingly simple:

```
createdb [opt ...] [dbname] [desc]
```

Notice that all of the arguments are optional. If executed with no arguments, `createdb` creates a new database named for the user executing the command. This is not what you want in most situations. `dbname` specifies the name of the database you want to create. `desc` is a comment or description associated with

the database. `opt` supplies one or more options that either affect `createdb`'s behavior or that are passed to the server to specify certain characteristics of the database created. The following options most immediately concern:

- **-D path** — Creates the database in `path` rather than the default location, `$PGDATA`
- **-e** — Echoes the SQL commands sent to the server to create the database
- **-O owner** — Assigns `owner` rather than the user executing `createdb` as the database owner

The following command creates a test database named `rhlnsa3` (witty, huh?) and adds a short description. You should execute this command as the `postgres` user:

```
-bash-3.00$ createdb -e rhlnsa3 "Test database for chapter 15"
CREATE DATABASE rhlnsa3;
CREATE DATABASE
COMMENT ON DATABASE rhlnsa3 IS 'Test database for chapter 15';
COMMENT
```

You can use single or double quotes around the string used to create the description. If you are unfamiliar with SQL, using the `-e` option to echo the SQL commands sent to the server is useful. The actual commands sent appear with terminating semicolons (;). In the absence of the `-e` option, you would see only summaries of the SQL statements executed, as illustrated in the following example:

```
-bash-3.00$ createdb rhlnsa3 "Test database for chapter 15"
CREATE DATABASE
COMMENT
```

To facilitate testing, create a new database user using the `createuser` utility, which is a wrapper around the SQL statements necessary to add a user. The syntax is simple:

```
createuser [-P [-E]] username
```

This command creates a database user named `username`. To assign a password, use the `-P` option. To have the assigned password encrypted, specify `-E`. Consider the following example:

```
-bash-3.0.0$ createuser -P -E bubba
Enter password for new user:
Enter it again:
Shall the new user be allowed to create databases? (y/n) n
Shall the new user be allowed to create more new users? (y/n) n
CREATE USER
```

This example creates a new database user named `bubba`, assigning an encrypted password as part of the process. The last two prompts ask if you want `bubba` to be able create new databases and create other users. `bubba` is only a test user, so he doesn't get any special treatment. Recall that you are using `ident` authentication with the `sameuser` authentication option, which means that the user created, `bubba` in this case, must also have a login account on the system.

Testing Connectivity to the Test Database

The final step of the PostgreSQL initialization involves using the test user (`bubba`) to connect to the database. While you have already established that the `postgres` user can connect to the server when you created the test database and the test user, it is important to make sure that normal users can also connect to the server and that the access rules you create work as you intend. To test the database server, use the following procedure:

1. Become the user for whom you created the database user account:

```
$ su - bubba
```

```
Password:
```

```
[bubba]$
```

You have to become `bubba` because `ident`-based authentication is in use, which means you have to be the same user as you use to connect to the database.

2. Use the `psql` command shown in the following example to connect to the `rhlnsa3` database:

```
[bubba]$ psql -W rhlnsa3
```

```
Password:
```

```
Welcome to psql 7.4.6, the psql interactive terminal
```

```
Type: \copyright for distribution terms
```

```
       \h for help with SQL commands
```

```
       \? for help on internal slash commands
```

```
       \g or terminate with semicolon to execute query
```

```
       \q to quit
```

```
rhlnsa3=>
```

`psql` is the PostgreSQL's shell or command interpreter. Notice how the default prompt is the name of the database followed by `=>`. Depending on the activity you are performing, the second-to-last character of the prompt changes.

3. Use the following SQL commands to create a new table:

```
rhlnsa3=> create table chapters (
```

```
rhlnsa3(> chapnum int,
```

```

rhlnsa3(> title varchar(80),
rhlnsa3(> pages int
rhlnsa3(> );
CREATE TABLE
rhlnsa3=>

```

Notice how opening a parenthesize statement causes the shell prompt to change from => to (> to indicate that it's waiting for matching closing parenthesis. The terminating semicolon is required.

4. Add data to the table using the following SQL statements:

```

rhlnsa3=> insert into chapters (chapnum, title, pages)
rhlnsa3-> values (15, 'Configuring a Database Server', 35);
INSERT 17148 1

```

In this case, the shell prompt became ->, indicating that it is waiting for a closing semicolon to terminate the SQL command.

5. Use the following query to retrieve the data you just added:

```

rhlnsa3=> select * from chapters;
chapnum | title | pages
-----+-----+-----
      15 | Configuring a Database Server | 35
(1 row)

```

6. Exit the PostgreSQL shell:

```

rhlnsa3=> \q

```

You can also use Ctrl+d to exit the PostgreSQL shell.

If the procedures described in the last few sections worked, your database server is up and running and working the way it should. The next section introduces a few of the PostgreSQL client programs that you will at least want to know exist.

Using the PostgreSQL Client Programs

PostgreSQL's client programs, like MySQL's, implement a user interface to the server. `psql`, as you just learned, provides the most complete and direct access to the server but requires you to know at least some SQL. Other utilities, like `createdb`, `createuser`, and their analogs, `dropdb` (for deleting a database) and `dropuser` (for deleting a user) are wrapper scripts that invoke SQL statements for you. If you are not a database guru (or don't want to be), you'll probably be most comfortable using the various wrapper utilities. Table 15-4 lists some the PostgreSQL client programs with which you will want to be familiar.

Table 15-4 PostgreSQL Client Programs and Utilities

PROGRAM	DESCRIPTION
<code>createdb</code>	Creates a new database in the database cluster
<code>createuser</code>	Creates a new user in the database cluster
<code>dropdb</code>	Deletes a database from the database cluster
<code>dropuser</code>	Deletes a user form the database cluster
<code>pg_dump</code>	Saves (<i>dumps</i>) the contents of a database or database object to a file
<code>pg_restore</code>	Reloads a database or database object using a file created by <code>pg_dump</code>
<code>psql</code>	Provides a command interpreter for interacting with a database server

Most PostgreSQL clients share a number of options in common. For example, `-U username` specifies the username to use when connecting to the server. `-W` indicates you should be prompted for a password. `-D /path` specifies the path to the PostgreSQL database cluster, which defaults to the value of the environment variable `$PGDATA` or the compiled in default (`/var/lib/pgsql/data`). `-e` causes wrapper scripts like `createdb` and `dropuser` to echo to standard output (`stdout`) the SQL commands used. Commands that operate on or require a connection to a specific database usually accept a database name as an argument. If a database name is not specified, it defaults to the name of the user connecting to the database specified using `-U username` or to the connecting user's login name if `-U username` is not specified.

Similarly, most PostgreSQL clients can use PostgreSQL-specific environment variables to set certain defaults. In addition to `$PGDATA`, which you've already seen, the variable `$PGDATABASE` stores the name of the database to use (or create or drop) unless overridden on the command line. `$PGHOST` specifies the name of the host on which the database server is running and so is usually used when connecting to a database running on a remote host. `$PGUSER` defines the default connection parameters, which usually consists of the PostgreSQL user name.

You've already seen the syntax for `createdb` and `createuser`, so we won't review it here. `dropdb` and `dropuser` drop (delete) a database or user, respectively, from the database cluster. `dropdb`'s syntax is:

```
dropdb [option ...] dbname
```

`dbname` identifies the database to drop and `option` (there can be multiple options) accepts the options previously described and, most importantly, the

`-i` option, which asks for confirmation before actually dropping the database. This is an important option because dropping a database deletes the data and data structures associated with the database. Unless you have previously saved the data, dropping a database is a permanent action. Because it is such a drastic action, only the database owner and the PostgreSQL superuser have privileges to drop a database. As a precaution, you should always save the database data before dropping it (use the `pg_dump` command, about which you'll learn shortly).

The `dropuser` wrapper utility deletes a specified user from the database cluster. Its syntax is:

```
dropuser [option ...] username
```

Replace `username` with the name of the PostgreSQL user account you want to delete. The possible values for `option` were described earlier and won't be repeated here, except to add the `dropuser`, like `dropdb`, also accepts the `-i` option to request interactive use.

Before dropping a database, you should use the `pg_dump` program to save the data unless you are absolutely, positively, 100 percent certain beyond the slightest shadow of a doubt that you won't need the data in the future. `pg_dump`'s syntax is:

```
pg_dump [option ...] dbname
```

As usual, `dbname` specifies the name of the database to dump. `option` controls the actual dump behavior. The various options already described also work with `pg_dump`, but it has a number of options specific to its behavior that you'll want to know about. `pg_dump` is usually used to archive and retrieve data, such as for backup and upgrade purposes, so the options discussed focus on that purpose.

A typical archive/restore operation consists of dumping the database, dropping it, recreating it, and reloading it with the dumped data. Using the `-C` option, `pg_dump` will begin the dump output with the SQL statements necessary to create the database and then connect to it. The rest of the dump will consist of `COPY` statements that use a PostgreSQL-specific extension for performing high-speed data loads. Use the `-f filename` option to specify the name of the output file. If you don't use this option, output goes to `stdout` and must be redirected to a file using the shell's `>` operator. To create an output file, finally, most suitable for use with `pg_restore` (described next), use the `-Fc` option, which specifies a custom output format specifically designed for use with `pg_restore` (`-Fp` outputs a plain ASCII text file with data and SQL statements and `-Ft` outputs a tar archive that `pg_restore` can read).

If you want to dump only the database schema (the actual design of the database, not its data) specify the `-s` option. Similarly, if you are interested in only the contents of a particular table, specify `-t table`, where `table` is the name of the table that interests you.

`pg_restore` is `pg_dump`'s counterpart and restores or reloads a PostgreSQL database dump created with `pg_dump`. It also accepts the same command-line options as `pg_dump`, so your learning curve has flattened out considerably. The difference between the two is that `pg_restore`'s argument is the name of an input file created by `pg_dump`.

So, given the following `pg_dump` command:

```
-bash-3.00$ pg_dump -Fc -C rhlnsa3 > rhlnsa3.db
```

You can restore the contents of the tables in the database `rhlnsa3` using the following `pg_restore` command (the `rhlnsa3` database must exist) after dropping any tables in the database:

```
-bash-3.00$ pg_restore -d rhlnsa3.dump
```

Table 15-5 lists the PostgreSQL server programs you'll want to know how to use.

You will rarely, if ever, need to invoke the `postgres` command directly. It is called by the `postmaster` command. `postgres` is responsible for processing queries for a single connection to the database server. When a connection starts, `postmaster`, which listens for incoming connection requests, starts a `postgres` process to handle that connection. In addition to serving as the multiuser server for a PostgreSQL databases, `postmaster` is also responsible for handling communication between individual `postgres` processes. You can also almost always rely on the PostgreSQL initialization script, `postgresql`, to start and stop the `postmaster` service, so you should hardly ever need to execute `postmaster` directly.

Table 15-5 PostgreSQL Server Programs and Utilities

PROGRAM	DESCRIPTION
<code>initdb</code>	Creates and initializes a PostgreSQL database cluster
<code>pg_ctl</code>	Controls a running database server instance
<code>postgres</code>	Processes queries for a single connection to a PostgreSQL database (usually started by <code>postmaster</code>)
<code>postmaster</code>	Starts <code>postgres</code> processes to handle incoming database connections and coordinates communication between <code>postgres</code> processes

Summary

Linux-based database servers are becoming as ubiquitous as Linux-based Web and email servers, so it is likely that you will need to create a database server at some point. While you don't need to be on a first-name basis with the finer points of database administration to configure a database server, it *does* help to be familiar with the general process. Just installing the software isn't usually sufficient, either. As an administrator, you usually need to be able to make sure that the database is accessible (or not as the case might be), that the initial accounts are secure, and that the server is working. This chapter showed you how to configure MySQL and PostgreSQL to a basic level of functionality. As you learned, installing them is easy, but the postinstallation configuration and testing can be tedious. The good news is that database installations are usually performed in conjunction with a DBA who provides guidelines and instructions for you to follow. You're rarely on your own.

Creating a VNC Server

IN THIS CHAPTER

- What Is VNC?
- Setting Up a VNC Server
- Testing the VNC

Providing network services for remote employees, whether they are road warriors barricaded in a hotel room or telecommuters working from a home office, is nothing new. As telecommuting becomes more common and broadband Internet access more widespread, system administrators are increasingly being asked to provide remote access to their networks. Various approaches to providing LAN services to disconnected employees have been tried over the years, including remote control software virtual private networks (VPNs). The goal has always been to make it possible for remote employees to use another computer or LAN-based services as if those employees were sitting in the office. VNC gives you remote access to an existing desktop system and all of the resources that it can access. This chapter describes how use Fedora Core and RHEL to create a VNC server, enabling telecommuters and other remote employees to access a Fedora Core- or RHEL-based LAN and LAN-based services. It also shows you how to configure a Linux system as a VNC client.

What Is VNC?

Just to get the acronym out of the way, VNC stands for virtual network computing, and it provides a way to fully control one computer from any other computer or similarly capable device situated anywhere on the Internet. One of

the virtues of VNC solutions over other methods is that VNC is cross-platform. You can access and control a Linux host from a Windows PC, something not possible with products like PCAnywhere. Another VNC advantage is that the protocol is optimized for Internet transmission, so it is actually possible and not glacially slow to run X applications across the Internet.

In addition to providing remote users access to LAN-based systems and services, VNC also has clear uses for help desk technicians, other technical support professionals, and system administrators. If you ever worked at a help desk, you know how difficult it is to get the “helpees” on the other end of the telephone to tell you what the screen looks like, what applications are running, and even to coherently describe the problem they’re having. You also know how hard it is to make sure that the person you’re helping types the commands you want typed or executes the problem resolution procedure properly. Using VNC, you have immediate access to the user’s system and can remotely diagnose and troubleshoot the problem at hand, all without having to leave your desk. VNC is a real boon for system administrators for the same reasons. Even if out of the office or at a remote site, the administrator always has access to management and monitoring tools that can make detecting and fixing a troublesome server a trivial undertaking.

Unlike older, less capable remote control software, VNC supports multiple incoming connections to the same VNC server, so it is possible for several users to connect to the same system and work collaboratively on a project, such as a presentation. Alternatively, you can use a single VNC server as an access concentration point from which properly authorized users can connect to other systems. Another difference between VNC and other remote control protocols is that VNC, described as “remote display protocol” or an *RFB* (remote framebuffer), is an open and nonproprietary protocol, whereas the other products (from Symantec, Citrix, Insignia Solutions, and Microsoft) are closed protocols that are closely tied to the Windows GUI.

What VNC is *not* is a VPN, or *virtual private network*. Speaking broadly, a VPN is a network configuration in which a main internal network has remote nodes (such as telecommuting employees) that use a VPN running over (perhaps it is better to say *across*) the Internet to access the main internal network. To achieve this, a secure (encrypted) tunnel is created between the main network and the remote nodes, and IP traffic is routed through that tunnel. VNC, on the other hand, while it *can* be used across a VPN and uses the Internet to transport packets back and forth, is usually used to provide access to a single system and to allow a remote user full control over that system. More succinctly, VPN makes a remote system part of the main network; VNC gives a remote user control over one system on the main network.

For more information about VNC, two of the best resources are the excellent VNC overview at uk.research.att.com/pub/docs/att/tr.98.1.pdf

and the RealVNC Web site (realvnc.com/). There is a connection between AT&T and RealVNC. Researchers at the AT&T UK research labs created VNC. RealVNC was formed by some of the AT&T researchers as a venture to commercialize VNC technology. The VNC software is licensed under the GPL, so RealVNC's business model depends on providing support, service, and value-added software.

Setting Up a VNC Server

In this context, a VNC server is the machine you want to access remotely. So, if you're at home and want to connect to the Linux system on your desk at work, the system at work is the server; the system at home is the VNC client. Figure 16-1 illustrates this arrangement.

To set up a VNC server, you'll need to install the `vnc-server` and `vnc` packages. The commands shown below will show you if they are installed:

```
$ rpmquery vnc-server
vnc-server-4.1.1-10
$ rpmquery vnc
vnc-4.1.1-10
```

If these packages are not installed, install them before proceeding.

Starting the VNC server is simplicity itself: execute the Perl script `vncserver` as a mortal user. `vncserver` is a wrapper script that handles the persnickety details of starting `Xvnc`, the X Window System VNC server. Speaking more precisely, `Xvnc` creates a VNC desktop on the server system to which VNC clients can connect. There is a configuration step that must be performed by the root user, but starting the server does not require any special privileges. The first time you start `vncserver`, you have to set the password that connecting clients must issue, as shown in the following example:

```
$ vncserver

You will require a password to access your desktops.

Password:
Verify:

New coondog.example.com:1 (bubba)' desktop is coondog.example.com:1

Creating default startup script /home/bubba/.vnc/xstartup
Starting applications specified in /home/bubba/.vnc/xstartup
Log file is /home/bubba/.vnc/coondog.com:1.log
```

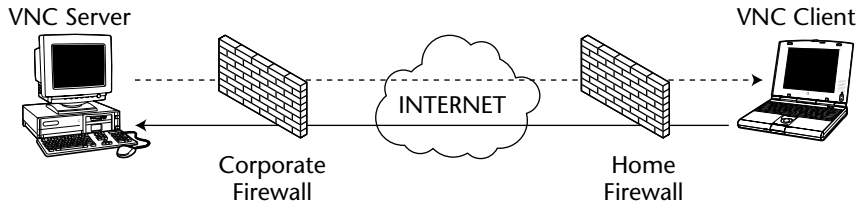



Figure 16-1 Typical VNC client and server configuration.

The output is important. The first line after setting the password indicates that `Xvnc` has created a new display, `:1` on the host `coondog.example.com`. You will need this information when you connect from the client system. `vncserver` asks for a password only the first time you start it. You can change the password later using the command `vncpasswd` or by removing the file `$HOME/.vnc/passwd`.

The next two lines tell you that a startup script, `/home/bubba/.vnc/xstartup`, has been created and that the script has been executed, that is, the applications it specifies are running on the `Xvnc` display. This means that when you connect to the VNC server, the client will have those applications already running. This also means that if you want to customize the desktop provided by the server, you can edit the `xstartup` file. Finally, `vncserver` tells you where to find the log file it creates, which will simplify troubleshooting the VNC server problems if you encounter any. When `vncserver` completes, a simple, unadorned VNC desktop is ready to accept connections.

Configuring Your Firewall for VNC

Well, your VNC server is *almost* ready to accept connections. VNC listens on port 5500 plus the X display number for incoming VNC client sessions. On a properly secured system, these ports are blocked at the firewall. You have to punch a hole in the firewall for that port so that VNC clients can get through to the server. This configuration step requires root access because you need to use the Security Level Configuration tool to modify your system's firewall setup (you *are* running a firewall, right?).

1. To start the Security Level Configuration tool, select Red Hat ⇨ System Settings ⇨ Security Level or type **`system-config-securitylevel`** at a command prompt. Figure 16-2 shows the Security Level Configuration tool's main screen.

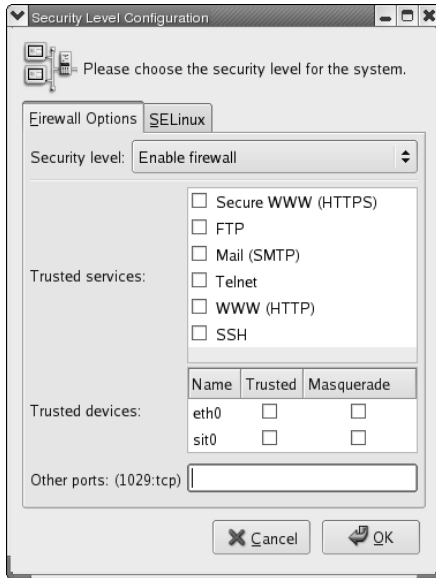


Figure 16-2 The Security Level Configuration tool.

The firewall configuration shown in Figure 16-2 is tight: no external access of any sort is permitted on this machine. You're about to change this. In the Other ports: (1029:tcp) text box, type `5901:tcp`. By default, VNC uses ports numbered 5900 plus the display number. In this example, the display number is `:1`, so the port number is 5901. If you were using display number `:21`, the port number would be 5912. The `:tcp` portion of the port number tells the firewall to open port 5901 for TCP connections because the remote framebuffer protocol uses TCP, not UDP.

2. After you have entered the appropriate port number (see Figure 16-3), click OK to save your change and close the Security Level Configuration tool. Click Yes when the tool warns you that you are about to overwrite your existing firewall configuration.

VNC clients can now access the VNC server, so the server configuration is complete.

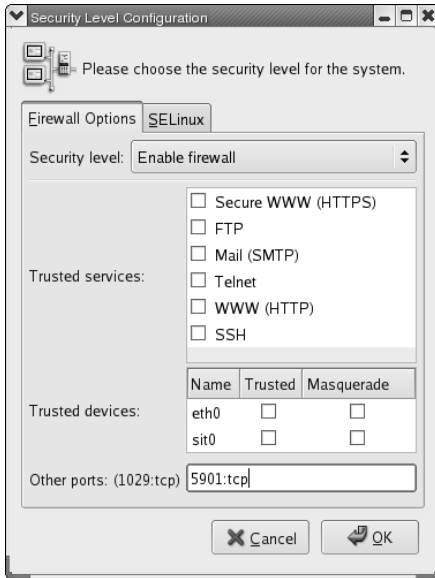


Figure 16-3 Opening TCP port 5901 with the Security Level Configuration tool.

Customizing the VNC Server

Sure, the server configuration is complete, but the default desktop (jump ahead to Figure 16-5) is ugly, unless you like the twm window manager and the plain gray background. Remember that `xstartup` file, `/home/bubba/.vnc/xstartup`? You can edit that to change the default desktop. Listing 16-1 shows the default `xstartup` file:

```
#!/bin/sh

# Uncomment the following two lines for normal desktop:
# unset SESSION_MANAGER
# exec /etc/X11/xinit/xinitrc

[ -x /etc/vnc/xstartup ] && exec /etc/vnc/xstartup
[ -r $HOME/.Xresources ] && xrb $HOME/.Xresources
xsetroot -solid grey
vncconfig -iconic &
xterm -geometry 80x24+10+10 -ls -title "$VNCDESKTOP Desktop" &
twm &
```

Listing 16-1 The default `xstartup` file.

As you learned in Chapter 9, X Window System startup files configure your X environment and start certain X programs each time you log in. In this case, `xstartup` does the following:

1. Executes the file `/etc/vnc/xstartup` if it exists and is executable.
2. Loads any X resources stored in the file `.Xresources` if it exists and is readable.
3. Invokes `xsetroot` to set the background color to solid gray.
4. Starts the `vnconfig` program minimized.
5. Starts an 80x24 `xterm` with a specific title.
6. Starts `twm`, the Tab Window Manager.

If you want your usual desktop, the one you get when you are sitting in front of the system, do what the instructions suggest and uncomment the following two lines:

```
# unset SESSION_MANAGER
# exec /etc/X11/xinit/xinitrc
```

You should also comment out the last four lines. The modified file should resemble Listing 16-2.

```
#!/bin/sh

# Uncomment the following two lines for normal desktop:
unset SESSION_MANAGER
exec /etc/X11/xinit/xinitrc

[ -x /etc/vnc/xstartup ] && exec /etc/vnc/xstartup
[ -r $HOME/.Xresources ] && xrdp $HOME/.Xresources
#xsetroot -solid grey
#vnconfig -iconic &
#xterm -geometry 80x24+10+10 -ls -title "$VNCDESKTOP Desktop" &
#twm &
```

Listing 16-2 `xstartup` modified to display your standard desktop.

The modified `xstartup` file starts your usual desktop because it invokes the system `xinitrc` file, which starts the standard X initialization process.

You might want to stick with the default VNC server configuration, though. Running X over a network connection is always going to be slower than running it directly unless you have a *very* fat pipe and no network congestion. The default VNC configuration runs the `twm` window manager, which is considerably faster than X desktop environments like KDE and GNOME. `twm` is also

faster than many modern window managers too. The point is that the less eye candy transmitted over the wire, the faster your VNC setup will be. Yes, twm might make your eyes bleed, and it definitely lacks features that many have come to expect from your window manager. However, if you are working remotely, you might not need the application integration and the eye candy as much as you need a responsive “desktop.” You can also customize twm by editing (or creating) a `twmrc` file, which might make it a little easier on your eyes.

Testing the VNC

Testing your VNC setup is simple. First, make sure that it works locally by starting a VNC client on the same system as the VNC server. If the VNC server you started in the section “Setting Up a VNC Server” isn’t running, restart it:

```
$ vncserver
```

```
New 'luther.kurtwerks.com:1 (kwall)' desktop is luther.kurtwerks.com:1
```

```
Starting applications specified in /home/kwall/.vnc/xstartup
```

```
Log file is /home/kwall/.vnc/luther.kurtwerks.com:1.log
```

Next, in a terminal window, start the VNC client, or viewer, by executing the command `vncviewer :n`, replacing `n` with the display number `vncserver` reported when it started (1, in the example). You will see the password prompt shown in Figure 16-4.

```
$ vncviewer :1
```

```
VNC Viewer Free Edition 4.1.1 for X - built Apr 27 2005 02:25:46
```

```
Copyright (C) 2002-2005 RealVNC Ltd.
```

```
See http://www.realvnc.com for information on VNC.
```

```
Sat May 21 01:01:32 2005
```

```
CConn:      connected to host localhost port 5901
```

```
CConnection: Server supports RFB protocol version 3.8
```

```
CConnection: Using RFB protocol version 3.8
```

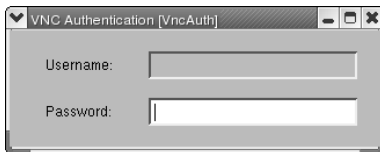


Figure 16-4 The VNC authentication dialog box.

Type the password you provided when you configured the VNC server and press Enter. Assuming that you are using the default VNC server configuration (the one that uses `twm`), the resulting screen should resemble Figure 16-5.

Figure 16-5 doesn't look terribly impressive, but the purpose of this exercise is to satisfy yourself that the server is working. You should be able to start applications, surf the Web (assuming that the server is connected to the Internet), and use the remote computer just as if you were sitting in front of it. Figure 16-6, for example, shows the Fedora Project Web site in a Mozilla browser session started from the VNC client.

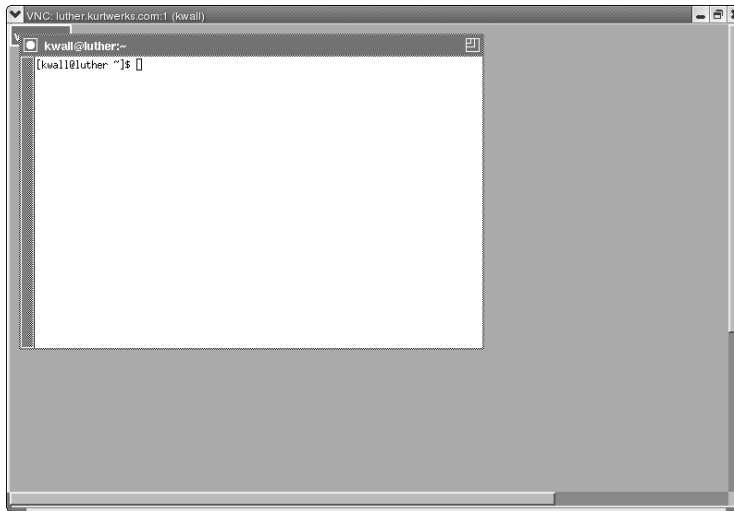


Figure 16-5 Viewing the default VNC desktop in a VNC client.

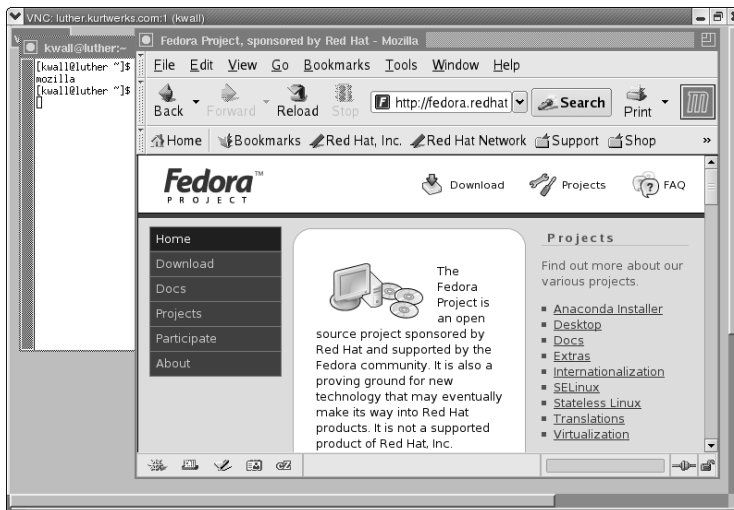


Figure 16-6 Using Mozilla in the VNC client.

You can also start the VNC client by selecting Red Hat ➤ Accessories ➤ VNC Viewer. If you use the menu, you will have to specify the VNC server to which to connect, as shown in Figure 16-7.

The server specification shown in Figure 16-7 included the display number. If you don't include this value, you won't get a connection. You can use either the IP address, as shown in the figure, or the hostname. In fact, if you use the hostname, you can use the FQDN or the alias, if one is defined in `/etc/hosts`. After establishing that the server is working, close the client session by pressing F8 while the viewer has the focus and selecting Exit Viewer from the pop-up menu. (See Figure 16-8.)

Now, test the configuration from a remote machine. Figure 16-9 shows a VNC client session running on Microsoft Windows XP. The client software is the RealVNC for Windows, which is the same product as used on Fedora Core and RHEL systems.

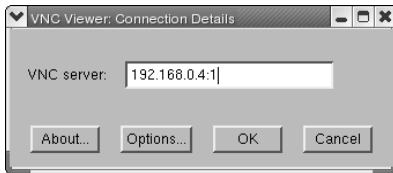


Figure 16-7 Specifying the target VNC server.

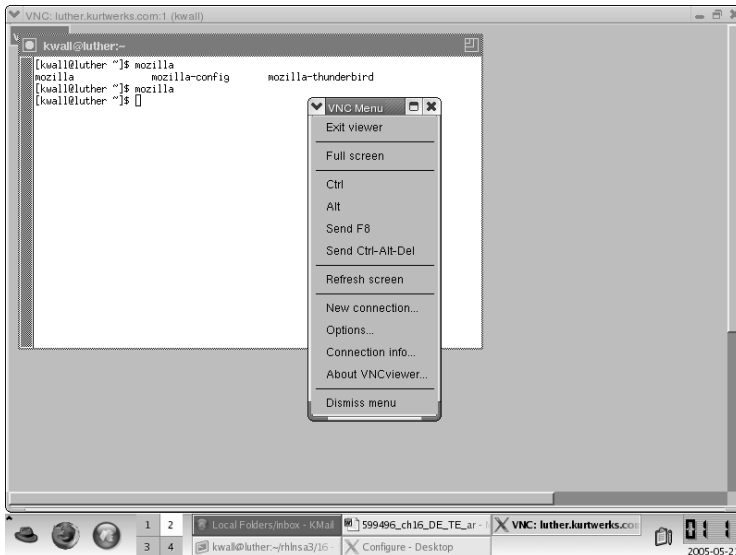


Figure 16-8 The VNC viewer pop-up menu.

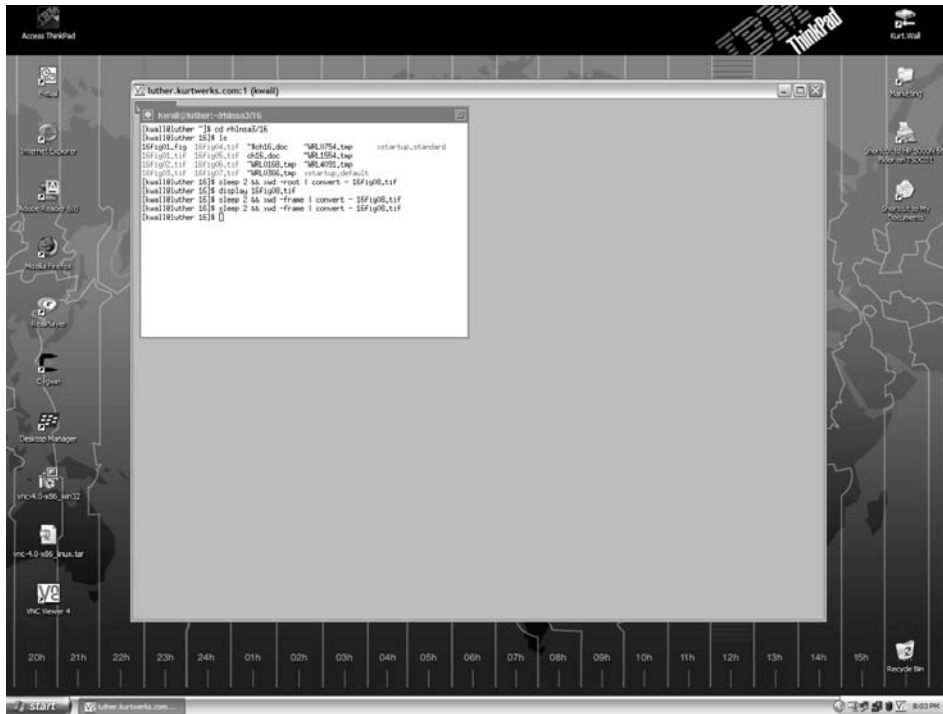


Figure 16-9 Using a Linux VNC server from Microsoft Windows.

To get the full effect of VNC, especially when you use the viewer from a Windows system, consider Figure 16-10, which shows a VNC session to the server configured in the first section of this chapter. In this case, the server was started using the modified `xstartup` in Listing 16-2, which displays the standard system desktop (KDE, in this case) rather than the faster, nimbler, and uglier `twm`.

To reiterate the point made earlier, running X across a network, especially the Internet, is going to be a slower affair than running it directly. On a LAN such as a home network (the environment in which the Figures 16-9 and 16-10 were taken), the performance hit will be minimal, and you might not notice or mind the difference. Across the Internet, you will find it intolerably slow running something like KDE, GNOME, or the more, shall we say, feature-rich window managers. Whether you use KDE with all the trimmings or something leaner like `twm`, you will *definitely* appreciate being able to access your desktop system from a remote location.



Figure 16-10 Running KDE on Microsoft Windows via VNC.

When you are done running the VNC server, you can kill the running process by passing the `-kill` option to `vncserver`:

```
$ vncserver -kill :1
Killing Xvnc process ID 30790
```

Replace :1 with the display number on which the VNC server was running.

Summary

Providing road warriors and telecommuters access to LAN-based services is an increasingly important requirement that system administrators are asked to meet. VNC is one way to meet that requirement. With minimal configuration on the server side and no configuration on the client side, you can provide remote access to one or more Linux systems using VNC. As always, poking a hole in your firewall to permit external access poses a security risk, so you have to weigh that risk against the advantages and decide whether VNC is the way you want to go. As you saw in this chapter, the advantages of remote access to your desktop system are hard to dispute.

Providing Additional Network Services

IN THIS CHAPTER

- Configuring a Time Server
- Providing a Caching Proxy Server

Any system administrator can testify that requests for new or enhanced services pop up constantly. Sometimes they arrive in a trickle; other times a fire hose delivers them. One commonly requested service is a time server, a service that provides the authoritative clock against which all other systems on the LAN sync their clocks. Perhaps the boss wants people to be able to share information on the company intranet and asks you to create a way to post Web documents to the intranet Web server. This chapter describes setting up two nonessential LAN-based services that select groups of intranet users (or you) might find useful. These services fall into the nonessential category because they provide functionality that is nice to have but without your LAN is still amply serviceable.

The nonessential services described in this chapter include an NTP-based time server and a caching proxy server. Naturally, these two topics hardly exhaust the possible list of conveniences users (and managers) can and do request. Some of the topics that might seem appropriate for this chapter are covered elsewhere in the book. For example, Chapter 16 describes how to set up a VNC server to give remote users access to their desktops. Building a Web server for your company intranet is no different from building a Web server that faces the Internet, a topic covered in Chapter 23. Likewise, Chapter 24 shows you how to configure some of the Web-based services users typically want, such as mailing lists, Web-based email, site search functionality, and RSS feeds.

More to the point, we had to limit the topics we covered, just as administrators have to limit the services they provide. And that point might be as important as the specifics of creating a time server or setting up a proxy server. You have to know when to say “No” to feature requests. In some cases, a requested feature might pose more of a security risk than you are willing to take. In other cases, a network enhancement might stress your network infrastructure or utilize network, system, and CPU resources better used for other purposes. In still other cases, you might simply lack the personnel or time to manage any more services. Whatever the case, you must be able and willing to say no to such requests, if only to preserve your sanity.

Configuring a Time Server

For this chapter’s purposes, a *time server* is a daemon that runs on one machine and to which other systems (in this case, clients on your LAN) synchronize their system clocks. In the common case, you synchronize your time server’s clock against one or more reference time servers that are situated outside your LAN. In some cases, the time server synchronizes its time against a specially designed clock. This hardware clock is a separate, single-purpose hardware device that maintains extremely accurate time. (More about these details shortly.)

The motivation for a time server is to keep the system time consistent throughout the LAN so that time-sensitive operations work reliably. In development shops, the `make` utility uses file timestamps to determine which components of a software project need to be rebuilt. If you have an inaccurate system clock, the timestamps on files will be inconsistent, causing project builds to fail or to behave unpredictably. Similarly, source code version control systems often rely on file and directory timestamps to track changes to files maintained in the source code repository. NFS is also sensitive to timekeeping irregularities. If the system clock on a client machine is significantly faster or slower than the system clock on the NFS server, you will run into problems saving files. Irregularities in system clocks between client systems and the server can adversely affect the mechanisms content management systems use to update and maintain complex Web sites.

To take one example from our own experience, we have an NFS server from which users mount home directories and project-specific work directories. Everything was working smoothly until we started seeing errors during project builds. The errors initially pointed to a problem with the server’s system clock. After we reset the system clock and synced it to an external time server, most users reported no further problems and project builds worked normally again, but some users continued to experience timestamp-related write errors on files mounted from the server. It took a couple of days for us to work out

that the only users continuing to have these problems were users whose systems had flaky system clocks that lost anywhere from 15 seconds to several minutes each day. After configuring these client systems to coordinate their system clocks with the NFS server's clock, the problem went away and has not returned. Eventually, we set up a time server for the entire LAN and configured all systems to synchronize to that time server.

Selecting a Time Server Solution

If you've decided (or have been told) that you need a time server, your options fall into three categories: hardware, software, or both. As mentioned a moment ago, the hardware solution involves installing a high-resolution clock in your facility and then configuring client systems to synchronize their system clocks against that dedicated device. If you have a bottomless pit of money for a budget, you can install an atomic clock, such as a cesium fountain clock, to keep ridiculously accurate time. In the great majority of cases, though, hardware clocks aren't quite so high end. Instead, these lower-end clocks are equipped with a radio or satellite receiver that monitors one of several radio or satellite broadcast frequencies specifically allocated to carrying a time signal. These time signals are broadcast by the United States Naval Observatory (USNO), the National Institute of Standards and Technology (NIST), or another official organization of the United States government. (Most countries provide similar services.) You can buy high-quality hardware clocks for \$1000. In fact, depending on your needs, you can get good, high-resolution, dedicated hardware clocks for a few hundred dollars.

NOTE For the record, USNO and NIST have multiple atomic clocks, so they keep ridiculously accurate time on your behalf and do so with your tax dollars. To learn more about the NIST timekeeping service, visit the NIST's Time & Frequency Division Web site at www.boulder.nist.gov/timefreq/service/its.htm. The USNO operates a comparable site at <http://tycho.usno.navy.mil/>.

The most common time server solution, especially for small(ish) networks and organizations, is strictly software-based. The simplest approach is to use the date program to set your system clock to the time broadcast by another system. The kid-tested, mom-approved, and syadmin-preferred method, though, is to use the Network Time Protocol, or NTP. NTP is an open standard that defines how Internet time servers work and how clients can communicate with these time servers to maintain accurate time. How accurate? If you believe the NTP documentation, the best precision is about 232 picoseconds, or 2.32×10^{-10} seconds, or 0.00000000232 seconds. If you need greater precision than 232 picoseconds, get that cesium clock.

NOTE David Mills wrote the NTP implementation universally used on Linux systems. Mills has done an excellent job of documenting his work. For an overview of time synchronization issues in general, see “Executive Summary: Computer Network Time Synchronization” at Mills’ Web site, eecis.udel.edu/~mills/exec.html. For more information about NTP, the authoritative Web site is ntp.org.

NTP consists of a daemon (`ntpd`), a small set of utility programs (`ntpq`, `ntpdcc`, `ntpdate`, `ntptrace`, `tickadj`, `ntpstime`, `ntptime`, `ntp-kegen`, and `ntpsim`), and the all-important configuration file, `/etc/ntp.conf`. The NTP daemon is dual-purpose. It acts as a server, listening for time synchronization requests and providing the time in response, and as a client, communicating with other time servers to get the correct time and adjust the local system accordingly. Table 17-1 briefly describes the utility programs.

Configuring the Time Server

Setting up your time server requires a small amount of preparation, implementation, and verification. You’ll need to perform the following tasks:

1. Install the NTP software.
2. Locate suitable time servers to serve as reference clocks.
3. Configure your local time server.
4. Start the NTP daemon on the local time server.
5. Make sure that the NTP daemon responds to requests.

Table 17-1 NTP Utility Programs

PROGRAM	DESCRIPTION
<code>ntpdate</code>	Sets the system date and time via NTP
<code>ntpdcc</code>	Controls the NTP daemon, <code>ntpd</code>
<code>ntp-keygen</code>	Generates public and private keys for use with NTP
<code>ntpq</code>	Queries the NTP daemon
<code>ntpsim</code>	Provides NTP simulation for development and testing
<code>ntptime</code>	Displays the time variables maintained by the Linux kernel
<code>ntptrace</code>	Traces a chain of NTP servers back to the primary source
<code>tickadj</code>	Sets certain time variables maintained by the Linux kernel

Installing the NTP software is simple. Use the `rpmquery` command to make sure that the `ntp` package is installed:

```
$ rpmquery ntp
ntp-4.2.0.a.20040617-4
```

The version number you see might be slightly different. If the `ntp` package isn't installed, install it using the installation tool of your choice before proceeding.

Selecting Reference Clocks

For your time server to keep and thus to serve accurate time, your local time server needs to synchronize its time against one or more master or reference clocks. NTP is a distributed application, meaning that servers and clients are dispersed, that any given client can request a time check from any given server (modulo access restrictions), and that the application continues to function in spite of the failure or inaccessibility of one or even many of the servers. NTP is also hierarchical and organizes time servers into several strata to reduce the load on any given server or set of servers. Stratum 1 servers are referred to as *primary servers*, stratum 2 servers as *secondary servers*, and so on. There are far more secondary servers than primary servers. Secondary servers sync to primary servers, and clients sync to secondary or tertiary servers.

NTP also provides for syncing to *pool servers*, a large class of publicly accessible secondary servers maintained as a public service for use by the Internet-connected computing community at large. The NTP pool time servers, organized into the subnet `pool.ntp.org`, use DNS round robin to assign randomly selected open access time servers to NTP clients (recall that an NTP server can also be an NTP client). The rationale for random server selection is to distribute the client load more or less equally across all servers participating in the pool *and* to ensure that clients are syncing to an appropriately distributed set of time servers.

The upshot for you is simple:

- Use one or more secondary servers as your local server's reference clock.
- Use the pool servers if possible.

Accordingly, this section configures an NTP server to use the pool servers and also shows you how to configure your local time server to use explicitly selected secondary servers.

To use the pool servers, NTP is ready to run after you install the `ntp` package. Listing 17-1 shows `ntpd`'s configuration file, `/etc/ntp.conf`, stripped of most comments and white space.

```

restrict default nomodify notrap noquery
restrict 127.0.0.1

# --- OUR TIMESERVERS -----
server pool.ntp.org
server pool.ntp.org
server pool.ntp.org

# --- GENERAL CONFIGURATION ---
server 127.127.1.0                # local clock
fudge 127.127.1.0 stratum 10
driftfile /var/lib/ntp/drift
broadcastdelay 0.008
keys /etc/ntp/keys

```

Listing 17-1 The default NTP configuration file.

The first two entries, beginning with the `restrict` directive, are, not surprisingly, restrictions on the listed IP addresses or hostnames. The first entry uses the keyword `default`, which means an IP address and mask of 0.0.0.0. The option flags, `nomodify`, `notrap`, and `noquery`, prevent the listed IP address from modifying, logging, or querying the NTP service on the server. The second rule, `restrict 127.0.0.1`, permits all NTP activity over the loopback interface. All activity is permitted because there are no option flags specified. To deny all activity, you would use the `ignore` flag, but you shouldn't do this on the loopback interface because doing so would prevent certain NTP administrative functions (issued using the `ntpd` command) from working properly.

The next three entries, beginning with the `server` directive, identify the time servers you want to use as reference clocks. In this case, `ntpd` is being configured to use the pool servers. Notice that the names are all `pool.ntp.org`. Even though the names are the same, the NTP server pool is configured to use DNS round robin, so three hostname lookups on the same name will return three different IP addresses. You can try this yourself to verify that round robin is working. Issue the command `host pool.ntp.org` at the command prompt and, unless your DNS client is broken, you should see output resembling the following:

```

$ host pool.ntp.org
pool.ntp.org has address 213.219.244.16
pool.ntp.org has address 216.27.185.42
pool.ntp.org has address 62.220.226.2
pool.ntp.org has address 69.37.143.241
pool.ntp.org has address 81.169.154.44
pool.ntp.org has address 82.219.3.1
pool.ntp.org has address 139.140.181.132
pool.ntp.org has address 146.186.218.60

```

```
pool.ntp.org has address 195.18.140.242
pool.ntp.org has address 203.217.30.156
pool.ntp.org has address 209.126.142.251
pool.ntp.org has address 212.23.29.225
pool.ntp.org has address 212.41.248.75
pool.ntp.org has address 212.254.25.164
pool.ntp.org has address 213.10.208.72
```

Normally, a hostname resolves to one and only one IP address, but when DNS round robin behavior is enabled, a single hostname can resolve to multiple IP addresses, the purpose being to equalize the load on any single system.

The general configuration section sets broad operational policies that control ntpd's overall behavior. The line `server 127.127.1.0` instructs the NTP daemon to use the local clock (referred to as an *undisciplined local clock*) if no external reference clocks are accessible. You can use any address in the range 127.127.1.0 to 127.127.1.255, although the convention is to use 127.127.1.0. The line `fudge 127.127.1.0 stratum 10` limits the use of the local lock by assigning it a very low place in the time server hierarchy, the intent being to prevent the local clock from interfering with other, likely more accurate time sources elsewhere on network and to enable (or, perhaps, compel) ntpd to look pretty hard for other time sources before using the undisciplined local clock. In its normal operation, ntpd listens for broadcasts from other time servers when trying to find a reference clock. If it finds a time server declaring itself at a higher stratum than 10, ntpd will use the higher-stratum clock instead of the undisciplined local clock.

The directive `driftfile /var/lib/ntp/drift` specifies the name of the file that stores the oscillation frequency of the local clock. NTP uses this frequency, which varies slightly over time, to make appropriate adjustments to the system time. The `broadcastdelay` directive sets the number of seconds (0.008 in this case) used to calculate the network latency or delay between the local server and a remote reference server. On a LAN, values between 0.003 and 0.007 seconds are suitable, but when two servers must communicate across the Internet, it is often necessary to use a longer delay value.

The last line, `keys /etc/ntp/keys`, tells NTP where to find the cryptographic keys used to encrypt exchanges between client and server machines. The purpose for encrypting the data exchange is to prevent an unauthorized reference server accidentally or deliberately sending time signals to your local time server. Another reason to use encryption is when you enable remote NTP administration and want to make sure that only properly authorized and authenticated systems can perform remote administration.

NTP version 4 (NTPv4) supports asymmetric encryption, more commonly known as *public key encryption*, using a method or protocol referred to as *autokey* but the default configuration on Fedora Core and RHEL systems as installed doesn't use it without some configuration work. If you wish to use

server-side encryption, the following procedure creates a basic autokey setup. You should use the following procedure on the system that will be configured as an NTP server. The next section, “Configuring an NTP Client,” describes configuring client authentication. Although advisable, it isn’t strictly necessary to set up encryption. Of course, wearing seat belts while driving is advisable, but isn’t strictly necessary.

1. Add the following lines to `/etc/ntp.conf`:

```
broadcast 224.0.1.1 autokey
crypto pw serverpassword
keysdir /etc/ntp
```

Replace `serverpassword` with a password of your choosing.

2. Generate the key files and certificates using the following commands:

```
# cd /etc/ntp
# ntp-keygen -T -I -p serverpassword
Using OpenSSL version 90701f
Random seed file /root/.rnd 1024 bytes
Generating IFF parameters (512 bits)...
IFF 0 60 81      1 49 111      2 1 2      3 1 2
Generating IFF keys (512 bits)...
Confirm  $g^{(q-b)} g^b = 1 \pmod p$ : yes
Confirm  $g^k = g^{(k+b)r} g^{(q-b)r}$ : yes
Generating new iff file and link
ntpkey_iff_ntpbeast.example.com- \
>ntpkey_IFFpar_ntpbeast.example.com.3318548048
Generating RSA keys (512 bits)...
RSA 0 24 112      1 11 135      3 1 4
Generating new host file and link
ntpkey_host_ntpbeast.example.com- \
>ntpkey_RSAkey_ntpbeast.example.com.3318548048
Using host key as sign key
Generating certificate RSA-MD5
X509v3 Basic Constraints: critical,CA:TRUE
X509v3 Key Usage: digitalSignature,keyCertSign
X509v3 Extended Key Usage: trustRoot
Generating new cert file and link
ntpkey_cert_ntpbeast.example.com->ntpkey_RSA- \
MD5cert_ntpbeast.example.com.3318548048
```

The output wraps (indicated by `\` in the listing) because of page layout constraints.

3. If `ntpd` is running, restart it:

```
# service ntpd restart
Shutting down ntpd: [ OK ]
Starting ntpd: [ OK ]
```

If `ntpd` is *not* running, start it:

```
# service ntpd start
```

```
Starting ntpd:
```

```
[ OK ]
```

4. Use the following `chkconfig` commands to make sure that `ntpd` starts in at boot time and in all multiuser run levels:

```
# chkconfig --level 0123465 ntpd off
```

```
# chkconfig --level 345 ntpd on
```

At this point, NTP's autokey encryption is enabled, and you've also made sure that NTP services will start and stop appropriately at boot time.

Configuring an NTP Client

Configuring an NTP client requires fewer steps than configuring a server does. You select the server to use as a reference clock, start the NTP daemon, `ntpd`, and you're done. The GUI-addicted can use the Date/Time Properties tool. Either start it from the menu (Red Hat ⇨ System Settings ⇨ Date & Time) or type `system-config-date` at a command prompt. Either way, you should see the screen shown in Figure 17-1.

If NTP is already running on your system, the Date & Time tab will be disabled (grayed out). Click the Network Time Protocol tab to configure NTP. To enable NTP, place a check mark in the Enable Network Time Protocol check box. Doing so enables the NTP Servers pane, as shown in Figure 17-2.

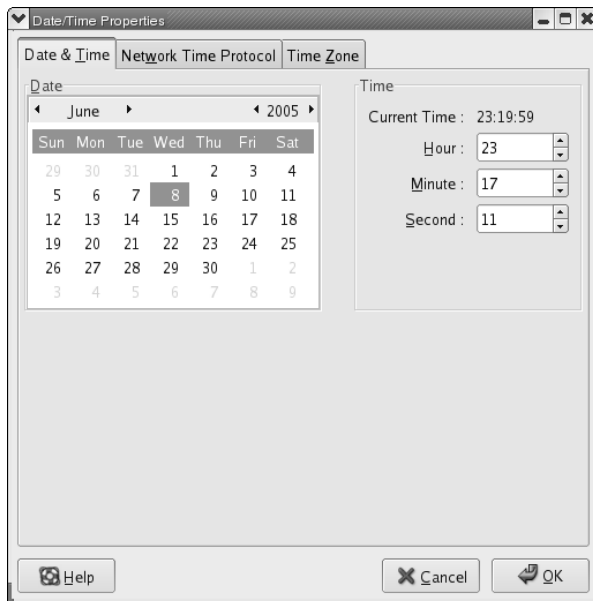


Figure 17-1 The Date/Time Properties tool.

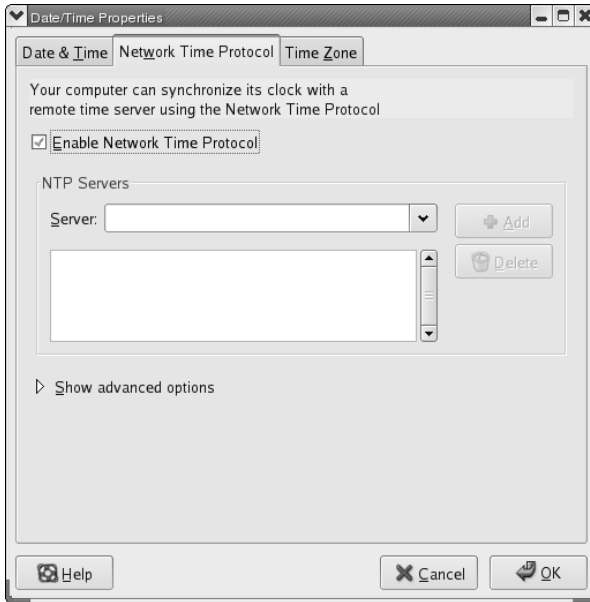


Figure 17-2 The Network Time Protocol tab.

Fedora Core and RHEL ship with two servers configured: `clock.redhat.com` and `clock2.redhat.com`. To use one of these servers, select it from the Server drop-down box and click the Add button. If you want to add a different server, type the name or IP address of the server you want to use and click the Add button to add it to the list. Figure 17-3 shows the added servers `pool.ntp.org` and `ntpbeast.example.com`. To delete a server, highlight in the list and click the Delete button.

To configure advanced NTP options, click the Show advanced options arrow. These options allow you to use the system clock (the undisciplined local clock described in the previous section) as a reference clock (enabled by default) and to use NTP broadcast (disabled by default). *NTP broadcast* causes the NTP daemon to listen for remote servers rather than configuring clients to use a specific server.

After you have made your configuration changes, click the OK button to close the Date/Time Properties tool. If you see a dialog box indicating that the tool is connecting to an NTP server, just be patient; this dialog box is strictly informational and disappears quickly.

CAUTION If you use the Date/Time Properties tool to configure your NTP client, any hand edits you might have made to `/etc/ntp.conf` will be lost. The lesson here is either not to use the graphical tool or to make a backup copy of `/etc/ntp.conf` *before* using the GUI tool.

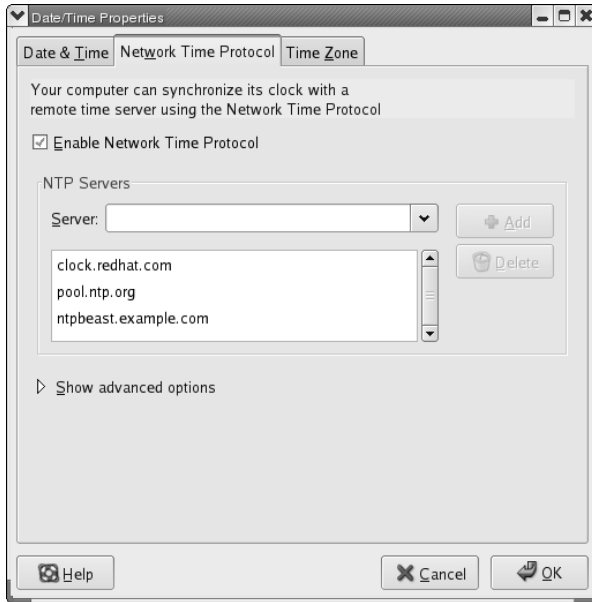


Figure 17-3 Adding NTP servers.

One reason you might not want to use the Date/Time Properties tool is that you cannot perform advanced customization of NTP. Moreover, if you want to use the pool servers and provide the name `pool.ntp.org`, the tool replaces that name with the first IP address that resolves to the hostname. The problem is that the Date/Time Properties tool does not take into account that `pool.ntp.org` uses DNS round robin. As a result, if you choose to use the NTP server pool, you will have to edit `/etc/ntp.conf` by hand. It might be simpler and less aggravating to edit the configuration file directly. The additional configuration steps described next require you to edit `/etc/ntp.conf` directly.

If you configured your NTP server to use autokey encryption, you will also need to configure any NTP clients you have to use autokey encryption. The following procedure walks you through doing so.

1. Add the following lines to `/etc/ntp.conf` on the client:

```
crypto pw clientpassword
keysdir /etc/ntp
server timeserver autokey
```

Replace `clientpassword` with the password you want to use on the client. Replace `timeserver` with the name or IP address of the system you configured as the server in the previous section.

2. Generate the client keys and certificate:

```
# cd /etc/ntp
# ntp-keygen -H -p clientpassword
```

We don't show the output of this command to conserve space.

3. Import the key created in the section on server configuration using some sort of encrypted mechanism, such as `sftp` or `scp`. The following example uses `scp`:

```
# cd /etc/ntp
# scp user@timeserver:/etc/ntp/ ntpkey_IFFkey_timeserver.3318548048 .
# ln -s ntpkey_IFFkey_timeserver.3318548048 ntpkey_iffkey_timeserver
```

Replace `user` with the name of a user that has `scp` privileges on the machine you are using as the time server. Under normal circumstances, this can be any user with a login account on the time server.

4. If `ntpd` is running, restart it:

```
# service ntpd restart
Shutting down ntpd: [ OK ]
Starting ntpd: [ OK ]
```

If `ntpd` is *not* running, start it:

```
# service ntpd start
Starting ntpd: [ OK ]
```

5. Execute the command `ntpq -p` to make sure that the NTP client can communicate with the designed server. The output should resemble the following:

```
# ntpq -p ntpbeast.example.com
      remote           refid      st t when poll reach  delay  offset  jitter
=====
ntpbeast.example 192.168.0.1      1 u  209 1024  377   37.200    7.199   0.226
```

Replace `ntpbeast.example.com` in the command with the name (or IP address) of the time server you are using.

As a final test, you can use the `ntpstat` command to query the time server from the host machine to ensure that you can retrieve the time. The output will show the server to which `ntpd` synchronizes, the clock's precision, and the interval on which `ntpd` resynchronizes:

```
# ntpstat
synchronised to NTP server (192.168.0.1) at stratum 3
time correct to within 70 ms
polling server every 128 s
```

After `ntpd` has been running for a while, you can `grep` the system log for NTP-related log entries to see `ntpd`'s activity. The following listing briefly illustrates what you might see:

```
# grep ntpd /var/log/messages
Feb 28 07:56:56 luther ntpd[3549]: ntpd 4.2.0a@1.1190-r Mon Oct 11 09:10:20
EDT2004 (1)
Feb 28 07:56:56 luther ntpd: ntpd startup succeeded
Feb 28 07:56:56 luther ntpd[3549]: precision = 1.000 usec
Feb 28 07:56:56 luther ntpd[3549]: Listening on interface wildcard, 0.0.0.0#123
Feb 28 07:56:56 luther ntpd[3549]: Listening on interface wildcard, ::#123
Feb 28 07:56:56 luther ntpd[3549]: Listening on interface lo, 127.0.0.1#123
Feb 28 07:56:56 luther ntpd[3549]: Listening on interface eth0, 192.168.0.4#123
Feb 28 07:56:56 luther ntpd[3549]: kernel time sync status 0040
Feb 28 07:57:16 luther ntpd[3549]: frequency initialized -67.149 PPM from \
/var/lib/ntp/drift
Feb 28 07:58:23 luther ntpd[3549]: synchronized to 192.168.0.1, stratum 2
Feb 28 07:58:22 luther ntpd[3549]: time reset -0.461319 s
Feb 28 07:58:22 luther ntpd[3549]: kernel time sync disabled 0041
Feb 28 08:01:35 luther ntpd[3549]: synchronized to 192.168.0.1, stratum 2
Feb 28 08:09:04 luther ntpd[3549]: kernel time sync enabled 0001
```

One line of output wraps due to page layout constraints. In this log snippet, you can see `ntpd`'s startup, a report of the interfaces on which it is listening, and, in the final six lines, the synchronization activity. For example, the line that reads `frequency initialized -67.149 PPM from /var/lib/ntp/drift` indicates that `ntpd` read the drift file to set the clock frequency. The following lines shows `ntpd` synchronizing to the clock at 192.168.0.1, subtracting 0.461319 seconds from the system clock, and then reconfirming the synchronization about three minutes later. This type of activity will continue throughout the day.

Playing Nicely and Wisely with NTP

Playing nicely with NTP means that you should adhere to the recommended guidelines for selecting an external time server as a reference clock. The rule of thumb is that if your network has less than 100 systems, you should use a stratum 2 server rather than a stratum 1 server. This guideline exists to limit the ever-increasing load on the stratum 1 servers. Rather than syncing 100 client systems to an external reference server, the preferred practice is to set up a time server inside your firewall (you *do* have firewall, right?), synchronize it with an external reference clock, and then synchronize your LAN clients with your internal time server. If you have a large network, that is, one with more than 100 NTP clients, it might be prudent to invest in a dedicated hardware clock and use that as your time server. Such a measure would reduce your reliance on an external resource *and* limit the load placed on public time servers. If you

wanted to be a really good netizen, you could even make your hardware time server part of the NTP public access server pool — see the NTP Web site at <http://www.ntp.org> for all the particulars.

Playing wisely with NTP means choosing at least three external reference clocks (we usually select four) and making sure that the reference clocks are dispersed geographically and in terms of network connections. The geographic dispersal is easy enough to achieve by selecting one from each compass point that you don't occupy. For example, if you are located in Pittsburgh, Pennsylvania, which is situated in the northeastern United States, choose reference clocks from the southeast, southwest, and northwest regions of the United States.

Network dispersal is somewhat more difficult to achieve because you want to try to locate reference clocks that use different Internet backbones or backbone providers. The theory is that if one backbone or a segment of one provider's backbone fails (because the fiber-optic cable is cut by a backhoe, say), access to that time server will be impaired. NTP is engineered to select an alternative server, but if all your servers use that same severed backbone, you're hosed. So, for example, if your primary network access uses UUNet, good geographic dispersal requires finding reference clocks that use, say, Sprint, MCI, or AT&T. As a practical matter, you can use the `tracert` command to examine the network paths packets follow to reach a given time server and then compare those paths, confirming that there are as few routing points in common between each time server as possible. It is a tedious undertaking, but if your network and the services it provides rely on accurate, consistent time services, it is well worth the effort.

Providing a Caching Proxy Server

What is a caching proxy server and why should you use one? A *caching proxy server* is software (and potentially hardware) that stores (*caches*) frequently requested Internet objects such as Web pages, Java scripts, and downloaded files closer (in network terms) to the clients that request those objects. When a new request is made for a cached object, the proxy server provides the object from its cache instead of allowing the request to go to the source. That is, the local cache serves the requested object as a proxy or substitute for the actual server. The motivation for using a caching proxy server is two-fold: to provide accelerated Web browsing by reducing access time for frequently requested objects and to reduce bandwidth consumption by caching popular data locally, that is, on a server that exists between the requesting client and the Internet. The HTTP acceleration feature speeds up Web browsing because cached pages need not be re-retrieved unless the original page has been updated since it was last cached.

NOTE Caching proxy servers are sometimes used to create content filters to block attempts to access certain sites or content. Quite aside from the Orwellian aspect of content filters, using a proxy server as a content filter misuses the proxy server. Limiting access to certain sites is easier to accomplish by blocking packets destined for banned sites. Moreover, because proxy servers utilize system resources, especially RAM, for caching data, the resources allocated to content filters are better applied to caching and retrieving data.

Nevertheless using a proxy as a single point through which to apply and simplify site-blocking is quite common. Such site-blocking proxy servers squander resources only on the system on which they're running, which can be an efficient approach if your site uses a single HTTP proxy for hundreds of desktops.

The Web proxy discussed in this chapter is called Squid. There is no intrinsic connection between the name Squid and the function of the proxy server. It's just a name that stuck. The Squid FAQ explains the name this way: "Harris' Lament says, 'All the good [names] are taken.' We needed to distinguish this new version from the Harvest cache software. Squid was the code name for initial development, and it stuck (Squid FAQ, squid-cache.org/Doc/FAQ/FAQ-1.html#ss1.3)."

Squid provides the basic caching and proxy functions just described. It also caches DNS lookups, to speed up subsequent DNS queries, performs nonblocking DNS queries, and implements *negative caching*, which means that Squid remembers when a request was made for an object (or for a DNS resource) that doesn't exist and doesn't try to retrieve it or find the nonexistent object in its cache. In addition to these proxy and caching services, Squid also has full SSL support, includes extensive access control, and features a rich logging system that you can use to fine-tune the caching behavior.

Finally, Squid can work as a transparent proxy. An "ordinary" (opaque?) proxy requires Web clients to specify the hostname and port of the proxy, which then forwards requests to the requested server. This has obvious disadvantages, the biggest one being that you have to configure all the Web clients to use the proxy. With transparent proxying, Web clients think they are communicating with the requested server when in fact they are communicating with the proxy. A transparent proxy still intercepts requests, but Web clients neither notice nor care.

Squid is included in both Fedora Core and RHEL, but depending on the type of installation you performed, it might not be installed. The following `rpmquery` command will show you if Squid is installed (you only need the Squid package):

```
$ rpmquery squid
squid-2.5.STABLE8-1.FC3.1
```


The version number might be slightly different by the time you read this text. If Squid is *not* installed, you'll obviously need to install it before proceeding. The configuration process includes the following steps:

1. Verifying the kernel configuration
2. Configuring Squid
3. Modifying the Netfilter configuration
4. Starting Squid
5. Testing the configuration

The following subsections provide the details for each of these steps.

Verifying the Kernel Configuration

This section is titled “Verifying the Kernel Configuration” because the kernel features you need, such as IP forwarding and Netfilter (iptables) support, are already in place on Fedora Core and RHEL systems. The most you should need to do is load a kernel module or enable IP forwarding. Nonetheless, knowing the kernel modifications that have to be made is useful if you decide at some point to discard the stock Fedora or RHEL kernel and roll your own.

The most important kernel feature you need is Netfilter support because Netfilter will handle the actual proxying of browser requests. Specifically, you need to enable Netfilter and the modules that support:

- Connection tracking
- IP tables
- Full NAT (Network Address Translation)
- Support for the REDIRECT target

For completeness' sake, you need support for the `/proc` file system and `sysctl` support for manipulating tunable runtime kernel options. These two options are pretty much *de jure* these days and they are definitely present in the stock Fedora and RHEL kernels.

The first thing to do is enable IP forwarding on the system that will run Squid. IP forwarding enables the kernel to send, or *forward*, packets that arrive on one network interface to another, an essential capability for proxying. The following `sysctl` command will show you if IP forwarding is enabled:

```
# sysctl -n net.ipv4.ip_forward
1
```

This command queries the kernel to see whether IP forwarding (actually IPv4 forwarding) is enabled. If the displayed value is 1, IP forwarding is enabled. If the output is 0, IP forwarding is not enabled and you need to enable it using the following command:

```
# sysctl -w net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
```

If you need to enable IP forwarding and you want to start Squid automatically at boot time, edit the file `/etc/sysctl.conf`, changing the line that reads

```
net.ipv4.ip_forward = 0
```

so that it reads

```
net.ipv4.ip_forward = 1
```

This will enable IP forwarding at boot time. Now that IP forwarding is turned on, you need to configure Squid.

Configuring Squid

The Squid configuration file on Fedora Core and RHEL systems is `/etc/squid/squid.conf`. The initialization script that controls Squid is `/etc/rc.d/init.d/squid`, which reads default values from `/etc/sysconfig/squid`. For your initial experimentation with Squid, only modify the Squid configuration file. It would easily consume several chapters to describe Squid's configuration and tuning adequately, so to get you up and running quickly, this section focuses on the bare minimum. You definitely want to spend some quality time with the configuration file's comments, because they are complete, understandable, and do a fine job of telling you more than you probably want to know about Squid's care and feeding. Table 17-2 lists the configuration settings with which you concern yourself in the short term.

Table 17-2 Key Squid Configuration Parameters

PARAMETER	DEFAULT VALUE	DESCRIPTION
<code>cache_effective_group</code>	<code>squid</code>	Identifies the group Squid runs as
<code>cache_effective_user</code>	<code>squid</code>	Identifies the user Squid runs as
<code>httpd_accel_host</code>	None	Defines the hostname of the real HTTP server (if using acceleration)

(continued)

Table 17-2 (continued)

PARAMETER	DEFAULT VALUE	DESCRIPTION
<code>httpd_accel_with_proxy</code>	<code>off</code>	Controls whether Squid runs as both an accelerator and a proxy
<code>httpd_accel_port</code>	<code>80</code>	Defines the port number of the real HTTP server (if using acceleration)
<code>httpd_accel_uses_host_header</code>	<code>off</code>	Enables Squid to function as a transparent proxy
<code>httpd_access</code>	<code>deny all</code>	Defines who can access the Squid server

As Table 17-2 shows, `cache_effective_user` and `cache_effective_group` identify the user ID (UID) and group ID (GID), respectively, under which Squid runs. The default values, `squid`, are the defaults configured in the Squid package shipped in Fedora Core and RHEL. You needn't change them.

`httpd_accel_with_proxy`, which defaults to `off`, controls whether Squid runs as a cache (or accelerator) and proxy or just as proxy. When set to `off`, Squid functions only as a proxy. If set to `on`, Squid works as both a cache *and* a proxy. If you are using Squid's caching functionality, you'll need to set `httpd_accel_port` to 80 and use `httpd_accel_host` to define the name the host running Squid. As it happens, the default port number of `httpd_accel_port` is 80, so you shouldn't need to change this. `httpd_accel_host` lacks a default value, so you'll need to change this value. If you want a transparent proxy server, set `httpd_accel_uses_host_header` to `on`. The default value, `off`, means that clients have to configure their Web clients to use a proxy server, which can be quite inconvenient for users and a pain for administrators to manage, especially across a LAN that is geographically dispersed or if some users are, shall we say, technically challenged.

The final value to configure is `httpd_access`, which controls who can access the Squid server and, therefore, who can surf the Web through the proxy. The default configuration is `deny all`, which prevents *any* user from accessing the proxy. As you can imagine, this policy is draconian. For experimentation purposes, set it to `allow all`, which permits all users to access the server. Yes, this is just as extreme as `deny all`, but the idea is to enable people to surf the Web. Make sure to learn how to use Squid's access control list (ACL) features to fine-tune the access policy.

The following listing shows the changes to make to `/etc/squid/squid.conf`:

```
cache_effective_user squid
cache_effective_group squid
httpd_accel_host squid.example.com
httpd_accel_with_proxy on
httpd_accel_port 80
httpd_accel_uses_host_header on
httpd_access allow all
```

Replace `squid.example.com` with the name of the system on which you are running Squid. To save needing to do so later, initialize Squid's cache using the command `squid -z`:

```
# squid -z
2005/03/02 22:54:59| Creating Swap Directories
```

Modifying Netfilter

Technically, modifying your netfilter firewall rules is an optional step. If you aren't using netfilter to maintain a firewall or to provide your LAN access to the Internet, you don't need to do anything with netfilter in order to get Squid to work. Presumably, however, you're running netfilter because you don't want Bad Things To Happen. If you don't fear crackers and other bad guys and so aren't running a netfilter firewall, skip this section.

If you're still reading, execute the following command:

```
# iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 \
-j REDIRECT --to-port 3128
```

Chapter 34 describes netfilter in detail, so don't feel as if you need to understand what this rule means. Simply stated, this rule sends HTTP requests to Squid, which services the request instead of the destination host (the external Web site). For the curious, an explanation of the specifics: This command updates the NAT or *network address translation* table (`-t nat`), appending a rule to the prerouting chain (`-A PREROUTING`). IP packets handled by the prerouting chain are modified as soon as they arrive and are not otherwise processed. The rule applies to TCP protocol packets (`-p tcp`) arriving on the network interface `eth0` (`-i eth0`) that are destined for port 80 (`--dport 80`). The modification that occurs is that packets going to port 80 are redirected (`-j REDIRECT`) to port 3128 (`--to-port 3128`), which is the port on which Squid listens.

Use the command `iptables -t nat -L` to verify that the rule is in effect:

```
# sudo /sbin/iptables -L -t nat
Chain PREROUTING (policy ACCEPT)
target     prot opt source                destination
REDIRECT   tcp  --  anywhere               anywhere               tcp dpt:http redirect
ports 3128
```

The output wraps, as indicated by the `\`, due to page layout constraints. As you can see, the `PREROUTING` chain has been modified to redirect HTTP packets to port 3128. Now you're ready to start Squid.

Starting Squid

To start Squid, use your old friend `service`:

```
# service squid start
Starting squid: . [ OK ]
```

If you wish, you can use the `chkconfig` command to set Squid to start and stop automatically:

```
# chkconfig --level 0123456 squid off
# chkconfig --level 345 squid on
```

The first command disables Squid in all run levels. The second command causes Squid to start when the system enters run levels 3, 4, or 5.

Testing the Configuration

To test the configuration, first configure your Web browser to use the proxy you've just set up. If you use Mozilla Firefox, select `Edit` ⇨ `Preferences` to open Mozilla's Preferences dialog box. (See Figure 17-4.)

On the `General` tab, click `Connection Settings` to open the `Connection Settings` dialog box. Click the `Manual proxy configuration` radio button and type the hostname or IP address of the proxy server in the `HTTP Proxy` text box. Type **3128** in the accompanying `Port` text box. The completed settings might resemble Figure 17-5.

Click `OK` to close the `Connection Settings` dialog box and `OK` again to save your changes and close the `Preferences` dialog box. Now, you should be able to type in a URL as you normally would and see the resulting page.

Nothin' to it, right?

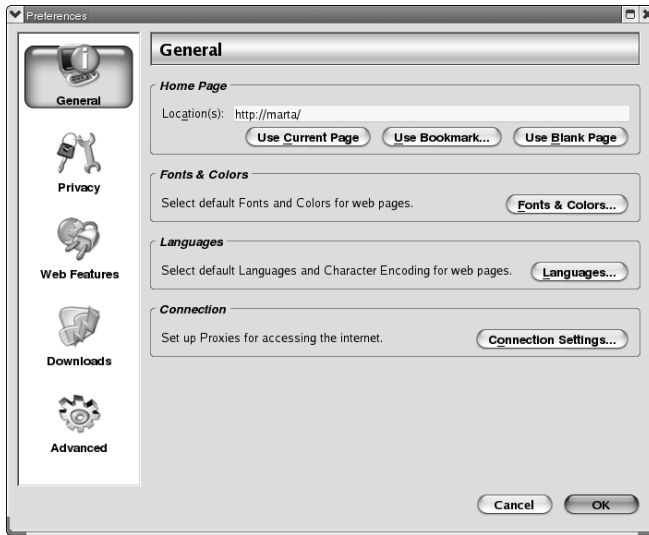


Figure 17-4 Firefox's Preferences dialog box.

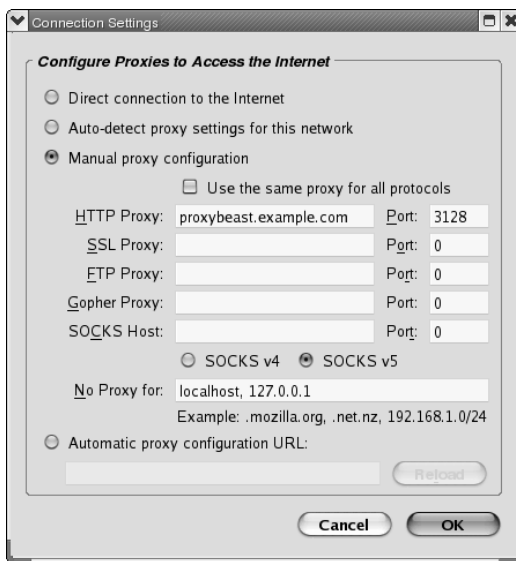


Figure 17-5 Firefox's Connection Settings dialog box.

Summary

For LANs that need to keep accurate time or if having 50 LAN clients each with a slightly different idea of what time it is offends your sense of order, the Network Time Protocol, NTP, is for you. NTP is easy to configure, brain-dead simple to use, and remarkably accurate and reliable. Perhaps the hardest part is deciding which remote time servers to use as reference clocks. Another useful, albeit infrequently requested, service is a caching proxy server. If you have limited bandwidth, Squid's caching capabilities can reduce bandwidth consumption, especially for sites that are more static than dynamic. Squid can also be used (or misused, perhaps) to limit who can access the Web.

Optimizing Network Services

IN THIS CHAPTER

- Getting the X Window System
- Optimizing NFS
- Optimizing NIS
- Optimizing Samba Networking
- Getting More from a Database Server

This chapter offers some tips for optimizing the performance of network services running on Fedora Core and RHEL systems. For example, X Window system desktops run faster when you reduce the amount of eye candy because giving up eye-popping graphics results in less bandwidth consumption when you run X applications across a network. Print queues can be configured to send large print jobs to high-speed printers. NFS disk farms can be designed to spread the load across multiple disks. The information in this chapter is only a starting point, however, and does not address larger issues of performance tuning, such as establishing a baseline, identifying performance goals, and ongoing performance monitoring. Rather, this chapter concerns itself with specific steps you can take to address known causes of slow performance. Chapter 32 goes into more of the theory behind performance tuning. As you read the tips and ideas in this chapter, bear in mind that you always have to balance your efforts along three axes: performance, usability, and security. Indeed, when it comes to server or application tuning, the pithy epigram might be, “Convenient, secure, fast. Pick any two.”

Optimizing the X Window System

Optimizing the X Window system is not easy. The core X protocol, despite what its detractors say, is actually pretty well tuned. Of course, as Keith Packard and Jim Gettys point out in their Usenix paper, “X Window System Network Performance,” (see <http://keithp.com/~keithp/talks/usenix2003/html/net.html>), protocol improvements are under way that should make X more efficient at the wire level. Nonetheless, the garden-variety system administrator usually lacks the time, not to mention the desire, to hack X at the level described in Packard’s and Gettys’ paper.

What exactly does “performance” mean vis-à-vis the X Window system? Depending on whom you ask, the phrase “X performance” might mean one or more of:

- *Drawing and refresh speed*, which refers to the time taken to render complex graphics (including 3-D graphics) on-screen
- *Startup and initialization time*, which refers to how long it takes the X Window System as a whole and, more pertinently, X applications to start and become ready to accept user input
- *Application responsiveness*, which refers to the user’s perception of how quickly or slowly an application responds to mouse and keyboard input
- *Client-server communication speed*, which refers to the round-trip time between an X client and an X server

For the most part, this section addresses the first three measurements of performance. The fourth metric, client-server communication speed, is difficult to address without either rewriting the X application in question or making significant changes in your network infrastructure. There are, naturally, some steps you can take to improve the performance of X or, more specifically, the Xorg system running on Fedora Core or RHEL.

- The biggest improvement comes from beefing up the hardware on which you are running X. There is simply no way around the fact that a faster CPU, a faster GPU (*graphics processing unit*) that supports hardware acceleration, more RAM, and more video RAM do more than anything else to speed up X. Naturally, hardware upgrades are not always an option, but they are the best option you have.
- Use a lightweight window manager. The eye candy and functionality provided by desktop environments such as GNOME and KDE carry a heavy price tag in terms of overall system performance. If, like us, you use X mostly as a platform for running xterms, a Web browser, and other graphical applications, you might not even miss the integration between applications that the desktop environments provide.

- Consider reducing the color depth at which you run X.org. Obviously, if you need millions of colors for heavy-duty image processing or video editing, 16-bit color will result in poor-quality images. An awful lot of the computing that takes place on a Fedora Core and RHEL systems is text processing, programming, and email. Twenty-four-bit and 32-bit color are computationally intensive; 16-bit color is much less demanding.
- Reduce the amount of eye candy on your desktop. A solid-color background is much less memory-intensive than a full-color wallpaper image. Transparent backgrounds in terminal emulators are cool but usually impose overhead on the windowing system to keep the background properly rendered and up to date. Shaped windows and borders are aesthetically pleasing but somewhat expensive in terms of resources.
- Run a local font server. Doing so makes the fonts you want available locally, so they do not have to be sent across the network. Similarly, you can free up memory and speed up the server's font handling by getting rid of fonts (that is, not loading them in `/etc/X11/xorg.conf`) that you do not use. For example, on a laptop computer, you can probably disable the 100-dpi fonts because you cannot run the X server at a high enough resolution to get any benefit from the better-quality fonts. Similarly, if you don't use the CID fonts and Cyrillic fonts, don't load them.
- Actively seek out lightweight X-based applications. We no longer use Mozilla, for example, because we have found that Firefox, the browser-only replacement for Mozilla, provides everything we need in a Web browser.
- Make sure that the X server is up-to-date. The X.org project is working to separate drivers from the rest of the system, which will enable you to download and install an updated driver for your card without having to refresh the entire X.org installation. Drivers sometimes improve dramatically from release to release, so stay informed about the driver for your card.
- Unload modules you do not use. The standard X.org installation, for example, loads the RECORD and XTRAP extensions. Edit `/etc/X11/xorg.conf` and comment out the lines that read:

```
Load "record"  
Load "xtrap"
```

Granted, these are only two modules with negligible memory and CPU impact, but every little bit helps.

- If you are motivated enough, consider recompiling X or at least the X applications you use most often for your specific system.

- Run X at a lower nice value, thereby increasing its priority relative to other processes. By increasing X's priority over other processes, you get a more responsive system. One way to do this is to start X using the command `nice -n -10 X : 0`. You need root access to use the `nice` command to increase a process's priority. If you run X using one of the display managers (`xdm`, `gdm`, or `kdm`), modify the `Xservers` file (`/etc/X11/xdm/Xservers`). Find the line that resembles the following:

```
:0 local /usr/X11R6/bin/X
```

Change the line so that it looks like the following, and then exit and restart X:

```
:0 local nice -n -10 /usr/X11R6/bin/X
```

TIP Counterintuitively, *lowering a process's nice value increases its priority relative to other processes. Think of it this way: the lower a process's nice value, the less nice it is to other processes.*

- If you run X for long periods of time, memory gets tied up as cache. Occasionally restarting X refreshes potentially stale caches and also frees up memory that might have been lost to leaks.

Don't try all of these suggestions at once because doing so will make it difficult, if not impossible, to pinpoint the change or changes that had the most affect. Rather, make one change and then test and record the result. Add each change incrementally, making sure to test and record the result, if any. At the end of the process, compare the results of your efforts to the pretuning performance to obtain a measure of how things have changed. The principle involved is to have a known baseline against which to measure the impact a given tuning measure makes.

Optimizing NFS

The subject of performance tuning deserves its own book, and, indeed numerous books exist on the general topic of performance tuning. Nevertheless, we can provide some general guidelines for improving NFS's performance and offer a few tips for maintaining a responsive NFS environment. The suggestions are far from exhaustive, and you must also take into account the needs and requirements of your network. To make the point that we make throughout this chapter, test your assumptions and tuning methods against baseline performance metrics taken from an untuned system. Comparing pre- with posttuning performance is the only way to ensure that your efforts make a positive difference.

NFS is sensitive to network performance, more specifically to network congestion and bandwidth problems. As a general rule, if NFS performance starts to degrade, you can be reasonably certain that heavy network traffic is the culprit. However, NFS traffic, especially at the server, tends to be “bursty,” characterized by periods of relative quiescence broken up with sharp upward spikes in activity. As a result, tuning efforts need to take into account such uneven access patterns to ensure optimal performance under load *and* during less strenuous periods.

Here are a few general suggestions you can apply to improve a system’s performance overall and the performance of an NFS server in particular. While most of these tips address a server system, they also have beneficial effects on a client system. Bear in mind that the impact of any changes will be more noticeable in large, heavily used NFS installations than in small installations where the total number of clients and servers is counted in single digits.

- Using a journaling file system offers two clear advantages for an NFS server. First, in the event of a crash, journaling file systems recover much more quickly than nonjournaling file systems. If you value your data, use a journaling file system on an NFS server. Second, journaling file systems need only update the journal to maintain data integrity, so an NFS server running a journaling file system “completes” I/O much faster because only the journal needs to be updated. After updating the journal, the server can safely issue an I/O completed reply to the clients. Meanwhile, the actual file system update occurs when the server is less busy.
- Spread NFS exported file systems across multiple disks and, if possible, multiple disk controllers. The purpose of this strategy is to avoid disk *hot spots*, which occur when I/O operations concentrate on a single disk or a single area of a disk. Similarly, distribute disks containing NFS exported file systems across multiple disk controllers. This measure reduces the amount of I/O traffic on any single controller, which improves the overall performance of the I/O subsystem.
- Replace IDE disks with serial ATA disks. If you have the budget for it, use FibreChannel disk arrays. FibreChannel, although markedly more expensive than IDE, serial ATA, and even SCSI, offers even faster performance. However, in small shops and for small servers, using FibreChannel is akin to killing gnats with a howitzer.
- If your NFS server is using RAID, use RAID 1/0 to maximize write speed and to provide redundancy in the event of a disk crash. RAID 5 seems compelling at first because it ensures good read speeds, which is important for NFS clients, but RAID 5’s write performance is lackluster, and good write speeds are important for NFS servers. Write performance is

critical because Linux's NFS implementation now defaults to synchronous mode (and has since about kernel version 2.4.7), meaning that NFS operations do not complete until the data is actually synced to disk.

CROSS-REFERENCE For more information about configuring NFS, particularly how to disable synchronous I/O, see Chapter 12.

- Consider replacing 10-Mbit Ethernet cards with 100-Mbit Ethernet cards throughout the network. Although only slightly more expensive than their 10-Mbit cousins, 100-Mbit cards offer considerably more throughput per dollar. The faster cards result in better network performance across the board, not just for NFS. Of course, to reap the benefits of 100-Mbit cards, they need to be used on clients and servers, and the gateways, routers, and switches must be capable of handling 100-MB speeds.
- In situations in which performance is the paramount concern, Gigabit Ethernet (1000 Mbit) is available, although expensive. Other high-performance network options, such as Myrinet and SONET (Synchronous Optical Networking), exist as well but are typically used as cluster interconnect solutions rather than as the underlying protocols for general purpose LANS or intranets.
- Replace hubs with switches. Network hubs, while less expensive than switches, route all network traffic across the same data channel. During periods of heavy activity, this single data channel can easily become saturated. Switches, on the other hand, transmit network packets across multiple data channels, reducing congestion and packet collisions and resulting in faster overall throughput.
- If necessary, dedicate one or more servers specifically to NFS work. CPU or memory-intensive processes, such as Web, database, and compute servers, can starve an NFS server for needed CPU cycles or memory pages.
- An increasingly inexpensive alternative is adding a NAS, or network-attached storage, device to the network. A NAS device is effectively a large box of disks attached to the network by assigning the NAS its own IP address. NAS devices speed up file access because file I/O is moved off the departmental or workgroup server and because NAS devices usually have special-purpose high-speed I/O chips. Parking NFS exports on a NAS can improve NFS performance significantly.
- A common NFS optimization is to minimize the number of write-intensive NFS exports. Automounted homes are useful, expected,

and therefore hard to eliminate, but for other exports it might be optimal to stop sharing heavily used project directories and require people to access the remote system that houses them via SSH or Telnet to minimize network traffic.

- In extreme cases, resegmenting the network might be the answer to NFS performance problems. Resegmenting the network to isolate NFS traffic on its own network segment reduces network saturation and congestion and allocates dedicated bandwidth to NFS traffic.

A good place to start evaluating NFS performance is to use the `nfsstat` command, which prints the NFS statistics maintained by the kernel. You can use `nfsstat` output to establish baseline performance metrics; to measure the effects, if any, of configuration changes; and to diagnose specific NFS performance problems. `nfsstat`'s general syntax is:

```
nfsstat [-acnr sz] [-o facility]
```

Table 18-1 explains `nfsstat`'s options.

By default, `nfsstat` displays statistics for both NFS and the underlying RPC service. Use the `-n` option to display on NFS statistics or the `-r` option to print only RPC statistics. Likewise, `nfsstat` displays client and server statistics unless you specify `-c` for client statistics (which makes little sense because the client does not maintain statistics in the current implementation) or `-s` for server statistics. The following listing displays NFS statistics only for an NFS server:

```
$ /usr/sbin/nfstat -n -s
Server nfs v2:
null      getattr  setattr  root      lookup    readlink
0          0% 0      0% 0      0% 0      0% 0      0%
read      wrccache write     create    remove    rename
0          0% 0      0% 0      0% 0      0% 0      0%
link      symlink  mkdir     rmdir     readdir   fsstat
0          0% 0      0% 0      0% 0      0% 0      0%

Server nfs v3:
null      getattr  setattr  lookup    access    readlink
0          0% 10861 43% 297      1% 6305 25% 95      0% 0      0%
read      write     create    mkdir     symlink    mknod
4151      16% 912    3% 621      2% 0      0% 0      0% 0      0%
remove    rmdir     rename    link      readdir    readdirplus
1236      4% 0      0% 0      0% 615    2% 0      0% 0      0%
fsstat    fsinfo    pathconf  commit
9          0% 9      0% 0      0% 91      0%
```

Table 18-1 nfsstat Command Options

OPTION	DESCRIPTION
-a	Prints all (NFS, RPC, and network) statistics for NFS clients and servers
-c	Prints NFS client statistics
-n	Prints only NFS statistics
-r	Prints only RPC statistics
-s	Prints only NFS server statistics
-z	Resets the kernel's NFS statistics counters to zero (not currently supported)
-o <i>facility</i>	Displays statistics for the specified <i>facility</i>

Notice that the statistics for NFS version 2 are all zeroes. This is so because this particular server is running NFS version 3. The displayed information shows the type of operation performed (arranged by RPC call), such as `getattr`, `read`, or `write`, the number of such operations performed, and the distribution of each operation. For example, 10,861 `getattr` operations have been performed, which represents 43 percent of all NFS server operations.

The `facility` argument for the `-o` option enables you to fine-tune the type of information `nfsstat` displays. `facility` can have one of the following values:

- **fh** — Displays utilization data on the server's file handle cache, including the total number of lookups and the number of cache hits and misses
- **net** — Shows network layer statistics, such as the number of received packets or number of TCP connections
- **nfs** — Displays NFS protocol statistics categorized by RPC call
- **rc** — Prints utilization data on the server's request reply cache, including the total number of lookups and the number of cache hits and misses
- **rpc** — Prints general RPC information

For additional information about improving the performance of NFS, see the Linux NFS-HOWTO on the NFS Web page at <http://nfs.sourceforge.net/nfs-howto/>. The NFS-HOWTO dedicates an entire section to performance tuning the Linux NFS implementation. Another valuable source of general performance tuning information, not only for NFS but also for all varieties of Linux issues, is the Linux Performance Tuning Web site at <http://linuxperf.nl.linux.org/>.

Optimizing NIS

How do you know when you have an NIS performance problem? If you issue a command that touches NIS-maintained information (such as `yppasswd`) that takes longer than you expect to complete, NIS might have problems. Other symptoms of NIS stumbling include error messages with phrases like “Busy try again later” or “Not responding.” Another sign of NIS problems is slow login times or apparent system hangs when you are logged in.

Frankly, with NIS+ for Linux no longer in development and traditional NIS in maintenance mode, not much can be done to improve NIS’s performance. Nevertheless, the following tips should help you avoid common problems with NIS’s speed.

- Make sure that your problem is with NIS and not with NFS-mounted exports. This is especially true when you are using automounted home directories, because problems that seem to be caused by NIS (such as slow login times) might actually be the result of problems mounting an exported home directory from a busy NFS server.
- Do not use recursive netgroups. A *recursive netgroup* is an NIS group defined in terms of other NIS groups. Recursive groups are quite convenient for administrators because they limit the maintenance burden, but resolving group names is time-consuming and thus slows down the server.

In large NIS installations, set up slave servers to reduce the load on the master server. Slave servers work best in situations in which the NIS maps are relatively static.

- Partition big NIS domains into smaller so-called *subdomains*. The idea here is to reduce the number of authoritative maps served by any single master server. This reduces the overall load on any one NIS server and minimizes the size of NIS maps that have to be pushed around.

As awful as doing so may sound, the best way to optimize NIS’s performance might be to replace it with something else, such as LDAP.

Optimizing Samba Networking

Probably the most important consideration for optimizing Samba performance is file transfer speed between the Samba server and clients. There are options that can be set in the `/etc/samba/smb.conf` file that will increase file transfer performance between the client and server. You can try them to determine if your performance increases after implementing them.

- **socket options** — These are controls on the networking layer of the operating system that enable tuning of the connection. For a complete list of options, refer to the `smb.conf` man page. A good choice for tuning your local network is to use `socket options = IPTOS_LOWDELAY TCP_NODELAY`.
- **dns proxy** — This option should be set to *no* unless your Samba server is acting as a WINS server. Setting this option to *no* prevents the server from doing unnecessary name lookups and increases system performance.
- **debug level** — This option should be set to 2 or less. Setting the debug level higher than 2 causes the server to flush the log file after each operation, a time-consuming process indeed.
- **level2 oplocks** — This option provides for caching of downloaded files on the client machine. Setting this option to `true` can increase system performance.

Getting More from a Database Server

Like the topic of performance tuning in general, database performance tuning would also require book-length treatment for fuller coverage. The following tips and references will get you started:

- **Use a different scheduler** — The Linux kernel supports four different process schedulers, each of which excel at handling certain types of workloads. Although you will want to test the results, the preferred and recommended scheduler to use for database systems is the so-called `cfq` scheduler. To use it, pass the following boot parameter to the kernel at boot time:

```
elevator=cfq
```
- **Use raw disk partitions** — Oracle supports storing data on *raw* or unformatted disk partitions. The advantage of using raw partitions is that Oracle's I/O processes bypass the kernel's data caches and buffers, significantly speeding up disk I/O.
- **Add memory** — Up to the limit supported by the underlying RDBMS and the kernel, database servers almost always benefit from adding RAM and/or swap.

- **Follow RDBMS vendor recommendations** — For the most up-to-date database server tuning techniques and suggestions, the RDBMS vendors are likely the best source of information. The following links will take you to the server optimization tips for the database server software discussed in this book:
 - *MySQL* — “Optimizing the MySQL Server”: <http://dev.mysql.com/doc/mysql/en/optimizing-the-server.html>
 - *PostgreSQL* — “PostgreSQL Hardware Performance Tuning”: www.postgresql.org/files/documentation/books/aw_pgsql/hw_performance
 - *Oracle* — “Tuning the Oracle 9i Database on Linux”: www.oracle.com/technology/oramag/webcolumns/2003/techarticles/scalzo_linux03.html

Summary

This chapter gave you some hints for improving the performance of Fedora Core and RHEL networking services. Although not strictly necessary on a server system, you learned some steps to take to speed up the X Window system. All of these suggestions boiled down to eliminating or at least reducing graphical eye candy. We also highlighted the performance issues that can arise with NFS and the measures you can implement to solve some of these problems. NIS can also be fixed, although the best “fix” is often to replace NIS with LDAP because LDAP is actively developed, whereas NIS is largely in maintenance mode. The Samba optimization tips largely centered around reducing the number and frequency of round trips to the server. Finally, we offered some suggestions to get better out of a database server, which mostly involved addressing I/O bottlenecks and using a different kernel process scheduler. Candidly, performance tuning requires a book all its own; we hope we have given you a good start.

PART

Three

Internet Services

Chapter 19: What Are Internet Services?

Chapter 20: Configuring BIND: The Domain Name System

Chapter 21: Configuring Mail Services

Chapter 22: Configuring FTP Services

Chapter 23: Configuring a Web Server

Chapter 24: Providing Web Services

Chapter 25: Optimizing Internet Services

What Are Internet Services?

IN THIS CHAPTER

- Learning about Secure Services
- Avoiding Less Secure Internet Protocols
- Using Your Linux Machine as a Server
- Configuring the xinetd Server
- Comparing xinetd with Standalone
- Configuring Linux Firewall Packages

An Internet Service can be defined as any service that can be accessed through TCP/IP-based networks, whether an internal network (intranet) or external network (Internet). Actually, TCP and IP are two of the protocols that are included in a group of protocols sometimes known as the *Internet protocols*. Since the two most frequently used or discussed protocols of the suite are TCP and IP, the entire suite is often referred to as just *TCP/IP*. Internet services can be provided through either secure or nonsecure TCP/IP connections. Common services are Telnet, FTP, SMTP, HTTP, ICMP, ARP, DNS, SSH, scp, sftp, and others.

The significance of TCP/IP as the basis for these services cannot be overlooked. TCP/IP provides a platform- and operating-system-independent protocol for these services. Any computer, running any operating system, can communicate with any other computer on the network if they both use TCP/IP protocols for establishing and maintaining the connection and formatting and transmitting the data.

The availability of a wide range of Internet services makes Linux machines versatile workhorses that can fulfill many different functions in a company's network. This chapter covers the wide range of common services that come standard with every Red Hat system.

CROSS-REFERENCE Chapter 11 explains the TCP/IP suite of protocols. Chapter 21 discusses mail transfer and SMTP. Chapter 22 explains setting up and using FTP. Chapter 23 covers HTTP and setting up an HTTP server.

Learning about Secure Services

Common services, such as Telnet and FTP, were written in the days when everyone trusted everybody else on the Internet. These services send all their traffic in plain text, including passwords. Plain-text traffic is extremely easy to eavesdrop on by anyone between the traffic's source and destination. Since the Internet has exploded in popularity, running insecure services such as these is not a good idea. That's why secure replacements have been developed. These replacements provide stronger authentication controls and encrypt all their traffic to keep your data safe. You should always run secure services instead of nonsecure services.

SSH

Secure Shell, also known as SSH, is a secure Telnet replacement that encrypts all traffic, including passwords, using a public/private encryption key exchange protocol. It provides the same functionality of Telnet, plus other useful functions, such as traffic tunneling.

This is what it looks like to SSH into a machine for the first time:

```
[vnavrat@buffy vnavrat$ ssh vnavrat@woolf.xena.edu
The authenticity of host 'woolf.xena.edu (123.456.789.65)'
can't be established.
RSA key fingerprint is
  b2:60:c8:31:b7:6b:e3:58:3d:53:b9:af:bc:75:31:63.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'woolf.xena.edu,123.456.789.65'
(RSA) to the list of known hosts.
vnavrat@woolf.xena.edu's password:
Welcome to woolf
Unauthorized usage prohibited. Please check your quotas.
vnavrat:~>
```

SSH asks you if you want to accept and trust the host key being sent to you as being the real key. This question is asked only once when you log in to a machine for the first time. After this first login, SSH behaves exactly like Telnet — you start SSH, it asks for your password, and then you see a regular terminal screen.

In addition to providing terminal access, SSH tunnels almost any other protocol through it. So, it is possible to tunnel POP, RCP, and other protocols through SSH to turn them into encrypted, more secure protocols. With enough imagination and practice, you can make almost anything more secure with SSH.

Following is an example of how to tunnel your mail through SSH to keep your password and mail encrypted and secure during transit. In this example, you use POP3 to retrieve your mail from the remote machine `buffy.xena.edu`. Normally you would tell your POP3 software to connect from your localhost to port 110 (the POP port) of `buffy.xena.edu`.

But in this example the first step is to configure your POP mailer to connect to port 16510 of your own machine, and put in the password for your account on `buffy.xena.edu`. The second step is to set up the SSH tunnel, which encrypts and forwards the traffic over the network to `terry.muhlenberg.edu`'s POP port.

To set up the SSH tunnel, type the following at the command line:

```
ssh -N -L 16510:127.0.0.1:110 terry@terry.muhlenberg.edu
```

And voilà! You are now sending and receiving your mail through an encrypted SSH tunnel.

scp

Secure Copy, also known as `scp`, is part of the SSH package. It is a secure alternative to RCP and FTP, because, like SSH, the password is not sent over the network in plain text. You can `scp` files to any machine that has an `ssh` daemon running.

The syntax of `scp` is

```
scp user@host:file1 user@host:file2
```

To see an example of copying a file named `camcontrol.doc` to remote host `main` from local host `terry` see Listing 19-1.

```
[terry@terry ~]$ scp camcontrol.doc terry@main:/home/terry
terry@main's password:
camcontrol.doc                                100% 117KB 117.0KB/s 00:00
```

Listing 19-1 Using the `scp` command to copy a file from the local PC to a remote PC.

And to copy a file named `nasatoc.doc` from remote host `main` to your current directory on local host `terry`, see Listing 19-2.


```
[terry@terry ~]$ scp main:/home/terry/nasatoc.doc .  
terry@main's password:  
nasatoc.doc                                100%   20KB   20.0KB/s   00:00
```

Listing19-2 Using the scp command to copy a file from the remote PC to a local PC.

If you have KDE installed on your system, you can use the Konqueror browser and the fish protocol to establish an SSH connection to a remote PC. Then you can drag and drop the files or directories you want to copy between the remote and local machine. Follow the steps here:

1. Open the Konqueror browser, enter **fish://<name or IP address of remote PC>** into the browser location field and press Enter. If this is the first time you are connecting to the remote host, you will be prompted about the connection, as shown in Figure 19-1. Click Enter to accept the information.
2. When prompted, enter the username and password and click OK to connect. Figure 19-2 shows the contents of my home directory on my home PC.
3. To copy or move files or directories from the remote PC to the local PC, select the files or directories you want to copy or move, right-click the selected items and either copy to or move the selected items from the pop-up menu, and browse to where you want to place them.

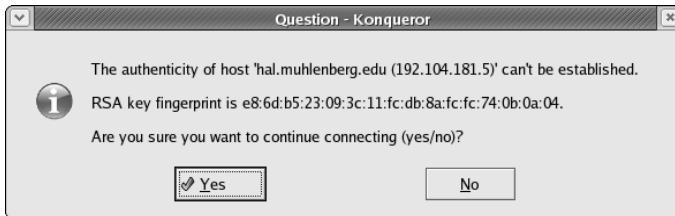


Figure 19-1 Using fish to make an SSH connection to a remote PC.

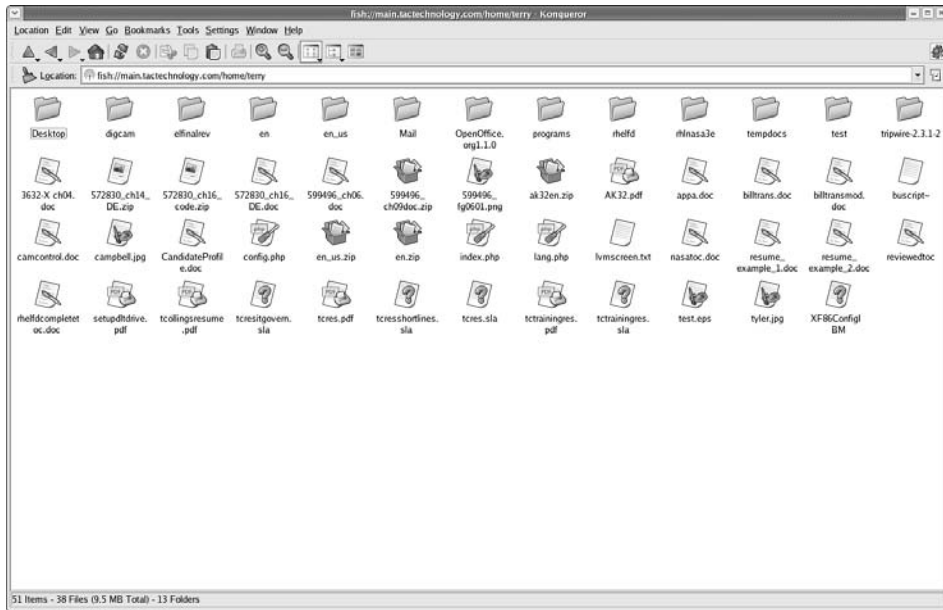


Figure 19-2 Using fish to display the directory contents of a remote PC.

sftp

Secure File Transfer Program, also known as sftp, is an FTP client that performs all its functions over SSH.

The syntax for sftp is:

```
sftp user@host:file file
```

For example, Listing 19-3 shows copying the file `tcres.pdf` from the remote PC main to the local PC.

```
[terry@terry ~]$ sftp terry@main:tcres.pdf tcres.pdf
Connecting to main...
terry@main's password:
Fetching /home/terry/tcres.pdf to tcres.pdf
/home/terry/tcres.pdf          100% 222KB 73.9KB/s 00:03
```

Listing 19-3 Using sftp to transfer files between a remote and local PC.

Less Secure Services

These are nonsecure services that should not be used, since they trust that the network is absolutely secure. Their secure equivalents should be used instead.

Using these services should be discouraged, because all their traffic is sent over the network in plain text. This means that anyone with a common sniffer such as `tcpdump` can see every keystroke that is typed in the session, including your users' passwords.

Telnet

Telnet is a protocol and application that enables someone to have access to a virtual terminal on a remote host. It resembles text-based console access on a UNIX machine.

Telnet is an application that's available almost everywhere. Because of this distribution, most beginning UNIX users use Telnet exclusively to communicate with other UNIX and NT machines. Since all Telnet traffic, including passwords, is sent in plain text, the Secure Shell (`ssh`) command should be used instead, if at all possible. SSH provides an equivalent interface to Telnet, with increased features, and most importantly, encrypted traffic and passwords.

This is what it looks like when you log into a machine with Telnet:

```
[terry@terry ~]$ telnet terry
Trying 127.0.0.1...
Connected to xena.
Escape character is '^]'.
Welcome to terry.muhlenberg.edu
login:
```

FTP

FTP is a ubiquitous file transfer protocol that runs over ports 20 and 21. For transferring software packages from anonymous FTP repositories, such as `ftp.redhat.com`, FTP is still the standard application to use. However, for personal file transfers, you should use `scp`. `scp` encrypts the traffic, including passwords. Once you have successfully logged on to an FTP server, you can type **help** for a list of available commands. Two important commands to remember are `put` to move a file from your machine to the remote machine, and `get` to pull a file from the remote server to your machine. To send multiple files you can use `mput`, and to retrieve multiple files you can use `mget`. `ls` or `dir` gives you a listing of files available for downloading from the remote site.

CROSS-REFERENCE To learn more about FTP commands, see Chapter 22.

rsync

rsync is an unencrypted file transfer program that is similar to RCP. It includes the added feature of allowing just the differences between two sets of files on two machines to be transferred across the network. Because it sends traffic unencrypted, it should be tunneled through SSH. Otherwise, don't use it. The rsync server listens on port 873.

rsh

rsh is an unencrypted mechanism for executing commands on remote hosts. Normally you specify a command to be run on the remote host on rsh's command line, but if no command is given, you are logged into the remote host using rlogin. rsh's syntax is

```
rsh remotehostname remotecommand
```

rlogin

rlogin is a remote login program that connects your terminal to a remote machine's terminal. rlogin is an nonsecure protocol, because it sends all information, including passwords, in plain text. It also enables an implicit trust relationship to exist between machines, so that you can use rlogin without a password.

finger

finger enables users on remote systems to look up information about users on another system. Generally finger displays a user's login name, real name, terminal name, idle time, login time, office location, and phone number. You should disable finger outside of your local network, because user information gathered from it could be used to compromise your system. The finger daemon listens on port 79.

talk and ntalk

talk and ntalk are real-time chat protocols. The talk server runs on port 517 and the ntalk server runs on port 518. To send someone else a talk request, type

talk or **ntalk username@hostname**. If that person's server is running a talk or ntalk daemon and the person is logged in, he or she will see an invitation to chat with you. talk and ntalk aren't as popular as they once were, since instant messenger clients have become very popular.

Using Your Linux Machine as a Server

The following sections give you an overview of what common server protocols are available on Linux.

HTTP

The most common Web server used on Linux is Apache. Apache is started out of a system's rc scripts. Apache is easily configurable, and its configuration files live in `/etc/httpd/conf/`. While Apache can be set to listen to many different network ports, the most common port it listens on is port 80.

CROSS-REFERENCE For more information on installing and configuring the Apache Web server, see Chapter 23.

sshd

The secure shell daemon (sshd) is started out of the system's rc scripts. Its global system configuration files are in `/etc/ssh`, and users' SSH configuration files are in `$HOME/.ssh/`. The SSH server listens on port 22.

TIP sshd can be configured to run on an alternative port. Running SSH on a port other than 22 comes in handy if port 22 is being blocked by a firewall. Running SSH on a different port also adds a small measure of security through obscurity. Automatic scanners used by hackers will miss that SSH is running on your machine if they don't find it running on the standard port they expect.

ftpd

The FTP daemon uses ports 20 and 21 to listen for and initiate FTP requests. Its configuration files `ftpaccess`, `ftpconversions`, `ftpgroups`, `ftphosts`, and `ftpusers`, are located in the `/etc` directory.

CROSS-REFERENCE You can find more information on setting up the FTP daemon in Chapter 22.

DNS

The Domain Name Service (DNS), which maps IP addresses to hostnames, is served by the `named` program on port 53. Its configuration file is `named.conf` in the `/etc` directory.

CROSS-REFERENCE To read more about setting up DNS on your Linux machine, see Chapter 20.

Configuring the xinetd Server

`xinetd` is a replacement for `inetd` that adds more security and functionality. `inetd` is the old workhorse of the Linux networking world, and `xinetd` is an improvement on an important program that has been around for several years. It incorporates new features that have been desired by system administrators for a few years now.

`xinetd` starts at system boot time, and waits and listens for connections to come in on the ports to which they are assigned in their `conf` files. After a connection request is made, if the service requested requires that a new server be spawned, then `xinetd` spawns a new server and keeps listening for new connection requests on the service port.

One of the most notable improvements of `xinetd` over `inetd` is that anyone can start network services. With `inetd`, only `root` can start a network service, and that restriction leads to a host of security problems.

`xinetd` supports encrypting plain-text services such as the `ftp` command channel by wrapping them in `stunnel`.

`xinetd` also enables you to do access control on all services based on different criteria, such as remote host address, access time, remote hostname, and remote host domain. In the past this kind of access control could be accomplished only with tools like `tcpwrappers`, or firewall software. Even then, `tcpwrappers` could only reliably control TCP traffic.

`xinetd` also takes the extra security step of killing servers that aren't in the configuration file and those that violate the configuration's access criteria. It can help prevent denial of service (DOS) attacks by limiting normal functions that can cripple the machine if there are too many of them occurring at the same time.

For example, `xinetd` can limit the number of incoming connections to the whole machine or from a single host to prevent network overflow attacks. It can limit the number of processes that are forked by a network service. To prevent the machine from being slowed to a crawl because of the incoming network traffic, `xinetd` can also stop a service if it is driving the machine's load up too high.

Log capabilities have also been improved in `xinetd`. For each service it runs, it can log the remote user and host address, the length of time the service has been running, and failed access control attempts.

xinetd is flexible enough to enable you to utilize increased security measures such as chrooted environments.

You may notice that the `xinetd.conf` file (see Listing 19-4) is much shorter than `inetd.conf`, making it easier to read through and customize. The last line says that all the files in the `/etc/xinetd.d` directory are read into the `xinetd.conf` file as well. Each service started by xinetd gets its own dedicated file in the `/etc/xinetd.d` directory. This way you can tell, with a glance at the `xinetd.d` file listing, what services are being started by it. You can easily enable or disable a service by setting the value of `disable` = to either yes or no.

```
#
# Simple configuration file for xinetd
#
# Some defaults, and include /etc/xinetd.d/
defaults
{
    instances            = 60
    log_type              = SYSLOG authpriv
    log_on_success        = HOST PID
    log_on_failure        = HOST
    cps                   = 25 30
}

includedir /etc/xinetd.d
```

Listing 19-4 The `xinetd.conf` file.

The last line of the `xinetd.conf` file shown in Listing 19-4 shows the directory location of the individual configuration files for the services that use xinetd. Listing 19-5 shows the `Krb5-telnet` configuration file that is located in the `/etc/xinet.d` directory.

```
# default: off
# description: The kerberized telnet server accepts normal telnet sessions, \
#             but can also use Kerberos 5 authentication.
service telnet
{
    flags            = REUSE
    socket_type      = stream
    wait            = no
    user            = root
    server          = /usr/kerberos/sbin/telnetd
    log_on_failure   += USERID
    disable         = yes
}
```

Listing 19-5 The xinetd config file entry required to set up a Kerberos 5 Telnet daemon.

If you use a Telnet server on your system, you should be aware that by default `telnetd` does not start up with `xinetd`. To start a Telnet daemon in `xinetd`, you should use the `groups = yes` option within the `xinetd.conf` file and change the `disabled = line` from `yes` to `no`.

Comparing `xinetd` and Standalone

Which services are standalone, and which are started from `xinetd`? Sometimes it can get confusing keeping track of how to start a certain service, or keep it from running in the first place. To control a service, you need to know what spawns it. Here is a general listing of what services are spawned from superservers, such as `xinetd`, and which are started on their own from `rc` scripts or `root`. The key differences between things that run standalone and those started by the `xinetd` server is the amount of config information required and availability — that is, whether the associated service needs to be constantly available and instantly responsive, or whether that service only needs to be available in response to occasional incoming requests. Services like a Web server could certainly be run by `xinetd`, but the overhead of starting and stopping it repeatedly would make your system incredibly slow and perhaps even crash it.

`xinetd`-Started Services

The following services are started from `xinetd`. Each of these services should ideally have its own file in the `/etc/xinetd.d` directory. You should look in that directory to find the appropriate file and open the file to change whether it is enabled or disabled.

- **chargen** — Random character generator that sends its traffic over TCP
- **daytime-udp** — Gives you the time over UDP
- **finger** — User information lookup program
- **kshell** — Restricts user access to the shell
- **rlogin** — Service similar to Telnet, but enables trust relationships between machines
- **swat** — Samba Web Administration tool
- **time** — Gives you the time
- **chargen-udp** — Random character generator that sends its traffic over udp
- **echo** — Echoes back all characters sent to it over TCP
- **gssftp** — Kerberized FTP server

- **rsh** — Remote shell
- **talk** — A talk (real-time chat) server
- **time-udp** — Gives you the time over UDP
- **comsat** — Notifies users if they have new mail
- **echo-udp** — Echoes back all characters sent to it over UDP
- **klogin** — Kerberos's answer to rlogin
- **ntalk** — A talk (real-time chat) server
- **rsync** — Remote file transfer protocol
- **telnet** — Telnet server
- **wu-ftp** — An FTP server
- **daytime** — Gives you the time over TCP
- **eklogin** — Encrypting kerberized rlogin server
- **krb5-telnet** — Kerberized Telnet server
- **rexec** — Provides remote execution facilities
- **sgi_fam** — File-monitoring daemon
- **tftp** — Trivial File Transfer Program

Standalone Services

These services are started from the `rc` scripts specifically written for them in the `rc` directories. You can enable or disable these services from those directories.

- **apache** — Web server
- **sshd** — SSH server
- **sendmail** — Mail server
- **qmail** — Mail server
- **postfix** — Mail server
- **thttpd** — Semilightweight Web server
- **boa** — Lightweight Web server
- **named** — DNS server
- **xfs** — X font server
- **xdm** — X display manager
- **portmap** — Maps RPC services to ports
- **rpc.quotad** — Serves quota information
- **knfsd** — Userspace portion of the NFS daemon

- **rpc.mountd** — NFS mount server
- **rpc.yppbind** — NIS server
- **squid** — Web proxy server
- **nessusd** — Penetration-testing server
- **postgresql** — Database server
- **mysql** — Database server
- **oracle** — Database server

CROSS-REFERENCE See Chapters 6 and 8 for more information about **xinetd** and standalone servers.

Configuring Linux Firewall Packages

Linux provides a few different mechanisms for system security. One of these mechanisms is Linux's firewall package. The primary firewall package available is **iptables**. **iptables** is a packet filtering firewall.

CROSS-REFERENCE To read more about **iptables** and Linux system security, see Chapter 34.

iptables is Fedora's and Red Hat's built-in IP firewall administration tool. Using **iptables** enables you to run a personal firewall to protect your Linux machine. If the Linux machine is a routing gateway for other machines on your network, it can act as a packet-filtering network firewall if more than one network interface is installed.

CROSS-REFERENCE For more information on setting up a Linux firewall, see Chapter 34.

Summary

As a general rule, you should endeavor to use programs that encrypt their network traffic, instead of using those that use unencrypted protocols.

If you must use a plain-text protocol, such as POP or rsync, wrapping it in SSH or SSL keeps your passwords and data safe from network listeners. SSH is versatile enough to wrap around almost anything.

In general, most services are started from `xinetd`. If a service you're looking into is not started from `xinetd`, it is probably being started from the machine's `rc` scripts.

Linux's network capabilities have grown a great deal in the past couple of years. Currently, a properly configured Red Hat server can function as many different network devices — anything from an AppleTalk or Samba server to an LDAP server or firewall. Once you get a grasp of how the servers are started and configured in general, the possibilities of what you can do are nearly endless.

Configuring BIND: The Domain Name System

IN THIS CHAPTER

- Understanding DNS
- Installing the Software
- Understanding Types of Domain Servers
- Examining Server Configuration Files
- Configuring a Caching DNS Server
- Configuring a Secondary Master DNS Server
- Configuring a Primary Server
- Checking Your Configuration

This chapter provides information about the Domain Name System (DNS), which is used for name address resolution. In this chapter, you learn how the Domain Name System works for finding hosts on TCP/IP networks. You also learn about different types of DNS servers and how to configure them for your own network. After configuring your servers, you'll learn about some diagnostic tools that can be used to check the configuration and performance of your DNS.

Understanding DNS

Name address resolution is, simply stated, the conversion of people friendly names into computer friendly numbers. Remember from Chapter 11 that each interface on the network has an IP address. This address is expressed as a group of numbers referred to as a dotted quad group. These groups of numbers present no problem to the computers in the network, but it is difficult for

humans to remember many groups of numbers. So, you need to be able to enter names and then have these names converted into numbers. Each time you type a Web site's address into your browser, the Domain Name System (DNS) goes to work. You enter names that are easy for you to remember, and the names are resolved into numbers that computers find easy to understand. Enabling efficient human/machine interaction is the function of name address resolution. In this chapter you learn how to install and configure the Domain Name System, which provides this name address resolution.

First, take a look at domain names and their organization using the domain name `tactechology.com`. The first part of this domain name, `tactechology`, is the name of the company, institution, or organization. The next part, after the period (dot in today's vernacular) is called the top-level domain (TLD). In addition to the `com` top-level domain, you will find many others. Table 20-1 shows other top-level domains in the United States.

Table 20-1 United States Top-Level Domains

TOP-LEVEL DOMAIN	MEANING
com	A TLD typically used to register a business
edu	A TLD assigned to an educational institution
gov	A TLD given to a U.S. government agency
mil	A TLD used by a branch of the U.S. military
net	A TLD used by a network affiliated organization
org	A TLD used by a noncommercial organization
int	An TLD assigned to an international organization
us	The U.S. domain, with each listing as a lower level
biz	Another TLD used for businesses
info	A TLD that can be used for any type of organization
name	A TLD used to register sites for individuals
museum	A TLD used to register a museum
coop	A TLD assigned to a cooperative organization
aero	A TLD used by the air transport industry
pro	A TLD not yet active but assigned to professions
travel	A TLD that should be available in late 2005 used by travel related companies

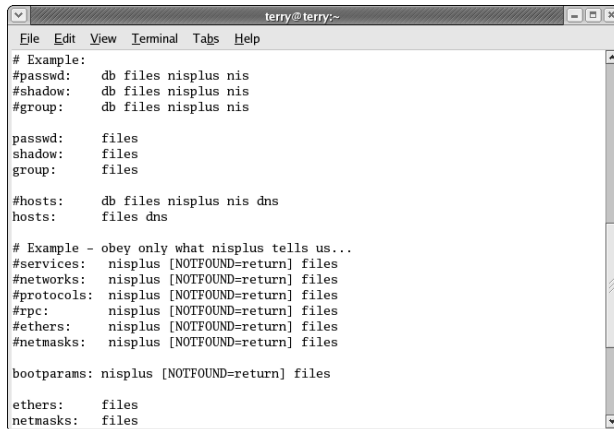
Top-level domains in other countries include a two-letter suffix, such as `fr` for France or `su` for Switzerland. Not all of the top-level domains are the same as the top-level U.S. domains, but a company in France could be `http://www.frenchcompany.com.fr`.

Large domains may be further broken down into subdomains. For example, the `tactech` site is `www.tactechtechnology.com`. Perhaps the accounting department runs their own Web server. To find their Web server, `tactech` contains the subdomain `acct.tactechtechnology.com`. An individual computer in the accounting department also has a hostname, for example, `payables`. The complete name for this computer is then `payables.acct.tactechtechnology.com`, and you can find its IP address by using the DNS to look it up.

When you type in a hostname, your system uses its resources to resolve names into IP addresses. One of these files is `/etc/nsswitch.conf` (`nsswitch` means name service switch), which contains a line telling the system where to look for host information. Figure 20-1 shows the `/etc/nsswitch.conf` file and the `hosts` line.

The information following the word `hosts` tells the system to first look at the local files, and then to use the Domain Name Service (DNS) to resolve the names into IP numbers. One of the local files searched is the `/etc/hosts` file.

CROSS-REFERENCE See Chapter 11 for more information about the `/etc/hosts` file.



```

# Example:
#passwd:    db files nisplus nis
#shadow:    db files nisplus nis
#group:     db files nisplus nis

passwd:     files
shadow:     files
group:      files

#hosts:     db files nisplus nis dns
hosts:      files dns

# Example - obey only what nisplus tells us...
#services:  nisplus [NOTFOUND=return] files
#networks:  nisplus [NOTFOUND=return] files
#protocols: nisplus [NOTFOUND=return] files
#rpc:       nisplus [NOTFOUND=return] files
#ethers:    nisplus [NOTFOUND=return] files
#netmasks: nisplus [NOTFOUND=return] files

bootparams: nisplus [NOTFOUND=return] files

ethers:     files
netmasks:   files

```

Figure 20-1 The `nsswitch.conf` file tells the resolver routines where to look for IP addresses.

The hosts file contains IP addresses and hostnames that you used on your sample network. So, why couldn't you use this file for name resolution? Well, you could on a small internal network that you controlled and that did not have very many IP addresses. But, the hosts file is not a manageable solution on a large network, because it is an impossible task to keep it up to date. You could not have control over every IP address.

After the system looks in the hosts file and fails to find the address, the next file checked is `/etc/resolv.conf`. This file contains the IP addresses of computers that are known as domain name servers, and these are listed in `/etc/resolv.conf` as just name servers. Figure 20-2 shows the `/etc/resolv.conf` file on one of the computers I use at work.



```
terry@terry:~  
File Edit View Terminal Tabs Help  
[terry@terry ~]$ cat /etc/resolv.conf  
; generated by /sbin/dhclient-script  
search muhlenberg.edu  
nameserver 192.104.181.5  
[terry@terry ~]$
```

Figure 20-2 The `/etc/resolv.conf` file points to domain name servers used to resolve IP addresses.

You could list up to three name servers, but one is all that is required. It is a good idea to list a second name server, thus enabling a connection to the second one of the name servers in case the first name server is down or not reachable. Don't list more than three name servers, because any over three are ignored.

Installing the Software

So far you have learned about name address resolution and the structure of domains in the United States. Now you will learn about the Domain Name System servers that resolve the name into IP numbers. The most common DNS server used in current Fedora Core and Red Hat Enterprise Linux distributions is BIND, or the Berkeley Internet Name Daemon. The latest release of BIND

can be obtained from the Internet Software Consortium at www.isc.org. A very convenient and easy way to install the latest version of BIND is to look for the distribution specific package. Check the Web site for your distribution to locate the RPM for BIND.

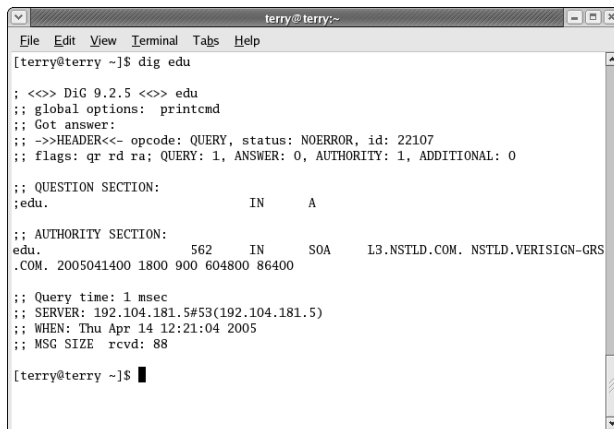
The RPM version of BIND is included with the Red Hat installation CD-ROMs. The RPM file can be installed at the command line by using the `rpm` command.

CROSS-REFERENCE See Chapter 30 for instructions on installing and upgrading packages.

Understanding Types of Domain Servers

A top-level domain server, one that provides information about the domains shown in Table 20-1, is typically referred to as a root name server. A search for `www.muhlenberg.edu` looks to the root name server for `.edu` for information. The root name server then directs the search to a lower-level domain name server until the information is found. You can see an example of this by using the `dig` command to search for the root name server for `.edu`, as shown in Figure 20-3.

The figure shows the root name server that provides information for the `.edu` domain. You can continue the search for the second-level domain by adding the name of the domain you are looking for, as shown in Figure 20-4.



```

[terry@terry ~]$ dig edu

; <<> DiG 9.2.5 <<> edu
;; global options: printcmd
;; Got answer:
;; ->HEADER<- opcode: QUERY, status: NOERROR, id: 22107
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 0

;; QUESTION SECTION:
;edu.                IN      A

;; AUTHORITY SECTION:
edu.                  562     IN      SOA     L3.NSTLD.COM. NSTLD.VERISIGN-GRS
.COM. 2005041400 1800 900 604800 86400

;; Query time: 1 msec
;; SERVER: 192.104.181.5#53(192.104.181.5)
;; WHEN: Thu Apr 14 12:21:04 2005
;; MSG SIZE rcvd: 88

[terry@terry ~]$

```

Figure 20-3 A search for the top-level root name servers for `.edu`.


```

terry@terry:~$ dig muhlenberg.edu

; <<> DiG 9.2.5 <<> muhlenberg.edu
;; global options: printcmd
;; Got answer:
;; ->HEADER<- opcode: QUERY, status: NOERROR, id: 49738
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1

;; QUESTION SECTION:
;muhlenberg.edu.                IN      A

;; ANSWER SECTION:
muhlenberg.edu.                86400   IN      A      192.104.181.5

;; AUTHORITY SECTION:
muhlenberg.edu.                86400   IN      NS      hal.muhlenberg.edu.

;; ADDITIONAL SECTION:
hal.muhlenberg.edu.            86400   IN      A      192.104.181.5

;; Query time: 1 msec
;; SERVER: 192.104.181.5#53(192.104.181.5)
;; WHEN: Thu Apr 14 12:24:25 2005
;; MSG SIZE rcvd: 82

terry@terry:~$

```

Figure 20-4 A search for the second-level domain shows the authoritative name server.

After you have found the domain you are looking for, information about that domain is provided by its local domain name servers. The three types of local domain name servers are master, or primary, slave, or secondary, and caching-only servers.

- **Master** — The master contains all the information about the domain and supplies this information when requested. A master server is listed as an authoritative server when it contains the information you are seeking and it can provide that information.
- **Slave** — The slave is intended as a backup in case the master server goes down or is not available. This server contains the same information as the master and provides it when requested if the master server cannot be contacted.
- **Caching** — A caching server does not provide information to outside sources; it is used to provide domain information to other servers and workstations on the local network. The caching server remembers the domains that have been accessed. Use of a caching server speeds up searches since the domain information is already stored in memory, and the server knows exactly where to go rather than having to send out a request for domain information.

Where does the information that the master and slave servers provide come from? The server(s) have been configured to provide it when asked. In the next section, you learn how to configure a server to provide domain name information.

Examining Server Configuration Files

Before you begin to configure your servers, you need to take a closer look at the files you need to configure. The number of files you will use depends on the type of BIND server you want to use. You need five files to set up the named server. Three files are required regardless of the configuration as a master, slave, or caching-only server, and two additional configuration files are used on the master server. There is also the script that is used to start the servers after they are configured, for a total of six possible files.

The three required files are:

- **named.conf** — Found in the `/etc` directory, this file contains global properties and sources of configuration files.
- **named.ca** — Found in `/var/named`, this file contains the names and addresses of root servers.
- **named.local** — Found in `/var/named`, this file provides information for resolving the loopback address for the localhost.

The two additional files required for the master domain server are:

- **zone** — This file contains the names and addresses of servers and workstations in the local domain and maps names to IP addresses.
- **reverse zone** — This file provides information to map IP addresses to names.

The following script is used to start the BIND server:

- **/etc/rc.d/init.d/named** — This is the BIND server initialization file used to start BIND. A sample file is installed with the RPM from the Installation CDs.

You begin with the `/etc/named.conf` file. An example of this file is shown in Listing 20-1.

```
//
// named.conf for Red Hat caching-nameserver
//

options {
    directory "/var/named";
    dump-file "/var/named/data/cache_dump.db";
    statistics-file "/var/named/data/named_stats.txt";
    /*
```

Listing 20-1 The `/etc/named.conf` file for a caching-only server installed by the default installation of BIND. (*continued*)

```

    * If there is a firewall between you and nameservers you want
    * to talk to, you might need to uncomment the query-source
    * directive below. Previous versions of BIND always asked
    * questions using port 53, but BIND 8.1 uses an unprivileged
    * port by default.
    */
    // query-source address * port 53;
};

//
// a caching only nameserver config
//
controls {
    inet 127.0.0.1 allow { localhost; } keys { rndckey; };
};

zone "." IN {
    type hint;
    file "named.ca";
};

zone "localdomain" IN {
    type master;
    file "localdomain.zone";
    allow-update { none; };
};

zone "localhost" IN {
    type master;
    file "localhost.zone";
    allow-update { none; };
};

zone "0.0.127.in-addr.arpa" IN {
    type master;
    file "named.local";
    allow-update { none; };
};
include "/etc/rndc.key";
```

Listing 20-1 *(continued)*

The named.conf file

Look at this file in more detail beginning with the lines starting with `//`. These are comment lines, and anything following them is ignored. Commands are passed to the file in the form of statements. Several of these statements are

shown in the sample file, but you actually can use seven configuration statements. These are listed here with a brief explanation of their function.

- **options** — Lists global configurations and defaults
- **include** — Gets information from another file and includes it
- **acl** — Specifies IP addresses used in an access control list
- **logging** — Specifies log file locations and contents
- **server** — Specifies properties of remote servers
- **zone** — Specifies information about zones
- **key** — Specifies security keys used for authentication

Information about the statement is contained within curly braces and terminated by a semicolon: *{information about server};*.

Options

The `options` statement is typically the first section of `named.conf`, and it contains information about the location of the files used by `named`. You can use only one `options` statement, but you can have more than one value for that statement. In the sample file shown in Listing 20-1, the `options` statement shows the path to the location where additional configuration files used by `named` are located. By specifying the directory where other files are located, you make it unnecessary to list the entire path to the file; you just list the name of the file for any files shown in the remainder of `named.conf`. Options statements use the following syntax:

```
options {
    value "property";
}
```

The list of values that can be used in the `options` statement is quite long. These values are shown, listed alphabetically, in Table 20-2.

Table 20-2 Options Values and Their Meanings

VALUE	MEANING
<code>allow-query</code>	Accepts queries only from hosts in the address list (by default queries are accepted from any host). Usage: <code>allow-query {"address-list"};</code>
<code>allow-transfer</code>	Zone transfers are accepted only by hosts in the address list (by default transfers are allowed to all hosts). Usage: <code>allow-transfer {"address list"};</code>

(continued)

Table 20-2 (continued)

VALUE	MEANING
auth-nxdomain	The server responds as an authoritative server (defaults to yes). Usage: auth-nxdomain "yes or no"; (choose one).
check-names	Hostnames are checked for compliance with the RFC. Usage: check-names "master or slave or response warn or fail or ignore"; (choose one from each group).
cleaning-interval	Specifies the time period before expired resource records are removed by the server (defaults to 60 minutes). Usage: cleaning-interval "number"; (specify number in minutes).
coresize	Specifies largest size for core dump files. Usage: coresize "size"; (specify size in bytes).
datasize	Limits server memory usage. Usage: datasize "size"; (specify size in bytes).
deallocate-on-exit	Detects memory leaks (default is no). Usage: deallocate-on-exit "yes or no"; (choose one).
directory	Path of the directory where server configuration files are located. Usage: directory "path to directory"; (specify path).
dump-file	If <code>named</code> receives a SIGINT signal, it dumps the database to the file specified here (defaults to <code>named_dump.db</code>).
fake-iquery	If set to yes, the server sends a fake reply to inverse queries rather than an error (default is no). Usage: fake-iquery "yes or no"; (choose one).
fetch-glue	If set to yes, the server obtains the glue records for a response (default is yes). Usage: fetch-glue "yes or no"; (choose one).
files	Limits number of concurrently open files (default is unlimited). Usage: files "number"; (specify number).
forward	If set to <code>first</code> , the servers listed in the <code>forwarders</code> option are queried first, and then the server tries to find the answer itself. If set to <code>only</code> , just the servers in the <code>forwarders</code> list are queried. Usage: forward "first or only"; (choose one).

Table 20-2 (continued)

VALUE	MEANING
forwarders	Shows IP addresses of servers to forward queries (default is none). Usage: forwarders "IP addresses of servers"; (specify IP addresses).
host-statistics	If set to yes the server keeps statistics on hosts (default is no). Usage: host-statistics "yes or no"; (choose one).
interface-interval	Specifies interval for searching the network for new or removed interfaces (default is 60 minutes). Usage: interface-interval "time"; (specify time in minutes).
listen-on	Specifies port and interfaces on which server listens for queries (default is port 53). Usage: listen-on "port {address list}"; (specify port number and address list).
max-transfer-time-in	Specifies time server waits for completion of inbound transfer (default is 120 minutes). Usage: max-transfer-time-in "time"; (specify time in minutes).
memstatistics-file	When deallocate-on-exit is set, specifies the file where memory statistics are written (defaults to named.memstats). Usage: memstatistics-file "path to file"; (specify path and filename).
multiple-cnames	When set to yes, enables multiple CNAME usage (default is no). Usage: multiple-cnames "yes or no"; (choose one).
named-xfer	Specifies path to the named-xfer program. Usage: named-xfer "path to file"; (specify path).
notify	When zone files are updated, this option, when set to yes, sends DNS NOTIFY messages (default is yes). Usage: notify "yes or no"; (choose one).
pid-file	Name of file holding process ID. Usage: pid-file "path to file"; (specify path and filename).
query-source	Specifies port and IP address used to query other servers. Usage: query-source "address port"; (specify IP address and port).

(continued)

Table 20-2 (continued)

VALUE	MEANING
<code>recursion</code>	The server recursively searches for query answers (default is yes). Usage: <code>recursion "yes or no";</code> (choose one).
<code>stacksize</code>	The amount of stack memory the server can use. Usage: <code>stacksize "number";</code> (specify the amount of memory).
<code>statistics-interval</code>	The time interval for logging statistics (default is 60 minutes). Usage: <code>statistics-interval "time";</code> (specify the time in minutes).
<code>topology</code>	Sets server preference for remote servers. Usage: <code>topology {"address list"};</code> .
<code>transfer-format</code>	When set to <code>one-answer</code> , only one resource record per message is sent. When set to <code>many-answers</code> , as many records as possible are transmitted in each message (default is one). Usage: <code>transfer-format "one-answer many-answers";</code> (choose one).
<code>transfers-in</code>	Maximum concurrent inbound zone transfers (default is 10). Usage: <code>transfers-in "number";</code> (specify the number).
<code>transfers-out</code>	Maximum concurrent outbound transfers. Usage: <code>transfers-out "number";</code> (specify the number).
<code>transfers-per-ns</code>	Limits inbound transfers from a single server (default is two). Usage: <code>transfers-per-ns "number";</code> (specify the number).

Include

The `include` statement lists the path and name of any files that you want to be included with the `named.conf` file. Use the same syntax used in the `options` statement to specify the path.

Acl

This option lets you specify a list of IP addresses in an access control list. Only hosts on this list have access to the server.

Logging

The logging statement is where you specify your server's logging options. The logging statement contains two additional items, the channel and the category.

The channel is where you specify the location of the logged information. Logged information can be written to a file, sent to the syslog, or thrown away by specifying the appropriate command. Choosing to send the information to a file gives you several additional choices for handling the information. You can set the number of versions to keep, the size of the files, and whether the severity of the information and the time and category are included with the other information in the file.

The syntax for the logging statement is similar to the syntax for the `option` statement. The following commands send the information to a file. Items in *italics* indicate information you need to enter.

```
logging {
    channel channel_name {
        file path to file
        versions specify number or unlimited
        size specify size in bytes }
}
```

If you want to send the information to the syslog, the syntax is:

```
logging {
    channel channel_name {
        syslog (choose where to send from following choices)
        (kern,user,mail,daemon,auth,syslog,lpr,news,uucp,cron,\
        authpriv,ftp,local0 thru local7)
```

To discard the information, choose `null` as the destination.

```
logging {
    channel channel_name {
        null;}
}
```

Next, you can set the severity level for the information written to the file or syslog. This section follows the sections shown previously for file or syslog. You also indicate here whether you want the time, category, and severity

included. If you are discarding the information, you don't need to set these parameters.

```
severity choose from critical,error,warning,notice,info,debug\ level,dynamic  
print-time choose yes or no  
print-severity choose yes or no  
print-category choose yes or no  
};
```

The category is where you specify the type of information to log. This value follows the severity and print parameters and takes the following syntax:

```
category category name {  
channel name; channel name;  
};
```

You can choose from over 20 categories. These are shown, alphabetically, in Table 20-3.

Table 20-3 Logging Categories

CATEGORY	TYPE OF INFORMATION LOGGED
cname	Information about CNAME references
config	Information about configuration files
db	Information about databases
default	The default if nothing is selected
eventlib	Information about event system debugging
insist	Details about failures from internal consistency checking
lame-servers	Information about LAME servers
load	Information about zone loading
maintenance	Information about maintenance
ncache	Information about negative caching
notify	Information about tracing the NOTIFY protocol
os	Information about operating system problems
packet	Dumps of all sent and received packets
panic	Information about faults that shut down the server
parser	Information about processing configuration commands
queries	Information about all received DNS queries

Table 20-3 (continued)

CATEGORY	TYPE OF INFORMATION LOGGED
response-checks	Information about response-checking results
security	Information about security status of server
statistics	Information about server statistics
update	Information about dynamic updates
xfer-in	Information about inbound zone transfers
xfer-out	Information about outbound zone transfers

Using the categories from the `logging` statement, you can obtain a large quantity of information about your server. This information can be useful if you are having problems with your DNS. You can enable logging for the area that you think is causing your problem and then read the appropriate log to find any messages that might indicate an error with your configuration.

server

In the `server` statement you can set the properties of a remote server. You can specify whether to send queries to the remote server from the local server, and you can set the method used for transferring information. The syntax for this statement is the same as for other statements. The following are valid values:

- **bogus** — Specify yes or no. No is the default and indicates that queries are sent to the remote server. Yes means that the remote server is not queried.
- **transfer** — Specify the number of transfers you want to allow.
- **transfer-format** — Specify whether you want one-answer or many-answers.
- **keys** — Specify key ID (currently not implemented).

zones

The remainder of the listings in `/etc/named.conf` shown in Listing 20-1 are zone statements. These zone statements refer to files that are called zone files. Additional options for zone statements exist, of course. Each zone statement begins with the word `zone` followed by the domain name and the data class. The four data classes are `in`, `hs`, `hesiod`, and `chaos`. If no type is specified, the default is `in`, for Internet.

Next follows the `type` option which specifies whether the server is a master, is a slave/stub, or is the hints file. A stub server loads only NS records, not the entire domain. The hints file is used to initialize the root cache and contains a list of root servers.

Next is the name of the zone file for the specified zone. This is a pointer to the file containing the data about the zone. You look at a zone file in detail a little later in this chapter.

The other options for the zone statements are listed here, along with an explanation of their function:

- **allow-query** — Accepts queries only from hosts in the address list (by default queries are accepted from any host).
- **allow-transfer** — Zone transfers are accepted only by hosts in the address list (by default transfers are allowed to all hosts).
- **allow-update** — Hosts in the address list are allowed to update the database.
- **also-notify** — Servers in the address list are sent a notify message when the zone is updated.
- **check-names** — Hostnames are checked for compliance with the RFC.
- **max-transfer-time-in** — Specifies the time the slave waits for a zone transfer.
- **notify** — When zone files are updated, this option, when set to `yes`, sends DNS NOTIFY messages (default is `yes`).

These options are the same as those shown in the options statement section and have the same function here. When listed in the options section, they apply to all zones, but if listed with a specific zone, they apply only to that zone. Settings listed in a specific zone section override those globally set in the options statement.

Zone Files

Zone files contain resource records (RR) about IP addresses. A typical zone file is shown in Listing 20-2.

```
@           IN SOA                localhost root (
                                42           ; serial
                                3H           ; refresh
                                15M          ; retry
                                1W           ; expiry
                                1D )         ; minimum

           IN NS                localhost
localhost IN A                  127.0.0.1
```

Listing 20-2 A zone file is used to provide information about a DNS server.

A zone file can contain many types of RRs, which are listed in the order in which they generally appear in the zone files, and explained next.

SOA – Start of Authority

The start of authority (SOA) is the first line in the zone file. The SOA identifies the name server as the authoritative source for information about that domain. Each zone file has only one SOA, and it contains the following data:

```
@ IN SOA main.tactechnology.com. mail.tactechnology.com. (/
    2000052101 ; Serial
    8h          ;Refresh
    2h          ;Retry
    1w          ;Expire
    1d)         ;Minimum TTL
```

The first character in the SOA line is a special symbol that means “to look at this domain.” IN means Internet. SOA means Start of authority.

In this example, the authoritative server for this domain is `main.tactechnology.com.`, and `mail.tactechnology.com.` is the e-mail address of the administrator. Note the trailing period after the domain names. If these are not included, the domain name is appended to the entry.

The opening parenthesis enables the first line to be extended so that anything between the opening and closing parenthesis is considered one line.

The information within the parenthesis is passed to other name servers, secondary masters that use this information to update their records. The line containing `2000052101 ; Serial` is the serial number of the file. Secondary servers compare this number with their stored information. If the numbers are the same, the information has not changed, and it is not necessary to download this file. If the serial numbers are different, the file is downloaded to update the information in the secondary server. The serial number can be any number desired as long as it can be incremented to indicate a revision to the file. The semicolon indicates that what follows to the end of the line is a comment.

- **Refresh** — The amount of time the server should wait before refreshing its data.
- **Retry** — The amount of time the server should wait before attempting to contact the primary server if the previous attempt failed.
- **Expire** — Means that if the secondary master is unable to contact a primary master during the specified period, the data expires and should be purged.
- **TTL** — Specifies the time to live for the data. This parameter is intended for caching name servers and tells them how long to hold the data in their cache.

All of the information contained by the SOA may be placed on one line, but it is usually written as shown previously. The order of the items is significant in the SOA header. Following the SOA header information are lines containing additional server information. Two of these lines, containing the abbreviations NS and A are shown in Listing 20-2. These abbreviations are explained here:

- **NS** — Name servers in this domain.
- **A** — The IP address for the name server.
- **PTR** — Pointer for address name mapping.
- **CNAME** — Canonical name, the real name of the host.
- **MX** — The mail exchange record. The MX record specifies the mail servers for the domain. If more than one MX server is present priority is determined by the address with the lowest number receiving the highest priority.
- **TXT** — Text information. You can enter descriptive information here.
- **WKS** — Well-known service. You can enter descriptive information here.
- **HINFO** — Host Information usually shows type of hardware and software.

The Reverse Zone File

In the previous example you looked at a sample zone file that uses the domain name. This method is called forward address resolution, since it uses a name to find an IP number and is the most common use of name resolution.

You can also find a name from an IP number, and this is called reverse address resolution. All you need to do is enter the IP address, and the server returns the domain name. Reverse address resolution requires the use of a reverse zone file. Listing 20-3 shows a sample reverse zone file.

```
@      IN      SOA      localhost. root.localhost. (
                                1997022700 ; Serial
                                28800      ; Refresh
                                14400      ; Retry
                                3600000    ; Expire
                                86400 )    ; Minimum
1      IN      NS       localhost.
1      IN      PTR      localhost.
```

Listing 20-3 A typical reverse lookup zone file.

Configuring a Caching DNS Server

Now you know which files need to be configured and what information that needs to be placed in them. You are ready to set up your own domain name servers. In this section you set up a caching server. As stated earlier, three files are used in all three types of server configurations. In a caching-only server, these three files make up the entire configuration. Fortunately for us, the default installation of BIND used on Red Hat Enterprise Linux creates the configuration files needed for a caching name server, but it is always a good idea to look at the files to be sure that they are correct.

Begin by verifying the zone information in `/etc/named.conf`. When you installed the BIND package, the `/etc/named.conf` file was created and it contained zone information for your localhost, but you need to check it to be sure. You are looking for two zone lines: one indicated by a `“.”`, referencing the file `named.ca`, and one shown as `“0.0.127.in.addr.arpa”`, referencing `named.local`.

Next, you need to check the configuration of the `/var/named/named.local` file. This file contains the domain information for the localhost and is typically created when BIND is installed. You usually don't have to make any changes to `/var/named/named.local`. This domain is a reverse domain in that it is used to map the loopback address 127.0.0.1 to the localhost name.

You need to check the `/etc/nsswitch` file to be sure it contains the following line:

```
hosts:      files dns
```

You need to check the `/etc/resolv.conf` file to make sure that the IP address (127.0.0.1) of your localhost is listed as a name server.

Finally, you need to check that your `/etc/host.conf` contains the word `bind`.

After you have completed all of the previous steps, it is time to start the `named` daemon and check your work.

Type **`service named start`** at a command prompt, press Enter, wait for the prompt to return, and then type **`rndc status`**. Press Enter. You will see output similar to this:

```
number of zones: 8
debug level: 0
xfers running: 0
xfers deferred: 0
SOA queries in progress: 0
query logging is off
server is up and running
```

You have successfully configured and started your caching name server.

Configuring a Secondary Master DNS Server

Next, you set up a secondary master or slave server for your domain. The procedure is similar to, and not much more difficult than, what you have already done. You already have three of the files in place and only need to slightly modify the `/etc/named.conf` file and add two more files to complete the slave configuration.

On the server you want to be the slave, go to the `/etc/named.conf` file and add two more zones, one for the forward lookup of your server and one for the reverse lookup. For the forward lookup, you need to add the following. (For this example, the master server is called `main.tacttechnology.com`, and the slave is `p200.tacttechnology.com`. Be sure to use your own domain name and IP address instead of the examples shown.)

```
zone "tacttechnology.com" {
    notify no;
    type slave;
    file "tacttech.com";
    masters { 192.168.1.1; };
};
```

For the reverse lookup, you add this section:

```
zone "1.168.192.in-addr.arpa" {
    notify no;
    type slave;
    file "tac.rev";
    masters { 192.168.1.1; };
};
```

After modifying the `/etc/named.conf` file, the configuration of the slave server is complete and you can move on to configuring the master server.

Configuring a Primary Master Server

The `/etc/named.conf` file on the master server also needs to be modified. Assuming that you already set up this server as a caching-only server, you just need to add the following lines to `/etc/named.conf`. (This example uses the names you defined earlier; be sure to use your own names and IP addresses.)

```
zone "tacttechnology.com" {
    notify no;
    type master;
    file "tacttech.com";
};
```

For the reverse lookup you add this section:

```
zone "1.168.192.in-addr.arpa" {
    notify no;
    type master;
    file "tac.rev";
};
```

Notice that you used the same names for the files on the master server as the slave server. This is because these files are downloaded by the slave in a zone file transfer and stored on the slave in the files shown by the file option. You did not create the files on the slave but created them on the master server.

NOTE The filenames, hostnames, and IP addresses I use here are just examples for the purpose of this explanation. Be sure to use filenames, hostnames, and IP addresses appropriate to your organization.

You now need to create the zone files that are referenced by the `/etc/named.conf` file. First you create the file `/var/named/tactech.com` by beginning with the Start of Authority section (SOA). For an explanation of the information contained in zone files, refer to the zone file section earlier in this chapter.

```
@ IN SOA main.tactechtechnology.com.mail.tactechtechnology.com. ( /
    200005203 ; Serial/
    8h; Refresh/
    2h; Retry/
    1w; Expire/
    1d); Minimum TTL/
```

Next you add name server and mail exchange information:

```
NS main.tactechtechnology.com./
NS terry.tactechtechnology.com./
MX 10 main;Primary Mail Exchanger/
MX 20 p200;Secondary Mail Exchanger/
```

Finally, you add information about your localhost, and mail, FTP, and Web server. You can also add information about every workstation on your network.

Next, you set up the reverse lookup zone file, which is called `tac.rev`. Again, you need to start with the SOA header as shown:

```
@ IN SOA main.tactechtechnology.com. mail.tactechtechnology.com.(
    200005203;Serial
    8h ; Refresh
    2h ; Retry
    1w ; Expire
    1d) ; Minimum TTL
```


Next, you add the information about your name servers and their IP addresses.

```
      NS      main.tactechnology.com.
1      PTR    main.tactechnology.com.
2      PTR    p200.tactechnology.com.
```

If you have done everything as explained here, your name server should be working properly after you restart it. You made some changes to the `/etc/named.conf` file, so before you can check what you did, you need to restart the named daemon.

NOTE The named daemon must be restarted whenever you make changes to `/etc/named.conf`. To restart named, you just need to enter the following command:

```
service named restart
```

Checking Your Configuration

After you finish configuring your master DNS server, you can check your configuration to be sure that it's working. You can use several tools to do your check. I talk about two of them here. Just be sure to substitute your domain and IP information when you run the commands. If your system is set up correctly, you should obtain similar results.

The Host Program

`host` enables you to find an IP address for the specified domain name. All that is required is the domain name of the remote host, as shown here with the command on the first line and the output from the command on the second line:

```
[root@laptop root]# host tactechnology.com
tactechnology.com has address 12.129.206.112
```

You can also search for resource record types by using the `-t` option and the type of resource record that you want to search. For example, if you want to find information about the mail server for a domain, enter the following command and receive the following output:

```
[root@terry named]# host -t mx tactechnology.com
tactechnology.com mail is handled by 10 mail.tactechnology.com.
```

The dig Program

`dig` can be used for debugging and obtaining other useful information. The basic syntax is:

```
dig (@server) domain name (type)
```

Items shown in parentheses are optional. Listing 20-4 shows an example of using `dig` to request information about the DNS servers at my school.

```
[root@terry named]# dig muhlenberg.edu

; <<>> DiG 9.2.2-P3 <<>> muhlenberg.edu
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 50760
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1

;; QUESTION SECTION:
;muhlenberg.edu.                IN      A

;; ANSWER SECTION:
muhlenberg.edu.                86400   IN      A      192.104.181.5

;; AUTHORITY SECTION:
muhlenberg.edu.                86400   IN      NS      hal.muhlenberg.edu.

;; ADDITIONAL SECTION:
hal.muhlenberg.edu.            86400   IN      A      192.104.181.5

;; Query time: 5 msec
;; SERVER: 192.104.181.5#53(192.104.181.5)
;; WHEN: Thu Sep  9 14:37:54 2004
;; MSG SIZE rcvd: 82
```

Listing 20-4 Using `dig` to search for IP addresses.

`dig` can also be used to do reverse lookups by using the `-x` switch and specifying the IP address, as shown in Listing 20-5. The first line in the listing is the command entered at the command prompt, and the remaining lines are the output from the command.

```
[root@terry named]# dig -x 192.104.181.5

; <<>> DiG 9.2.2-P3 <<>> -x 192.104.181.5
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 26961
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2
```

Listing 20-5 Using `dig` to search for domain names. (*continued*)

```
;; QUESTION SECTION:
;5.181.104.192.in-addr.arpa.      IN      PTR

;; ANSWER SECTION:
5.181.104.192.in-addr.arpa. 86400 IN      PTR      hal.muhlberg.edu.

;; AUTHORITY SECTION:
181.104.192.in-addr.arpa. 86400 IN      NS      hal.muhlberg.edu.
181.104.192.in-addr.arpa. 86400 IN      NS      stimpy.nwip.muhlberg.edu.

;; ADDITIONAL SECTION:
hal.muhlberg.edu.          86400   IN      A        192.104.181.5
stimpy.nwip.muhlberg.edu. 86400   IN      A        192.104.181.207

;; Query time: 6 msec
;; SERVER: 192.104.181.5#53(192.104.181.5)
;; WHEN: Thu Sep  9 14:39:46 2004
;; MSG SIZE rcvd: 170
```

Listing 20-5 *(continued)*

Both of the listings provide information about the exact same servers. In Listing 20-4, I used a domain name for the search, and I received the IP addresses associated with the domain name. In Listing 20-5, I searched using an IP address, and I received the host and domain name of the servers.

Summary

This chapter provided a brief but concise overview of the Domain Name System using a small, simple network as an example. You looked at name address resolution and found out how names that are easy for people to remember are converted to numbers that computers can understand. You obtained the BIND software package and installed it on your system and located the files that have to be configured for BIND to work. After configuring the files, you set up a caching name server and tested its operation. You then configured a slave and master name server and looked at some of the diagnostic utilities available for the BIND package.

Configuring Mail Services

IN THIS CHAPTER

- Email Explained
- Introducing SMTP
- Configuring Sendmail
- Using the Postfix Mail Server
- Serving email with POP3 and IMAP
- Maintaining Email Security

Using electronic mail, you can send messages to and receive messages from other computer users anywhere in the world. Email, as it is more commonly known, is a very powerful and useful tool. In this chapter you learn how email works and then configure your systems to put email to work for you.

Email Explained

An email message, just like a letter sent through regular mail, begins with a sender and ends with a recipient. Between sender and recipient are many postal workers who ensure that the letter is properly handled. Even though the sender and recipient never see these workers, they perform essential functions in moving the mail. Email works in a similar fashion, and although there are not many people between the sender and receiver, programs perform the same function. These programs use network protocols to do the job of ensuring that the message goes from sender to receiver. In this chapter you configure email to run across TCP/IP protocols.

CROSS-REFERENCE Chapter 11 explains the TCP/IP protocol suite.

Before configuring an email client or server, you need to understand how email works and the programs to use or make available to your users. Several key components are essential for email to work properly, and as a system administrator it is your responsibility to configure the following items.

- Programs:
 - A mail user agent for users to be able to read and write email
 - A mail transfer agent to deliver email messages between computers across a network
 - A mail delivery agent to deliver messages to users' mailbox files
 - A mail-notification program to tell users that they have new mail (optional)
- The SMTP protocols for packaging email and transferring email messages between MTAs

Other communication and mail-storage components include TCP/IP ports through which programs communicate, mail queues in which to store email during various phases of the delivery process, and mailbox files, which are the final destinations for delivered email messages.

This chapter explains these items in greater detail in later sections. The next section tracks email messages through the process from sending to delivery to illustrate how the various components work together. After learning how the components work, you configure these components to build a fully functioning email system for your server and clients.

Tracing the Email Delivery Process

In classic UNIX style, email creation, transmission, and delivery consist of several discrete tasks, each of which is handled by a program designed specifically for that task. The value of this segmented method is that you can replace one element, such as the program used to compose email, without affecting the functionality of the entire system. The usual email delivery process involves three components, a mail user agent, a mail transfer agent, and a mail delivery agent.

Mail User Agent (MUA)

To be able to send mail, you or your users need a program called a *mail user agent*, commonly abbreviated as MUA and widely referred to as a *mail client*. The MUA provides users an interface for reading and writing email messages. Two types of MUAs are available: those that have a graphical user interface (GUI), such as Mozilla Thunderbird or KMail, and those that have a command

line interface (CLI), such as Mutt or elm. Whether your MUA is a GUI or CLI, the functionality is the same. After composing an email message, the MUA sends it to a mail transfer agent (MTA), which transmits the message across the network.

Mail Transfer Agent (MTA)

A mail transfer agent, usually abbreviated as MTA, is the program that sends the message across the network. MTAs work without any intervention by the user. In fact, most users are unaware of the MTA; they just see their mail client. The MTA reads the information in the *To:* section of the email message and determines the IP address of the recipient's mail server. Then the MTA tries to open a connection to the recipient's server through a TCP/IP port, typically port 25. DNS, the Domain Name System, plays a role in mail deliver, too. MTAs query a domain's DNS mail exchanger, or MX, record, to determine the host that is responsible for receiving mail for that domain. If the MTA on the sending machine can establish a connection to the destination MTA, it transmits the message using the Simple Message Transfer Protocol (SMTP).

In some cases, the message might not be immediately deliverable, so the sending MTA will store in a local queue and attempt retransmission after a set amount of time. In other cases, the receiving MTA might refuse delivery, causing the message to *bounce* and be returned to the sender's MUA. Assuming that all goes well, the receiving MTA adds header information to the message. This header contains information for tracking the message and ensuring that it is delivered. Finally, the receiving MTA passes the message to yet another program, the mail delivery agent, for delivery to the recipient's mailbox.

NOTE The default MTA installed on your Fedora Core or RHEL system is called Sendmail. You can change this, as described later in this chapter.

Mail Delivery Agent (MDA)

After the MDA (sometimes referred to as a *local delivery agent*, or LDA) receives the message from the MTA, it stores the new message in the recipient's mailbox file. The mailbox file, known traditionally as *the mail spool*, is almost always identified by the recipient's username. One popular MDA on Fedora Core and RHEL systems is named procmail. Ordinarily, the location of the user's mailbox file is `/var/spool/mail/username`. In some cases, the user's mailbox file will be a file or directory in the user's home directory. Procmail isn't the only possible MDA. Users who rely on POP or IMAP clients to retrieve their email ordinarily don't use procmail. Rather the POP or IMAP client is itself the MDA.

WHY IS IT CALLED BIFF?

Mail notification tools are called *biffs* because the original BSD mail notification tool was a utility named `biff`. The BSD `biff` utility was in turn named after a dog that chased frisbees in the halls at the University of California (Berkeley) during the development of 4.2BSD. A legend claimed that the dog would also bark whenever the mailman arrived, but `biff`'s author denies this (see the `biff` entry in the Jargon File (version 4.4.7)), <http://catb.org/~esr/jargon/html/B/biff.html>.

The next step in the process is optional. Some users (and many MUAs) provide some sort of visual or audible notification that new mail has arrived. Classically, these programs were known as *biffs*, for example, the eponymously named `biff`; `xbiff`, an X-based version; and `tknewsbiff`, a Tk-based `biff` that tells when new Usenet news arrives. Some *biffs* play a sound when new mail arrives, others display an icon of a full mailbox, and still others increment an on-screen counter showing the number of new or unread messages.

Even the lowly shell invokes the `biff` mail notifier, checking your mailbox file once a minute. If new mail has arrived, the shell displays a message just before it displays the next system prompt. It won't interrupt a program you're running. You can adjust how frequently the mail notifier checks and even which mailbox files to watch. See the `biff` man page (**man biff**) for the specifics.

The last step of email delivery occurs when the message recipient reads the message using her own MUA. Kind of anticlimactic, eh?

Introducing SMTP

In the section describing the MTA, you learned that messages are sent between MTAs using SMTP. *SMTP* is the TCP/IP protocol for transferring email messages between computers on a network. SMTP specifies message movement between MTAs by the path the message takes. It is sometimes described as a *store-and-forward* protocol because messages transmitted via SMTP don't necessarily go directly from the sender's MTA to the recipient's MTA. Rather, network conditions and other considerations (not relevant here) might cause email messages to pass through other MTAs on other network computers. These intermediate MTAs briefly store the message before forwarding it to another MTA or to a gateway that sends it to an MTA on another network.

The SMTP protocol can transfer only ASCII text. It can't handle fonts, colors, graphics, or attachments. If you want to be able to send these items, you need to add another protocol to SMTP, MIME. MIME stands for Multipurpose Internet Mail Extensions. MIME is implemented on top of the SMTP protocol and

enables you to add colors, sounds, and graphics to your messages and still deliver them via SMTP. For MIME to work, you must have a MIME-compliant MUA (most are).

If it isn't clear yet, SMTP (with or without MIME) has a major shortcoming: it assumes that the sending and receiving MTAs are *always* connected to the Internet. What happens when this assumption fails, say, for users who have dial-up Internet access? Enter POP, the Post Office Protocol.

Understanding POP3

Two other protocols are part of the email process: Post Office Protocol version 3, known as POP3, and Internet Message Access Protocol version 4, known as IMAP4. POP3 was developed to solve the problem of what happens to messages when the recipient is not connected to the network. POP3 runs on a server that is connected to a network and that continuously sends and receives mail. The POP3 server stores any messages it receives until the message recipients request them.

Without POP3 (or IMAP), and MUA and MTA can't communicate. That is, if you want to read your mail, you need POP3 (or IMAP) to retrieve messages that arrive at your MTA while you're offline. POP3 uses the MTA's storage to hold messages until they are requested. When users want to check their email, they connect to the POP3 server and retrieve messages that were stored by the server. After retrieval, the messages are stored locally (that is, locally to the MUA) and you can use the MUA on your PC to read them at your leisure. Of course, your MUA has to understand the POP3 to be able to communicate with the POP3 server, but most MUAs speak fluent POP3 these days so this is rarely a problem. The messages you retrieve to your PC are then typically removed from the server.

What happens if users don't want to remove their email from the server and instead want to access their email from any given PC? Why, they use Internet Message Access Protocol, of course!

Understanding IMAP4

The Internet Message Access Protocol version 4 (IMAP4) provides much more sophisticated email-handling functionality than SMTP or POP3 do. IMAP4 has more features. IMAP4 enables you to store email on a networked mail server, just as POP3 does. The difference is that POP3 requires you to download your email before your MUA reads it, whereas IMAP4 enables your email to reside permanently on a remote server, from which you can access your mail. And you can do so from your office, your home, your PDA, your cell phone, or anywhere else. Your MUA must understand IMAP4 to retrieve messages from an IMAP4 server.

WHEN EMAIL PROBLEMS HAPPEN

Email gets lost in transport about as often as the dog eats your homework. This achievement is quite impressive when you consider the volume of email sent each day. On occasion, email messages get garbled or misrouted, just as a post office sometimes does with snail mail. Email can become garbled or lost due to lightning storms and other events that disrupt power supplies. Any network that is using unshielded 10Base-T cable is more susceptible to email interruptions because the cabling is not protected against atmospheric power disturbances such as lightning. Luckily, this issue is becoming less important as cabling standards improve. However, it is always a good idea to install a surge protector on your system to protect yourself from damage caused by fluctuations in the power supply.

Power failure and electrical interference notwithstanding, however, most email problems are the result of user or system administrator error. For example, the user may type an incorrect email address. Misconfiguring an email server causes email problems. Make sure that mailboxes sit in the right directory for your distribution.

If you add file system quotas to mailboxes, be aware that at some point, users may not be able to receive new messages until they (or you) clean up the mail files. Quotas are especially an issue for IMAP users. POP3 and file system quotas intersect at the user's home directory, where as IMAP4 and file system quotas meet at the mail spool on the server. If a POP3 user goes over quota, simply deleting unnecessary files will allow additional email to be downloaded. If an IMAP4 user goes over quota, it might not even be possible for that user to connect to the server to delete messages. Worse, until the quota issue is solved, the over-quota user's mail will bounce because the mailbox is full.

CAUTION POP3 and IMAP4 don't interoperate. While there are email clients and servers that speak both protocols, you can't use a POP3 client to communicate with an IMAP4 server or an IMAP4 client to communicate with a POP3 server. When you configure an email server, you must decide whether your users need POP3 or IMAP4 functionality (or both). IMAP4 servers usually require much more disk space than POP3 servers because the email remains on the mail server unless the users or system administrator delete it.

Configuring Sendmail

A number of mail transport agents are available for Red Hat Enterprise Linux, including Sendmail, Postfix, and Qmail. The most widely used MTA is Sendmail, which is the default MTA on Fedora Core and RHEL systems. Postfix is

growing in popularity and is considerably easier to configure, so you will learn how to make it the default MTA in the section titled “Configuring Postfix.”

Before you begin configuring Sendmail, be sure it’s installed. It probably is, because the installation program installs Sendmail. But just to be sure, check it out. The following example shows how to check using the `rpmquery` command. The output shows not only that Sendmail is installed, but which version of Sendmail is installed:

```
# rpmquery -a | grep sendmail
sendmail-cf -8.13.4-1.1
sendmail-doc-8.13.4-1.1
sendmail-devel-8.13.4-1.1
sendmail-8.13..4-1.1
```

The version numbers you see might be slightly different.

You’ll need at least the `sendmail` package. If you intend to perform any sort of Sendmail reconfiguration, you’ll want the `sendmail-cf` package, too. If you want to know more about Sendmail than this chapter tells you, the `sendmail-doc` package is handy. Unless you are doing Sendmail development work, though, you probably won’t need the `sendmail-devel` package. If Sendmail isn’t installed install it before proceeding. If Sendmail is installed, make sure that it starts at boot time. You can use the following `chkconfig` command to verify this:

```
# chkconfig --list sendmail
sendmail          0:off  1:off  2:on   3:on   4:on   5:on   6:off
```

In the example, the `sendmail` service, which controls the Sendmail MTA, will start in run levels 2, 3, 4, and 5 and stop in run levels 0, 1, and 6. If you don’t see this type of output, use the following `chkconfig` command to correct it:

```
# chkconfig --levels 0123456 sendmail off
# chkconfig --levels 2345 sendmail on
```

Finally, ensure that Sendmail is running and start it if it isn’t. You have several ways to check whether Sendmail is running. Pick your favorite. The following command uses `ps` and `pidof` to query the process table for running Sendmail instances:

```
# ps uw -p $(pidof sendmail)
USER      PID %CPU %MEM  VSZ  RSS TTY      STAT START   TIME COMMAND
root      3706  0.0  0.6  8652 3292 ?        Ss   22:45   0:00 sendmail: accepti
ng connections
smmsp     3714  0.0  0.5  7704 2580 ?        Ss   22:45   0:00 sendmail: Queue r
unner@01:00:00 for /var/spool/clientmqueue
```

Notice that the TTY fields are `?`, indicating that the sendmail processes are not associated with a terminal. This usually means that the process is running as daemon, which is the case with Sendmail. If you don't see output indicating that Sendmail is running, execute the following command to start Sendmail:

```
# service sendmail start
Starting sendmail:                [ OK ]
Starting sm-client:               [ OK ]
```

Another technique you can use is to telnet to localhost, port 25. If Sendmail is running, you'll see something resembling the following:

```
# telnet localhost 25
Trying 127.0.0.1...
Connected to localhost.localdomain (127.0.0.1).
Escape character is '^]'.
220 bookbeast.example.com ESMTP Sendmail 8.13.1/8.13.1; Sat, 12 Mar 2005 22:51:
43 -0500
QUIT
```

Type `QUIT` to exit the session. This output means that Sendmail is running and responding to incoming SMTP sessions. On the other hand, if the output looks like the following, Sendmail *isn't* running and you need to start it using the command shown earlier.

```
# telnet localhost 25
Trying 127.0.0.1...
telnet: connect to address 127.0.0.1: Connection refused
telnet: Unable to connect to remote host: Connection refused
```

Once you know that Sendmail is installed and running, you are ready to start configuring it.

Configuring Sendmail

Many system administrators think that Sendmail is difficult to configure. If you look at its configuration file, `/etc/mail/sendmail.cf`, this might seem to be the case. However, Fedora Core and RHEL install a default Sendmail configuration file that works for most sites.

If you need to edit the configuration file at all, you may need to make only a couple of minor changes. Here are the key lines a Red Hat system administrator might want to edit in `/etc/mail/sendmail.cf`. The line in question begins with the uppercase letters `DS` and occurs near line 100. The following example specifies the host named `mailbeast.example.com` as a mail relay host.

```
# "Smart" relay host (may be null)
DS
```

Change the line to define the name of the mail relay host. The *mail relay host* is the name of the computer that sends and receives mail on your network. There should not be *any* space between DS and the hostname.

```
# "Smart" relay host (may be null)
DSmailbeast.example.com
```

Fortunately, you do not have to be a Sendmail expert to perform most configuration chores. For all but the most unusual or extreme mail system configurations, all you need is a copy of the predefined configuration files in `/etc/mail`, `sendmail.mc`. The basic procedure is to modify `sendmail.mc` for your own needs, regenerate `/etc/mail/sendmail.cf` using the m4 macro processor, as explained in a moment, and then test your configuration. This method enables you to make incremental changes, minimizing the risk of major problems. Fedora Core and RHEL ship with a generic Sendmail configuration file (`/etc/mail/sendmail.cf`).

NOTE Be sure to install the `sendmail-cf` package if it is not already installed because it contains the m4 macro files needed for configuring Sendmail using the m4 method described in the next section.

The m4 Macro Processor

What is a macro? A *macro* is a symbolic name for a long string of characters, much like a keyboard macro is a shorthand way to type a long series of keystrokes. Sendmail rules are generated from the macros in Sendmail macro files. The location of the generic Sendmail macro file for Red Hat is `/etc/mail/sendmail.mc`. In addition, Sendmail's own macro files are stored in a set of subdirectories in `/usr/share/sendmail-cf`. The rules in the Sendmail macro file generate the default Sendmail configuration file, `/etc/mail/sendmail.cf`. The m4 utility is a macro processor that reads the macro file and generates the configuration file.

An example of a macro in `/etc/mail/sendmail.mc` is the `OSTYPE` macro that identifies the operating system. Remember that Sendmail runs on many different operating systems, not just UNIX and Linux. On a Linux system, if you look at Sendmail's macro file, you see the following line, which tells Sendmail which operating system it's running on so that Sendmail runs properly:

```
OSTYPE(`linux')dnl
```

On Linux, the `OSTYPE` macro comes predefined so that you don't need to worry about it. The `dn1` token introduces an m4-style comment, which causes m4 to stop processing input until it encounters the end of the line.

The good news, though, is that you needn't worry about the intricacies of m4. The `sendmail-cf` package installs a Makefile in `/etc/mail` that you can use to regenerate your `sendmail.cf` file. If you really want complete, technical information about the macro file and how it works, read the `/usr/share/sendmail.cf/README` file.

NOTE The default `sendmail` configuration does not listen to the network for incoming connections. Be sure to read the comment in `/etc/mail/sendmail.mc` for a description of setting this parameter.

Understanding and Managing the Mail Queue

Sometimes email messages don't go out immediately and instead remain in one of the originating MTA's outgoing mail queues. The reasons for the delay can vary. Perhaps your network is down. Maybe your intranet's connection to the Internet is sporadic. The recipient's MTA might be unavailable. Whatever the reason, users can continue to compose email with their MUAs. When they send their messages, Sendmail puts the message into the mail queue and keeps trying to send the message at intervals defined in the Sendmail configuration file or passed as a command-line parameter to the Sendmail daemon when it starts. You can find out what these intervals are by checking the initialization script that starts Sendmail.

The following listing shows the file `/etc/sysconfig/sendmail`.

```
DAEMON=yes
QUEUE=1h
```

The line beginning with `QUEUE` defines the retry interval to retry as one hour (1h). You can specify the interval in `h` (hours), `m` (minutes), or `s` (seconds). Recent versions of Fedora Core and RHEL set the variable `QUEUE` to 1h. Some distributions hard-code the interval right into the Sendmail invocation, for example, `sendmail -q1h`.

Setting Up Aliases to Make Life Easier

Mail aliases are useful for creating distribution lists and for making access to users more convenient. For example, if people have trouble spelling someone's name, you can create an alias with alternate spellings, so if someone misspells

the name, the mail still reaches the intended recipient. You can also alias a nonexistent user to a real user. For example, you could set up an alias, `bubba`, which redirects all mail intended for `bubba` to the real user `kurt`. The aliases file is usually `/etc/aliases`. The following example contains entries for the following:

- System aliases for `mailer-daemon` and `postmaster`, which are required by RFCs for proper mail system operation.
- Redirections for pseudoaccounts such as `lp`, `shutdown`, and `daemon`. Most of these are all aliased to `root` by default, but you can change them.
- User aliases, such as `bubba`.
- Distribution lists, such as `authors`.

```
# Basic system aliases -- these MUST be present.
mailer-daemon: postmaster
postmaster:    root
# General redirections for pseudo accounts.
daemon:        root
lp:            root
sync:          root
shutdown:      root
usenet:        news
ftpadm:        ftp
ftpadmin:      ftp
ftp-adm:       ftp
ftp-admin:     ftp

# trap decode to catch security attacks
decode:        root

# Person who should get root's mail
root:          kurt

#Users
bubba:         kurt

#Distribution lists
terry,kurt:    authors
```

To create an entry in the aliases file, use your favorite editor. Each entry consists of the username, followed by a colon, followed by some sort of white space, followed by the alias. Multiple usernames can be specified on the left

side of the colon to create a primitive distribution list. After you save the file, you must run the `newaliases` command to make the changes take effect. This step is necessary because Sendmail looks at the binary file `/etc/aliases.db` to read alias information. The `newaliases` command reads your aliases text file and updates the binary file.

Using Other Sendmail Files and Commands

Glancing through your Sendmail configuration file shows that Sendmail uses several files and directories. The following list describes some of them.

- **/usr/sbin/sendmail** — The sendmail daemon executable image
- **mailq** — Shows the contents of the mail queue:

```
# mailq
/var/spool/mqueue is empty
Total requests: 0
```

- **/var/spool/mqueue** — The directory that holds the mail queue
- **/var/spool/mail** — The directory that holds users' mail spools (the mailbox file), for example:

```
# ls -l /var/spool/mail/*
-rw-rw---- 1 terry mail      0 Jun 21 23:53 /var/spool/mail/terry
-rw-rw---- 1 kurt  mail    554 Mar 14 21:48 /var/spool/mail/kurt
-rw----- 1 root  root   6416 Jan 26 04:02 /var/spool/mail/root
```

TIP For security, be sure that all mailbox files are readable and writable only by their owners or the `mail` group.

- **/etc/mail/access** — Lists usernames, email addresses, domain names, and IP addresses not permitted to send mail to your system (Yes, this file is counterintuitively named.)
- **/etc/mail/relay-domains** — Lists hosts (by name, partial names, or IP addresses) that are permitted to relay email through your system
- **/etc/mail/local-host-names** — Lists other names for your system
- **/etc/mail/virtusertable** — Maps email addresses to usernames on the system

Using the Postfix Mail Server

After you read this section, you might decide that Sendmail is far too complex and baffling, or at least more complicated than Postfix. While Sendmail *might* surpass Postfix in terms of configurability and features, this is only true of corner cases, that is, of extremely obtuse or unusual mail system configurations. Postfix is used every day at sites that handle thousands and tens of thousands of messages per day, so Postfix probably provides all of the functionality you'll need with a fraction of the frustration and aggravation that accompanies learning arcane configuration hieroglyphics.

The best part is that Postfix is fully compatible with Sendmail at the command level. For example, the command to regenerate the Postfix alias database is `newaliases`, the same name that Sendmail uses; the primary Postfix daemon is named `sendmail`, just like Sendmail. The similarity is deliberate, for Postfix was designed to be a high-performance, easier-to-use replacement for Sendmail. A single example might illustrate why it is easier to configure and use. As you learned in the previous section, the (unintuitive) Sendmail syntax for defining a mail relay host is:

```
DSrelay.example.com
```

Postfix's syntax is eminently clearer:

```
relayhost = relay.example.com
```

If this admittedly simple example doesn't convince you that Postfix is easier to configure and use, consider this: it is Sendmail, not Postfix, that needs a meta-configuration-language (the m4 macros described earlier) to generate or modify configuration files.

Switching to Postfix

By default, Fedora Core and RHEL use Sendmail. Switching to Postfix is simple, but before doing so, stop Sendmail:

```
# service sendmail stop
```

You want to stop Sendmail before changing anything so that no incoming mail gets stuck in Sendmail's queues before it is delivered. Of course, you should make sure that Postfix is installed:

```
$ rpmquery postfix
postfix-2.2.2-2
```


Install Postfix before proceeding. Otherwise, to make Postfix your MTA, click Main Menu ⇨ Preferences ⇨ More Preferences ⇨ Mail Transport Agent Switcher or execute the command `system-switch-mail` at a command prompt. Either way, you should see the dialog box shown in Figure 21-1.

Click the Postfix radio button and then click OK to save your change and close the dialog box. After the change is applied, you will see the confirmation message shown in Figure 21-2.

Click OK to proceed. The Mail Transport Agent Switcher does most of the heavy lifting for you, so most of what you need to do is tweak and fine-tune the Postfix configuration, as described in the next section.

Configuring Postfix

Postfix's primary configuration file is `/etc/postfix/main.cf`. You will need to check or edit at least the following variables:

- The `mydomain` variable specifies your domain name:

```
mydomain = example.com
```

- The `myhostname` variable identifies the local machine's fully qualified domain name:

```
myhostname = coondog.example.com
```

- The `myorigin` variable identifies the domain name appended to unqualified addresses (that is, usernames without the `@example.com` goober attached):

```
myorigin = $mydomain
```

This causes all mail going out to have your domain name appended. Thus, if the value of `mydomain` is `possum_holler.com` and your username is `bubba`, then your outgoing mail will appear to come from `bubba@possum_holler.com`.

- The `mydestination` variable tells Postfix what addresses it should deliver locally. For a standalone workstation, which is a system that is connected directly to the Internet and that has some sort of domain name resolution running, you want mail to that machine and to `localhost` (and/or `localhost.$mydomain` and/or `localhost.localdomain`) delivered locally, so the following entry should suffice:

```
mydestination = $myhostname, localhost, localhost.$mydomain
```

Postfix supports a larger number of configuration variables than the four just listed, but these are the mandatory changes you have to make.



Figure 21-1 The Mail Transport Agent Switcher.

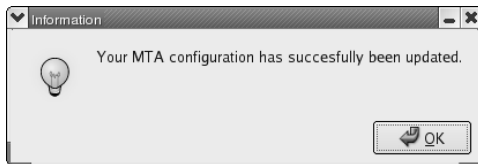


Figure 21-2 Successfully changing the MTA.

Create or modify `/etc/aliases`. At the very least, you need aliases for Postfix, `postmaster`, and `root` in order for mail sent to those addresses to get to a real person. Here are the contents of my initial `/etc/aliases` file:

```
postfix: root
postmaster: root
root: bubba
```

After creating or modifying the aliases file, regenerate the alias database using Postfix's `newaliases` command:

```
# /usr/sbin/newaliases
```

You are finally ready to start Postfix:

```
# service postfix start
Starting postfix: [ OK ]
```

Make sure that Postfix will start when you boot the system. This should be taken care of by the MTA switching tool, but it never hurts to double-check. You can use the `chkconfig` commands shown in the following example:

```
# chkconfig --levels 0123456 sendmail off
# chkconfig --levels 0123456 postfix off
# chkconfig --levels 2345 postfix on
```

Finally, modify your syslog configuration to handle Postfix log messages appropriately. Many system administrators, including the authors, prefer that mail log messages go to their own files to avoid cluttering up the primary system log. So, we use the following entries in `/etc/syslog.conf`, which controls the system log:

```
*.info;*.!warn;authpriv.none;cron.none;mail.none; -/var/log/messages
*.warn;authpriv.none;cron.none;mail.none;         -/var/log/syslog
mail.*;mail.!err                                  -/var/log/mail.log
mail.err                                           -/var/log/mail.err
```

The first two lines keep any mail-related messages from being logged to `/var/log/messages` and `/var/log/syslog`. The third line logs everything but errors to `/var/log/mail.log`. The last line drops all error messages from Postfix into `/var/log/mail.err`. The `-` character before each filename tells the system logging daemon, `syslogd`, to use asynchronous writes, which means that the logging daemon does not force log messages out to the specified file before returning control to the system. This measure helps Postfix run somewhat faster, especially on a heavily loaded system but can lead to data loss if the machine crashes before buffered data is flushed to disk.

Naturally, you have to restart `syslogd` to cause these changes to take effect:

```
# service syslog restart
```

At this point, you have a basic, functional Postfix installation. There is a great deal more customization that you *can* do and might *want* to do, but what has been covered here should get you started and offer some insight into the simplicity of Postfix installation and configuration.

Running Postfix behind a Firewall or Gateway

If the system on which you run Postfix is behind a firewall, uses a mail host, or otherwise lacks a direct or constant Internet connection, you probably want to define a relay host that handles your system's outbound email. In this case, Postfix will simply hand off locally generated email to the relay host, which must be configured to relay for you. For a system that sits on an internal network and that doesn't have a direct connection to the Internet, add the following entries to `/etc/postfix/main.cf`:

```
relayhost = mailhost.$mydomain
disable_dns_lookups = yes
```

`mailhost.$mydomain` (replace `mailhost` with the actual name of the relay host) handles actual mail delivery. If you don't run DNS on your internal network, the second line prevents Postfix's SMTP client from performing DNS lookups and instead causes Postfix to retrieve the IP address for the relay host from `/etc/hosts`, so make sure that `/etc/hosts` contains the fully qualified domain name, IP address, and alias (if one exists) for the relay host you specify.

TIP You can also specify the relay host's IP address in `/etc/postfix/main.cf` using the syntax:

```
relayhost = [192.168.0.1]
```

Notice that the IP address is enclosed in square brackets. The square brackets implicitly disable DNS lookups. That is, the square brackets imply `disable_dns_lookups = yes`.

If you make these (or other) changes to the Postfix configuration file, you have to tell Postfix about them. Use the following command to do so:

```
# service postfix reload
Reloading postfix: [ OK ]
```

The next section, "Running Postfix on a Mail Host," shows you how to create a mail host that handles incoming mail for the systems on your network.

Running Postfix on a Mail Host

At the end of the previous section, you configured Postfix to use a mail host, sometimes called a smart host, mail hub, or mail relay, for delivering outbound mail. In this section, you configure the mail host to process outbound mail for such client systems. This configuration assumes that the relay host, named `mailbeast` (just an example), is the sole point of entry and exit for all email traffic entering the network from the Internet and exiting the network from client systems.

As you did on the client systems, you need to set the following configuration variables on `mailbeast`:

- `$myhostname`
- `$mydomain`
- `$myorigin`
- `$mydestination`

In addition, `mailbeast` needs to be told for which systems it can relay mail. Doing so involves setting two additional configuration variables, `$mynetworks` and `$relay_domains`. `$mynetworks` defines a list of trusted SMTP clients, that is, the list of clients that Postfix will allow to relay mail. `$relay_domains` defines the destinations to which Postfix will relay mail.

Define `$mynetworks` using an explicit list of network/netmask patterns. Consider the following `$mynetworks` setting:

```
mynetworks = 192.168.0.0/24, 127.0.0.0/8
```

TIP If you have trouble deriving the appropriate netmask to use, remember the `ipcalc` tool introduced Chapter 12.

This directive states that any system with an IP address in the range 192.168.0.1 to 192.168.0.254 or in the loopback network can relay mail through the Postfix server. You need to use values that reflect your internal network, of course.

Where `$mynetworks` defines who is permitted to relay using the Postfix server, `$relay_domains` identifies to where email can be relayed. By default, Postfix relays mail to any address that matches `$mynetworks` and `$mydestination` (the default value of `$relay_domains` is `$mydestination`). To add relay destinations, specify a comma- or space-delimited list of hostnames or domains. For example, the following directive allows relays to `$mydestination`, the domain `example.com` (and any sub-domain of `example.com`), and the host `mailbeast.otherexample.com`:

```
relay_domains = $mydestination, example.com,  
                mailbeast.otherexample.com
```

Notice how the long line is continued using white space at the beginning of the next line. After making these changes, use the `reload` command shown earlier (`service postfix reload`).

Serving Email with POP3 and IMAP

The mail system configurations discussed so far assumed that all systems on your network run some sort of MTA. Obviously, this is an unwarranted assumption. For example, Windows systems used as desktop network clients ordinarily do not have an MTA of their own. Such systems require email access

using IMAP or POP3 (or Web-based mail, discussed in Chapter 24). This section shows you how to configure IMAP and POP3 servers. Worth noting is that you can provide both IMAP and POP3 services, but that clients usually need to use one or the other or chaos will ensue. Bear in mind also that IMAP, while more feature-rich than POP3, imposes a significantly higher disk space penalty on the server, especially if users decide to store all of their email on the server. POP3 is slimmer than IMAP, but heavy POP3 usage can dramatically bog down a mail server due to the overhead involved in clients polling the server for new mail.

Setting up an IMAP Server

The IMAP implementation configured in this section is the Dovecot IMAP server. As an extra bonus, the Dovecot IMAP server *also* speaks POP3. We've selected Dovecot for several reasons. First, it supports POP3 and IMAP, simplifying initial setup and ongoing maintenance. So, if you configure the IMAP server using the procedures described in this section, you get a POP3 server for free unless you specifically disable POP3 services. Second, Dovecot also supports POP3S and IMAPS (Secure POP3 and Secure IMAP, respectively), which wrap the authentication and data exchange processes in SSL-based encryption (using OpenSSL). Finally, Dovecot is also ready to run after the necessary packages have been installed, modulo the steps described in the following paragraphs.

First, make sure that the `dovecot` package is installed. The following `rpmquery` command shows you whether this package is installed. If not, install the `dovecot` package before proceeding:

```
# rpmquery dovecot
dovecot-0.99.14-4.fc4
```

The version number you see might be slightly different.

Configuring Dovecot

If the necessary packages are installed, configure the `dovecot` service to start when the system boots. If you don't intend to provide an IMAP server, you can disable the IMAP services as described shortly. Use the following commands to start `dovecot` at boot time:

```
# chkconfig --levels 0123456 dovecot off
# chkconfig --levels 345 dovecot on
```

Testing Cyrus

To test the server, connect to the POP3 server as a mortal user using `telnet`:

```
$ telnet localhost pop3
Trying 127.0.0.1...
Connected to localhost.localdomain (127.0.0.1).
Escape character is '^]'.
+OK dovecot ready.
quit
+OK Logging out
Connection closed by foreign host.
```

To close the connection, type **quit** (or **QUIT**) and press Enter. This example used `telnet` to connect to the POP3 port (port 110). If you see anything other than `+OK mumble` displayed, check your configuration. If you can connect to the POP3 server, you will be able to retrieve messages using the POP3 protocol.

Next up, connect to the IMAP server, again using `telnet`:

```
$ telnet localhost imap
Trying 127.0.0.1...
Connected to localhost.localdomain (127.0.0.1).
Escape character is '^]'.
* OK dovecot ready.
. logout
* BYE Logging out
. OK Logout completed.
Connection closed by foreign host.
```

To close the connection, type **. logout** (or **. LOGOUT**) and press Enter. This example used `telnet` to connect to the IMAP port (port 143). If you see anything other than `* OK mumble` displayed, check your configuration. At this point, your IMAP and POP servers are up and running and ready to service IMAP and POP clients.

Maintaining Email Security

Do you think you have nothing to hide? Maybe *you* don't, but email security is always a privacy issue even if you aren't mailing credit card numbers or corporate secrets. Using S/MIME (secure MIME) for security is only one of many steps to take to protect the integrity of your own and your users' email.

NOTE This section briefly covers some of the most common vulnerabilities that affect email security. For more information about email security, see the Sendmail Web site at <http://www.sendmail.org/> and the Postfix Web site at <http://www.postfix.org/>.

Protecting against Eavesdropping

Your mail message goes through more computers than just yours and your recipient's because of store-and-forward techniques. All a cracker has to do to snoop through your mail is use a packet sniffer program to intercept passing mail messages. A *packet sniffer* is intended to be a tool that a network administrator uses to record and analyze network traffic, but the bad guys use them too. Dozens of free packet sniffing programs are available on the Internet.

Using Encryption

Cryptography isn't just for secret agents. Many email products enable your messages to be encrypted (coded in a secret pattern) so that only you and your recipient can read them. Lotus Notes provides email encryption, for example. One common method is to sign your messages using digital signatures, which makes it possible for people to confirm that a message purporting to come from you did in fact come from you. Another typical approach, which can be used with digital signatures, is to encrypt email itself. Combining digital signatures with encryption protects both the confidentiality of your email and its authenticity. Fedora Core and RHEL ship with GNU Privacy Guard, or GPG, which provides a full suite of digital signature and encryption services.

Using a Firewall

If you receive mail from people outside your network, you should set up a firewall to protect your network. The *firewall* is a computer that prevents unauthorized data from reaching your network. For example, if you don't want anything from `ispy.com` to penetrate your net, put your net behind a firewall. The firewall blocks out all `ispy.com` messages. If you work on one computer dialed in to an ISP, you can still install a firewall. Several vendors provide personal firewalls, and some of them are free if you don't want a lot of bells and whistles.

Don't Get Bombed, Spammed, or Spoofed

Bombing happens when someone continually sends the same message to an email address either accidentally or maliciously. If you reside in the United States and you receive 200 or more copies of the same message from the same person, you can report the bomber to the FBI. The U.S. Federal Bureau of Investigation has a National Computer Crimes Squad in Washington, DC, telephone +1-202-325-9164.

Spamming is a variation of bombing. A spammer sends unsolicited email to many users (hundreds, thousands, and even tens of thousands). You easily can be an accidental spammer. If you choose your email's Reply All function, and you send a reply to a worldwide distribution list, you might be perceived by some of the recipients as a spammer.

Spoofing happens when someone sends you email from a fake address. If spoofing doesn't seem like it could be a major problem for you, consider this: you get email from a system administrator telling you to use a specific password for security reasons. Many people comply because the system administrator knows best. Imagine the consequences if a spoofer sends an email faking the system administrator's email address to all the users on a computer. All of a sudden, the spoofer knows everyone's passwords and has access to private and possibly sensitive or secret data. Spoofing is possible because plain SMTP does not have authentication capabilities. Without authentication features, SMTP can't be sure that incoming mail is really from the address it says it is. If your mail server enables connections to the SMTP port, anyone with a little knowledge of the internal workings of SMTP can connect to that port and send you email from a spoofed address. Besides connecting to the SMTP port of a site, a user can send spoofed email by modifying his or her Web browser interfaces.

TIP You can protect your data and configure your mail system to make mail fraud more difficult. If someone invades your mail system, you should report the intrusion to the Computer Emergency Response Team (CERT). You can find the reporting form on the Internet at `ftp://info.cert.org/pub/incident_reporting_form`.

Be Careful with SMTP

Use dedicated mail servers. First of all, keep the number of computers vulnerable to SMTP-based attacks to a minimum. Have only one or a few centralized email servers, depending on the size of your organization.

Allow only SMTP connections that come from outside your firewall to go to those few central email servers. This policy protects the other computers on

your network. If your site gets spammed, you have to clean up the central email servers, but the rest of your networked computers are okay.

If you use packet filtering, you need only configure your email servers. Packet filtering analyzes packets based on the source and destination addresses. The analysis decides whether to accept the packets and pass them through to your networks or to reject them as being unsafe. Firewalls often use packet-filtering techniques. The latest stable kernel, 2.6, has powerful built-in packet filtering capabilities.

Summary

This chapter explained the steps a message takes through MUAs, MTAs (Sendmail and Postfix), TCP/IP protocols, LDAs, mail-notification programs, mail queues, and mailboxes. Along the way you learned how to configure both the client and server sides of an email system.

Configuring FTP Services

IN THIS CHAPTER

- Introducing vsftpd
- Configuring vsftpd
- Advanced FTP Server Configuration

In today's pecking order of Internet services, FTP, the File Transfer Protocol, arguably places third behind email's enormous popularity and the Web's visual appeal. Despite its second-class status, FTP is a fundamental Internet service, one that almost every Internet user has used at one time or another. This chapter shows you how to install, configure, and maintain the Very Secure FTP daemon, vsftpd, the FTP server package that comes with Fedora Core and RHEL.

FTP itself is relatively straightforward and uncomplicated to install, configure, maintain, and monitor. For the lion's share of FTP installations, vsftpd works with few modifications. Minimal tweaks are necessary to customize the FTP server for your site. When problems arise, though, they are usually the result of unexpected interactions between the FTP server and various security measures. After you've read this chapter, you should be able to configure an FTP server with your eyes closed.

NOTE A complete Fedora Core or RHEL installation also installs `in.tftpd`, an FTP server implementing the Trivial File Transfer Protocol (TFTP). However, TFTP is used almost exclusively for PXE boot services and for booting diskless workstations, such as X terminals and slave nodes in clusters, or transferring files to other diskless devices, such as network routers and bridges, so it is not mentioned again in this chapter.

Introducing vsftpd

The default FTP server daemon on Fedora Core and RHEL systems is vsftpd, the Very Secure FTP Daemon, which has a project Web site at <http://vsftpd.beasts.org/>. Red Hat Software feels confident enough about vsftpd, in fact, to use it to power their own FTP site. So does the OpenBSD FTP site. So do a lot of other sites. vsftpd is extremely lightweight in that it makes sparing use of system resources and does not rely on system binaries for parts of its functionality. It can be tuned, to some degree, to use even fewer resources if need be. For more information about vsftpd, see the vsftpd Web site at <http://vsftpd.beasts.org/>.

TIP In this chapter, the text **vsftpd** refers to the name of the product. When **vsftpd** is used to refer to the actual daemon binary, it appears in monospaced text, for example, `vsftpd`.

To the standard FTP services defined in RFC 959, the core RFC (Request for Comment) that defines the FTP protocol, vsftpd offers the additional security features and usability enhancements listed here:

- Support for virtual IP configurations
- Support for so-called virtual users
- Can run as a standalone daemon or from `inetd` or `xinetd`
- Configurable on a per-user or per-IP basis
- Bandwidth throttling
- IPv6-ready

Unlike older versions of products formerly known as Red Hat Linux, you no longer have to install a special RPM to provide anonymous FTP services. A couple of tweaks to the vsftpd configuration file and you are set to go. One of the reasons a special RPM is no longer required is that vsftpd is self-contained — that is, it doesn't need access to system binaries, such as a statically linked `/bin/ls`, to operate. Rather, vsftpd provides internal implementations of commands other FTP daemons (such as the venerable `wu-ftp`) expect the host operating system to provide.

Although Fedora Core and RHEL prefer and install vsftpd, ProFTPD and NcFTPd deserve mention because they are widely used at busy FTP sites. ProFTPD (www.proftpd.org) is a free FTP server licensed under the GPL. Roughly modeled on the Apache Web server, ProFTPD was designed to be more configurable and more secure than vsftpd. ProFTPD was written from

scratch. Other Linux FTP servers, including vsftpd, evolved from the original BSD `ftpd` server. The following key features distinguish ProFTPD:

- Per-directory access configuration using `.ftpaccess` files, much like Apache's `.htaccess` file
- An anonymous FTP root directory unencumbered by required directory structures and system binaries
- Support for hidden files and directories
- Self-contained and does not need to use system binaries or libraries, reducing the likelihood of exploits that take advantage of external programs
- Runs as an unprivileged user in standalone mode, decreasing exposure to security attacks that attempt to exploit its root privileges

NcFTPD (<http://www.ncftp.com>) is a commercial FTP server that, like ProFTPD, was written from scratch, optimized for anonymous FTP service, and designed for high performance. Its primary architectural features are its self-described “no-forks” design — not spawning child processes to handle incoming connections and individual directory listings — and its independence from `inetd` and `xinetd`. It runs as a standalone server. It is *not* free software, but its features, security, and performance make it a popular FTP server.

Configuring vsftpd

Depending on the type of installation you selected, the installer, might or might not have installed vsftpd. To find out, execute the command `rpmquery vsftpd`. If the output resembles the following, vsftpd is installed:

```
# rpmquery vsftpd
vsftpd-2.0.1-5
```

If, on the other hand, you see this message, you must at least install the binary RPM before continuing with this chapter:

```
# rpmquery vsftpd
package vsftpd is not installed
```

If vsftpd is installed, configure it to start at boot time using the `chkconfig` command:

```
# chkconfig --levels 0123456 vsftpd off
# chkconfig --levels 345 vsftpd on
```

Alternatively, you can use the graphical Service Configuration tool. To do so, type `system-config-services` at a command prompt or select Main Menu ⇨ System Settings ⇨ Server Setting ⇨ Services. When you see the screen shown in Figure 22-1, scroll down to the `vsftpd` entry near the bottom of the services scroll box.

Click the check box next to `vsftpd` to enable it and then click Save to save your changes. Select File ⇨ Exit to close the Service Configuration Tool after saving your changes.

The stock `vsftpd` installation creates a basic functioning FTP server that works for users with their own login accounts on the system and for anonymous FTP, using either the `anonymous` or `ftp` login names. Just to be sure that everything is working, however, do a quick smoke test. If not already present (it should be), add the following line to the bottom of `/etc/vsftpd/vsftpd.conf`, the `vsftpd` configuration file:

```
listen=YES
```

This entry configures `vsftpd` to run as a standalone daemon. The case is important, so add the line as shown.

Start `vsftpd`:

```
# service vsftpd start
```

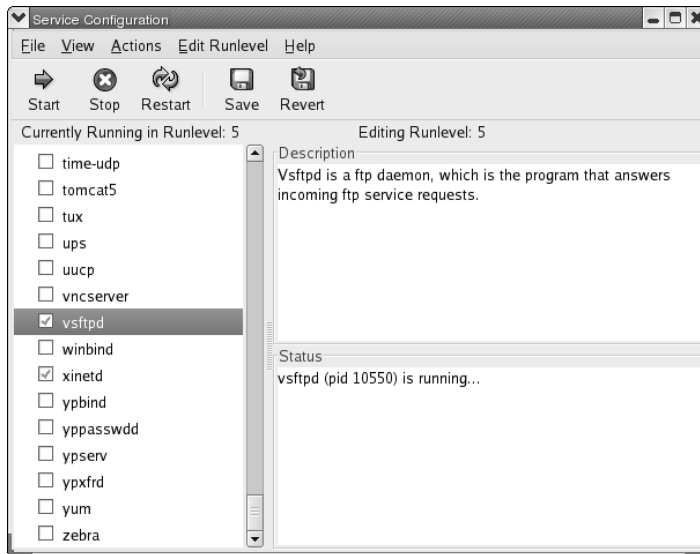


Figure 22-1 The Service Configuration tool.

Finally, try to log in as an anonymous user. You can use a login name of `ftp` or `anonymous`:

```
$ ftp localhost
Connected to localhost (127.0.0.1).
220 (vsFTPD 2.0.1)
Name (localhost:bubba): ftp
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls -a
227 Entering Passive Mode (127,0,0,1,100,97)
150 Here comes the directory listing.
drwxr-xr-x  3 0      0      16 Jan 22 14:17 .
drwxr-xr-x  3 0      0      16 Jan 22 14:17 ..
drwxr-xr-x  2 0      0      6 Oct 04 06:36 pub
226 Directory send OK.
ftp> close
221 Goodbye.
ftp> bye
```

If all has gone well, and it should have, you will be able to log in as the anonymous user, as shown in the preceding example. Of course, you haven't populated the FTP directories yet, so there's not much to see. In its default configuration, `vsftpd` uses `/var/ftp/pub` as the download directory, so if you want to add content to your FTP server with no server reconfiguration, populate `/var/ftp/pub`.

The default configuration is only a start, a base you should customize to enhance security and to fit your needs. In this section, you learn how to fine-tune the default FTP server configuration.

The first step in FTP server configuration is to become familiar with the configuration files that control the server's behavior. Table 22-1 lists and briefly describes `vsftpd`'s configuration files.

Table 22-1 vsftpd Configuration Files

FILE NAME	DESCRIPTION
<code>/etc/vsftpd/vsftpd.conf</code>	Controls the operation of FTP daemon, <code>vsftpd</code>
<code>/etc/vsftpd/ftpusers</code>	Lists the users <i>not</i> allowed to login via FTP
<code>/etc/vsftp/user_list</code>	By default, defines user permitted access via FTP

Configuring User Level FTP Access

The `/etc/vsftpd/ftpusers` file is the simplest to understand. It contains a list of user or account names, one per line, that are not allowed to log in using FTP. This file is used to increase security. For example, if a cracker somehow obtains the root password but (stupidly) tries to log in as root using FTP, the login attempt will fail. Notice that the filename is annoyingly counterintuitive: user accounts listed in this file are *not* permitted to log in to the system via FTP. In general, `/etc/vsftpd/ftpusers` is used to prevent privileged user accounts, such as root, from using FTP to obtain access to the system. The following code shows the default `/etc/vsftpd/ftpusers` file:

```
root
bin
daemon
adm
lp
sync
shutdown
halt
mail
news
uucp
operator
games
nobody
```

So, to prevent a user named bubba from using FTP to log in, or, rather, to prevent bubba from logging in to the system via FTP, add bubba to the end of `/etc/vsftpd/ftpusers`. In most cases, these default entries should be sufficient, but if you install a software package, such as a database package, that requires one or more special user accounts, consider adding such special accounts to `/etc/vsftpd/ftpusers` in order to maintain strict limits on how the FTP server can be accessed.

TIP The default vsftpd configuration as installed by Fedora Core and RHEL allows anonymous FTP access. If you choose *not* to enable anonymous FTP access, add the user `ftp` to `/etc/vsftpd/ftpusers` and set `anonymous_enable=NO` in `/etc/vsftpd/vsftpd.conf`.

The `/etc/vsftpd/user_list` file serves a purpose similar to `/etc/vsftpd/ftpusers`, limiting FTP access, but it is more flexible. If you compare the two files, though, you see that the users listed in `/etc/vsftpd/user_list` are the same ones listed in `/etc/vsftpd/ftpusers`. So, what

is the difference between the two files? `/etc/vsftpd/ftpusers` unconditionally denies access to the system via FTP; `/etc/vsftpd/user_list` can be used to deny or permit access, depending on the value of the `userlist_deny` directive in `/etc/vsftpd/vsftpd.conf`. If `userlist_deny` is set to `NO` (that is, `userlist_deny=NO`), then `vsftpd` allows FTP access only to the users listed in `/etc/vsftpd.user_list`. If `userlist_deny` is set to `YES` (that is, `userlist_deny=YES`), no user listed in `/etc/vsftpd/user_list` will not be permitted to login via FTP. Such users will not even be prompted for a password.

Configuring vsftpd Features

By far, the most important (and potentially the longest) `vsftpd` configuration file is `/etc/vsftpd/vsftpd.conf`. The configuration directives in this file enable you to exercise finely grained control over `vsftpd`'s behavior. The configuration file itself has a pleasantly simple format. Each line is either a comment, which begins with `#`, or a directive. Directives have the form `option=value`. Most of the configuration options are Boolean, so they are either on or off, or, rather, `YES` or `NO`. A second group of configuration options take numeric values, and a third, considerably smaller set of configuration options accept string values.

To organize the discussion of `/etc/vsftpd/vsftpd.conf`, we start with the default configuration file provided in Fedora Core and RHEL. It is shown in the following listing, with most of the comments removed to preserve space and to make it easier to read. Like many configuration files, lines that begin with the hash sign (`#`) denote comments that the program ignores.

```
anonymous_enable=YES
local_enable=YES
write_enable=YES
local_umask=022
#anon_upload_enable=YES
#anon_mkdir_write_enable=YES
dirmessage_enable=YES
xferlog_enable=YES
connect_from_port_20=YES
#chown_uploads=YES
#chown_username=whoever
#xferlog_file=/var/log/vsftpd.log
xferlog_std_format=YES
#idle_session_timeout=600
#data_connection_timeout=120
#nopriv_user=ftpsecure
#async_abor_enable=YES
```

```
#ascii_upload_enable=YES
#ascii_download_enable=YES
#ftpd_banner=Welcome to blah FTP service.
#deny_email_enable=YES
#banned_email_file=/etc/vsftpd.banned_emails
#chroot_list_enable=YES
#chroot_list_file=/etc/vsftpd.chroot_list
#ls_recurse_enable=YES
pam_service_name=vsftpd
userlist_enable=YES
listen=YES
tcp_wrappers=YES
```

The first configuration option, `anonymous_enable=YES`, allows anonymous FTP access. You can set this to `NO` if you do not want to enable anonymous FTP. The directive `local_enable=YES` allows local users (users with accounts on the system on which `vsftpd` is running) to access the system via FTP. Similarly, `write_enable=YES` enables all variations of the FTP commands that allow FTP users to modify the file system, such as `STOR` (the FTP `put` and `mput` commands for uploading files) and `DELE` (the FTP `del` command for deleting files). As a rule, it is unwise to permit FTP users to modify the file system, so if security is a concern, you might consider disabling write commands by setting `write_enable` to `NO`.

At first glance, it might seem reasonable to grant local users write access, or at least to be able to log in. However, the reason local users should *not* be allowed to log in is that FTP is a clear-text protocol; usernames and, more importantly, passwords, are transmitted in clear text. Anyone with a packet sniffer monitoring IP traffic to and from your FTP server's IP address can grab the authentication information and use it to compromise your system. Where FTP is concerned, one can argue that anonymous access is actually *less* of a security risk than normal, authenticated access because anonymous FTP does not require transmitting sensitive passwords over the wire. In fact, for local users, `sftp sftp`, part of the OpenSSH suite of programs, is vastly superior to plain vanilla FTP because it encapsulates FTP inside a secure, encrypted communication session (see "Using SFTP" at the end of this chapter).

The directives `anon_upload_enable=YES` and `anon_mkdir_write_enable=YES` control whether or not anonymous FTP users can upload files and create directories, respectively. These two directives are commented out, so anonymous uploads are disabled. Hopefully, it is obvious that allowing anonymous uploads is potentially dangerous. That said, should you decide to permit anonymous uploads, you will want to investigate additional `vsftpd` configuration directives that restrict this type of access. In particular, use the `chown_uploads=YES` directive to change the ownership of uploaded files to

the username specified by the `chown_username` directive. See the section titled “Enabling Anonymous Uploads” for more information about creating a more secure anonymous upload environment using `vsftpd`.

TIP For more secure FTP-like behavior, consider using the `sftp` (secure FTP) command or the very nice `lftp` FTP client. `sftp` is part of the OpenSSH suite of commands and implements a secure version of FTP. `lftp`, similarly, uses SSH over fish (that is, SSL using blowfish encryption), a very handy feature.

In the configuration file shown, the `chown_username` is `whoever`. We recommend changing this to `nobody` (that is, `chown_username=nobody`) because the user named `nobody` has special security semantics associated with it that reduce security risks. One of these semantic features is that the `nobody` user is not allowed to login via FTP. Another feature of the `nobody` user is that it usually does not have a login shell; in the default `/etc/passwd` file on Fedora Core and RHEL systems, `nobody`’s login shell is `/sbin/nologin`.

The various `xferlog` directives `xferlog_enable`, `xferlog_file`, and `xferlog_std_format`, control the location of the transfer log and the format of entries in this log file. If `xferlog_enable` is set to `YES`, file uploads (if permitted) and downloads are recorded in a log file, widely referred to using the shorthand expression *xferlog*. The `xferlog_file` directive specifies the name of the `xferlog` file. The default log file is `/var/log/vsftpd.log`, which should be suitable for most purposes. If you want to capture file transfers in a more mnemonically named file, you might use `xferlog_file=/var/log/xfer.log`, but this is only a suggestion.

If you want to change the log entry format from the standard format, set `xferlog_std_format=NO` and add the directive `log_ftp_protocol=YES`, which will cause all FTP protocol requests to be dumped into the log file. Again, the standard format should be fine. Logging FTP at the protocol level is mostly valuable when trying to isolate server or network problems or debugging `vsftpd` itself.

If you have ever wondered how FTP servers show you special messages when you `cd` into certain directories, the `dirmessage_enabled=YES` directive is how `vsftpd`, at least, accomplishes it. If this directive is set to `YES`, the first time a user enters a new directory, `vsftpd` displays the contents of a file named `.message`, if it exists. You can change the message file using the `message_file` directive. For example, `message_file=readme` sets the message file to `readme`. You can use the message file to display special notices specific to certain directories.

The `ftpd_banner` directive allows you to display a site-specific banner message when users connect to the server. For example, the directive `ftpd_banner=*** Welcome to the Possum Holler FTP Server ***` looks like the following when users first log in:

```
220 *** Welcome the Possum Holler FTP Server ***
User (192.168.0.1:(none)): ftp
331 Please specify the password.
Password:
230 Login successful. Have fun.
ftp>
```

Notice that the `ftpd_banner` directive is not embedded between double quotes (" "). This is because `vsftpd` would display the quotes as part of the banner.

If you want to deny access to your server based on the email address provided as part of an anonymous login, uncomment the `deny_email_enable=YES` and put the email addresses you want to deny access into a file named `/etc/vsftpd.banned_emails`. If you want to store the banned addresses in a different file, uncomment the `banned_email_file=/etc/vsftpd.banned_emails` directive and change the filename. Although this measure can be used as a security feature, it is extremely weak because it is trivial to change an email address and even more trivial to issue a bogus email address to an FTP server. Moreover, `vsftpd`, and Fedora Core and RHEL in general, offer much stronger and more secure methods for limiting or denying access.

The two `chroot` entries, `chroot_list_enable=YES` and `chroot_list_file=/etc/vsftpd.chroot_list`, affect the behavior of the `vsftpd` server when it runs in `chroot` mode. The first directive, if uncommented, causes `vsftpd` to execute a `chroot` to the home directory of local users when they log in. For this to work, however, the file `/etc/vsftpd.chroot_list` must contain a list of the users to whom this measure will be applied. `/etc/vsftpd.chroot_list` is the default. If you want to change it, uncomment `chroot_list_file=/etc/vsftpd.chroot_list` and change the filename appropriately. If you want to list users who should not be `chrooted`, set `chroot_local_user=YES` in `/etc/vsftpd/vsftpd.conf`. In this case, users listed in `chroot_list_file` will not be `chrooted` to their home directory.

Oh, what does *chroot* mean? *chroot* is short for *change root* and refers to changing a process's or a user's root directory so that it only sees a limited subset of the entire file system. For example, `chrooting` to `/home/bubba` turns `/home/bubba` into a process's root file system. Thereafter, `/home/bubba` is effectively the same as `/`, and the process or user can navigate no higher in the file system than `/home/bubba`. The only files and directories accessible are

those in `/home/bubba` and its subdirectories. More practically, chrooting a user or process is an access restriction technique and a way to limit damage to a prescribed area should the user or process go awry.

As explained earlier, `vsftpd` runs in standalone mode rather than from `xinetd` because the `listen` directive is set to `YES`. The default `vsftpd` configuration provided with in Fedora Core and RHEL does not provide an `xinetd` configuration file in `/etc/xinetd.d`, so you need to make sure that `listen=YES` is set *and* that `vsftpd` starts at boot time (provided, of course, you intend to provide FTP services). To enable `vsftpd` at boot time, execute the following commands (as root):

```
# /usr/sbin/chkconfig --levels 0123456 vsftpd off
# /usr/sbin/chkconfig --levels 345 vsftpd on
```

The first command disables the `vsftpd` service in all run levels. The second command reenables `vsftpd` in run levels 3, 4, and 5.

The `ls_recurse` directive enables the `-R` option to `vsftpd`'s built-in `ls` command, so executing the command `ls -R` during an FTP session performs a recursive `ls` of all files and subdirectories of the current directory. The directive `pam_service_name=vsftpd` defines the name that `vsftpd` uses to interact with PAM. If you want to use TCP wrappers, finally, make sure that `tcp_wrapper=YES` is enabled, which causes `vsftpd` to use the access control features available with TCP wrappers and to check `/etc/hosts.allow` and `/etc/hosts.deny` to evaluate which remote systems to allow access to.

Disabling Anonymous FTP

On the off chance you decided to disable anonymous FTP entirely, this is easily accomplished. The easiest way is to remove the `ftp` user from `/etc/passwd` and `/etc/group`:

```
# cp -p /etc/passwd /etc/passwd.ftp
# cp -p /etc/group /etc/group.ftp
# userdel -r ftp
userdel: /var/ftp not owned by ftp, not removing
# find / -user 50 | xargs rm -r
```

Ordinarily, `userdel`'s `-r` option removes files in `ftp`'s home directory (`/var/ftp`), but it doesn't work in this case because the `ftp` user doesn't own `/var/ftp`, `root` does. `userdel` also removes the `ftp` user from `/etc/group`, so you needn't execute the `groupdel` command. The `find` command locates all the files owned by the `ftp` user and deletes them. You have to use the numeric UID (50) instead of the username (`ftp`) because the username no

longer exists. You might not want to execute the command if you have populated the FTP server with files that you can't easily replace.

The problem with this method is that if you later decide to permit anonymous FTP, you have to recreate the `ftp` user and group because, as configured, `vsftpd` doesn't allow *any* FTP login if the user `ftp` is not present in the password file. That's why we made backup copies of `/etc/passwd` and `/etc/group` before executing `userdel`.

A more flexible approach is to add `ftp` to `/etc/vsftpd/user_list` and set `userlist_deny=YES` and `anonymous_enable=NO` in `/etc/vsftpd/vsftpd.conf`. It is *not* sufficient to comment out `anonymous_enable=YES`, because that will default to permitting anonymous FTP. This approach disables anonymous FTP while permitting regular FTP. However, if you use this method, remove any other users from `/etc/vsftpd/user_list` that you do want to be able to log in via FTP.

Advanced FTP Server Configuration

The information in the previous sections should enable you to get a basic, fully functioning FTP server up and running in 15 minutes, if you type *really* slowly. To be sure, there's nothing wrong with a plain vanilla FTP server. It will do what it is supposed to do, provide FTP access to designated files, reliably, quickly, and efficiently. Unfortunately, you will not be taking advantage of some of `vsftpd`'s best features. For example, perhaps you do not want to run `vsftpd` as a standalone daemon but prefer to run it via `xinetd`; maybe, despite the recommendation not to do so, you want to permit anonymous uploads; perchance you want or need to provide FTP services for guest users; mayhap you have decided to run `vsftpd` over SSL. This section describes how to enable each of these options.

Running vsftpd from xinetd

As remarked earlier in the chapter, the standard `vsftpd` installation does not install a configuration file for `xinetd`. However, all is not lost. `vsftpd`'s author provides a sample `xinetd` configuration file. Listing 22-1 shows this sample file, modified for Fedora Core and RHEL.

NOTE The preferred method for starting `vsftpd` is as a service, using the command `service vsftpd start`. It also works “out of the box” on any Fedora Core or RHEL system.

```
# default: on
# description:
#   The vsftpd FTP server serves FTP connections. It uses
#   normal, unencrypted usernames and passwords for authentication.
#   vsftpd is designed to be secure.
service ftp
{
    socket_type      = stream
    wait             = no
    user             = root
    server           = /usr/sbin/vsftpd
    server_args      = /etc/vsftpd/vsftpd.conf
    log_on_success   += DURATION USERID
    log_on_failure   += USERID
    nice             = 10
    disable          = no
}
```

Listing 22-1 An xinetd configuration file for vsftpd.

The modified file is available in this chapter's directory on the accompanying CD-ROM as `vsftpd.xinetd`. You can copy the file from the CD-ROM into `/etc/xinetd.d/vsftpd`, or you can create the file yourself. One last time, however, we do not recommend starting `vsftpd` from `xinetd` because the performance will be crummy. To express it more prosaically, starting `vsftpd` from `xinetd` is like teaching a pig to sing: it annoys the pig and you won't like the results.

Enabling Anonymous Uploads

Enabling anonymous uploads is also like teaching a pig to sing, but we're not going to beat that simile into the ground (but we *will* mangle a metaphor if we can get away with it). Seriously, anonymous uploads pose all sorts of security risks. The two that come to mind are someone uploading a virus or trojan to your server and having your FTP server become a *warez server*, that is, a transfer point for illegal or cracked copies of software, music, and/or movies. Becoming a warez server isn't so much a security risk as it is a freedom risk, as in, the authorities might put you in jail if they discover you're running a warez server. You get the point.

The obligatory security lecture notwithstanding, how do you enable anonymous uploads?

1. Edit `/etc/vsftpd/vsftpd.conf` and change or add the entries shown in Table 22-2.
2. Create a directory for anonymous uploads:

```
# mkdir /var/ftp/incoming
```

3. Change the permissions of `/var/ftp/incoming`:

```
# chmod 770 /var/ftp/incoming
```

```
# chmod g+s /var/ftp/incoming
```

The first `chmod` command give the user and group full access to `/var/ftp/incoming`. The second command turns on the set-group ID (colloquially known as *setgid* or *sgid*) bit on the directory, causing any file created in this directory to have its group ownership set to the group that owns the directory.

4. Make the `ftp` user the group owner of `/var/ftp/incoming`:

```
# chgrp ftp /var/ftp/incoming
```

5. Restart `vsftpd`:

```
# service vsftpd restart
```

These steps create a directory, `/var/ftp/incoming`, to which anonymous FTP users can upload files but from which no one can retrieve files. Nor can anyone see a directory listing of the upload directory during an FTP session.

Enabling Guest User FTP Accounts

What is a guest FTP user? Why would you want to use this feature? A *guest user*, referred to in `vsftpd`'s documentation as a *virtual user*, describes any nonanonymous login that uses a login name that does not exist as a real login account on the FTP server. The purpose of guest user accounts is to provide broader privileges to FTP users than anonymous FTP provides without giving local users FTP access. Recall the discussion of why local user FTP access is a dodgy proposition: it exposes local user's authentication information to packet sniffers because FTP is a clear-text protocol. Guest users provide a measure of security because if the guest account is somehow compromised, it has access to only the FTP server, not to the system as a whole. Guest users also serve as an administrative convenience because you can create a guest user that has access to certain files that are not available to anonymous users. You can create multiple guest users each with their own specific access rights and FTP environments, too, although the guest user configuration presented in this section describes only a single guest account.

Table 22-2 Configuration Directives for Anonymous Uploads

DIRECTIVE	DESCRIPTION
<code>anon_umask=077</code>	Sets the umask used for files created by the anonymous user to <code>077</code>
<code>anon_upload_enable=YES</code>	Permits anonymous uploads
<code>write_enable=YES</code>	Permits FTP write commands

To create a guest user account, use the following procedure.

1. Create a guest user account:

```
# useradd -d /var/ftp/rhlnsa3 -s /sbin/nologin authors
```

This command created the user `authors` with a home directory of `/var/ftp/rhlnsa3` and a login shell of `/sbin/nologin`, which disables local logins for that account.

2. Add some content to this directory:

```
# echo 'This is a test file.' > /var/ftp/rhlnsa3/test.file
# chown authors:authors /var/ftp/rhlnsa3/test.file
```

The first command creates simple text file named `/var/ftp/rhlnsa3/test.file`. The second command changes the user and group ownership to `authors`.

3. Decide who will be permitted to access the `authors` account and then create a text file that with pairs of lines that specify the login name and password.

```
bubba
grits
marysue
greens
```

In this example, the users are named `bubba` and `marysue` and their passwords are `grits` and `greens`, respectively. They are stored in a file named `virtusers.txt`.

4. Create a Berkeley database file from the text file you created in Step 3. Use the following command:

```
# db_load -T -t hash -f virtusers.txt /etc/vsftpd/vsftpd_login.db
```

This command creates a database file with the contents of `virtusers.txt` and stores it in `/etc/vsftpd/vsftpd_login.db`. For additional security, change the permissions on the generated file so that only root owns it:

```
# chmod 600 /etc/vsftpd/vsftpd_login.db
```

5. Create a PAM (Pluggable Authentication Module) file in `/etc/pam.d` that contains the following entries:

```
auth required /lib/security/pam_userdb.so db=/etc/vsftpd/vsftpd_login
account required /lib/security/pam_userdb.so db=/etc/vsftpd/vsftpd_
login
```

You can use the `vsftpd.pam` file in this chapter's code directory on the included CD-ROM. These directives tell the PAM user database module to use the `vsftpd_login.db` file (the `.db` extension is assumed) for FTP logins.

6. Edit `/etc/vsftpd/vsftpd.conf`, adding the following entries or editing existing ones to match:

```
pam_service_name=ftp
guest_enable=YES
guest_username=authors
local_enable=YES
write_enable=NO
anon_world_readable_only=NO
chroot_local_users =YES
```

`pam_service_name=ftp` tells `vsftpd` to use the PAM file you created in Step 5. `guest_enable=YES` enables guest user functionality. `guest_username=authors` maps all guest logins to the authors account created in Step 1. `local_enable=YES` permits local users to login, which is required for guest users. The `anon_world_readable_only=NO` directive makes the contents of the `/var/ftp/rhlnsa3` directory readable by guest users. Finally, `chown_local_users=YES` causes `vsftpd` to `chroot` the authors user to `/var/ftp/rhlnsa3`.

7. Start, or restart, the `vsftpd` service:

```
# service vsftpd restart
```

8. Test it out, using one of the user/password combinations you created in Step 3:

```
$ ftp localhost
Connected to localhost.localdomain.
220 (vsFTPD 2.0.1)
530 Please login with USER and PASS.
Name (localhost:bubba): marysue
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
```

```

ftp> ls
227 Entering Passive Mode (127,0,0,1,253,85)
150 Here comes the directory listing.
-rw-r--r--    1 503      504      8192 Mar 20 15:44 ch22.doc
226 Directory send OK.
ftp> quit

```

The password typed at the prompt was `greens`. For the record, the user `marysue` does not have a local login account; she only exists in `vsftpd`'s guest user database (`/etc/vsftpd/vsftpd_login.db`).

That's it; guest user access is configured. For more information about guest users, including an example of creating multiple guest user accounts, see the `vsftpd` documentation, specifically, the configuration files in `/usr/share/doc/vsftpd-2.0.3/EXAMPLE/VIRTUAL_USERS` and `/usr/share/doc/vsftpd-2.0.3/EXAMPLE/VIRTUAL_USERS_2` (the version number might have changed by the time you read this).

Running vsftpd over SSL

`Vsftpd`'s Secure Sockets Layer (SSL) support convincingly answers one of the primary criticisms of FTP, passing authentication information in clear text. In fact, `vsftpd` can use SSL to encrypt FTP's control channel, over which authentication information is passed, and FTP's data channel, over which file transfers occur. To use SSL, with `vsftpd`, you need to set at least the `ssl_enable=YES` in `/etc/vsftpd/vsftpd.conf`. If you want to fine-tune `vsftpd`'s SSL-related behavior, become familiar with `vsftpd`'s SSL-related configuration directives, listed in Table 22-3.

Table 22-3 SSL-Related Configuration Directives

DIRECTIVE	DESCRIPTION
<code>allow_anon_ssl=YES</code>	Permits anonymous users to use SSL
<code>dsa_cert_file=path</code>	Specifies the location of the DSA certification file (optional)
<code>force_local_data_ssl=YES</code>	Forces local user's FTP sessions to use SSL on the data connection
<code>force_local_logins_ssl=YES</code>	Forces local user's FTP sessions to use SSL for the login exchange
<code>rsa_cert_file=path</code>	Specifies the location of the RSA certificate file (default is <code>/usr/share/ssl/certs/vsftpd.pem</code>)

(continued)

Table 22-3 (continued)

DIRECTIVE	DESCRIPTION
ssl_ciphers=DES-CBC3-SHA	Specifies the SSL ciphers vsftpd will accept
ssl_enable=YES	Enables vsftpd's SSL support (required for all other <code>ssl_*</code> directives)
ssl_sslv2=YES	Enables vsftpd's SSL version 2 protocol support
ssl_sslv3=YES	Enables vsftpd's SSL version 3 protocol support
ssl_tlsv1=YES	Enables vsftpd's SSL TLS version 1 protocol support

1. Add the following entries to `/etc/vsftpd/vsftpd.conf`:

```
ssl_enable=YES
allow_anon_ssl=YES
force_local_data_ssl=YES
force_local_logins_ssl=YES
ssl_tlsv1=YES
```

2. Create a self-signed RSA certificate file:

```
# cd /usr/share/ssl/certs
# make vsftpd.pem
umask 77 ; \
PEM1=`/bin/mktemp /tmp/openssl.XXXXXX` ; \
PEM2=`/bin/mktemp /tmp/openssl.XXXXXX` ; \
/usr/bin/openssl req -newkey rsa:1024 -keyout $PEM1 -nodes -x509 -
days 365 -out
$PEM2 ; \
cat $PEM1 > vsftpd.pem ; \
echo ""      >> vsftpd.pem ; \
cat $PEM2 >> vsftpd.pem ; \
rm -f $PEM1 $PEM2
Generating a 1024 bit RSA private key
.....+++++
...+++++
writing new private key to '/tmp/openssl.yYS512'
-----
You are about to be asked to enter information that will be
incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or
a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
```

```

Country Name (2 letter code) [GB]:US
State or Province Name (full name) [Berkshire]:Allegheny
Locality Name (eg, city) [Newbury]:Pittsburgh
Organization Name (eg, company) [My Company Ltd]:KurtWerks
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:localhost
Email Address []:root@localhost

```

3. Start or restart the `vsftpd` service:

```
# service vsftpd start
```

Now you're ready to test the connection. The trick here is to test the connection using an FTP client that supports SSL. Not many do, certainly not the command line `ftp` program that is part of Fedora Core and RHEL. However, the `lftp` program does support SSL, so you can use it to verify that your SSL configuration is correct.

Using SFTP

As an alternative to configuring `vsftpd` to play nicely with SSL, you can use `sftp-server`, a program that is part of the OpenSSH (Secure Shell) suite of secure client and server programs. `sftp-server` implements the server-side portion of the FTP protocol. You do not invoke it directly; the SSH daemon, `sshd`, does so when it receives an incoming FTP request. Happily, you have nothing to configure for secure FTP. Well, you need to have the OpenSSH-related packages installed, but they are part of the standard Fedora Core 3 and RHEL installation. The following `rpmquery` command should remove all doubt:

```
# rpmquery openssl,{-clients,askpass,server}
openssl-4.0p1-2
openssl-clients-4.0p1-2
openssl-askpass-4.0p1-2
openssl-server-4.0p1-2

```

If these packages are not installed, install them before proceeding. If they are installed, make sure the following line appears in `/etc/ssh/sshd_config`:

```
Subsystem      sftp          /usr/libexec/openssh/sftp-server
```

This directive tells `sshd` to execute the program `/usr/libexec/openssh/sftp-server` to service the SFTP subsystem. Again, this entry should be part of the stock installation, but if it isn't, add it to the configuration file and then restart the SSH daemon using the following command:

```
# service sshd restart
```

From the client's perspective, very little changes. The client command to execute is `sftp` rather than `ftp`, and the set of supported commands is more limited than it is standard FTP commands. One important difference that between clear-text FTP and secure FTP is that `sftp` does not support anonymous FTP; users will always be prompted to provide a password unless they have set up SSH keys. However, this should not be a problem because you can configure `vsftpd` to provide anonymous FTP and then use OpenSSH to provide secure, authenticated FTP service for users that have valid accounts on the system. The two services can exist side by side because `sftp` uses port 115 and FTP uses ports 25.

Summary

This chapter discussed providing FTP services on your Fedora Core or RHEL system. `vsftpd`, the preferred FTP server for standard FTP, can be configured for both anonymous and local user access. To eliminate the security exposures of local user FTP access, you can also configure `vsftpd` to enable guest accounts, which provide some of the benefits of local user access without the security risks. In today's Internet, SSL-based encryption serves as the standard for security-conscious system administrators, and `vsftpd` supports a minimal level of SSL encryption; it is up to FTP users to find FTP *clients* that support SSL. As an alternative to `vsftpd` and SSL, you can simply use the secure FTP server, `sftp-server`, which is a standard part of the OpenSSH. It is easier to configure than `vsftpd` is with SSL because OpenSSH's `sftp` server is already configured.

Configuring a Web Server

IN THIS CHAPTER

- Introducing Apache
- Configuring Apache
- Implementing SSI
- Enabling CGI
- Enabling PHP
- Creating a Secure Server with SSL

The Apache Web server is the most popular Web server on the planet. Individuals and organizations use Linux primarily to create an inexpensive and stable Web server. So it shouldn't surprise you to find Apache running on a Linux server. This chapter shows you how to configure and maintain the Apache Web server on your Fedora Core or RHEL system. In addition to learning Apache's basic care and feeding, you'll also learn how to configure Apache to support server-side includes, CGI, and PHP. The chapter ends with a tutorial on creating a secure Web server using SSL.

Introducing Apache

This section highlights the history of the Apache Web server, describes its key features, and provides pointers to additional Apache resources that you may find useful.

The Apache Web server's origins are the National Center for Supercomputing Applications (NCSA) HTTP server (referred to as *httpd* in this chapter). NCSA's *httpd* was the most popular of the early HTTP servers, and its source code was in the public domain. After NCSA's active development of their

httpd effectively stopped (late 1994 and early 1995), a small group of Web administrators who had modified the source code to address their own site-specific needs or to patch various bugs gathered together to coordinate their activities and merge their code changes into a single code tree.

In April 1995, this loosely organized group of Web masters, the original Apache Group, released the first official version of Apache intended for public consumption, Apache 0.6.2. Despite its known warts, the Apache server was instantly popular. However, even as the development team continued to stabilize the existing code base, add new features, and generate documentation, other members undertook a fundamental redesign that rather quickly (August 1995) resulted in Apache 0.8.8. This first (beta) release based on the redesigned architecture still bears a strong resemblance to the Apache available today. Another round of code base stabilization, bug fixes, feature enhancements, and documentation updates concluded on November 30, 1995. The next day, December 1, 1995, the Apache Group released Apache 1.0, the version that cemented Apache's status as the Internet's number 1 HTTP server.

Although the actual numbers fluctuate and overall growth of Apache's usage (and of the Web servers in general) has begun to flatten out, periodic Internet-wide surveys conducted by Netcraft (www.netcraft.com/survey) consistently demonstrate that even 10 years after its first releases, Apache continues to be the most widely used Web server, surpassing all other Web servers combined.

CROSS-REFERENCE For more information about Apache's history, read

"About Apache" at http://httpd.apache.org/ABOUT_APACHE.html.

Apache Features

The complete list of Apache's features easily runs to three or four pages, far too many and far too tedious to recite here. In fact, many of its "features" are downright mundane because it performs the same function as any other Web server: sending some data down the wire in response to a properly formatted request from a Web client. These features are more properly called "functionality" because Apache is a standards-driven program providing mandated functionality. That is, many of its capabilities are the result of its full compliance with version 1.1 of the HTTP protocol. Apache implements all the required functionality described in RFC2616, the document that defines the current Web server protocol, HTTP/1.1 (HyperText Transfer Protocol version 1.1). Of course, the manner in which Apache implements mandated features is anything but mundane.

Apache's standout qualities are its speed, configurability, stability, and rich feature set. Most benchmark studies have shown Apache to be faster than

WHENCE THE NAME “APACHE”?

Here is the answer to the most frequently asked question about Apache, taken directly from the Apache FAQ

(<http://httpd.apache.org/docs/misc/FAQ.html#name>):

3. Why the name “Apache”?

The name “Apache” was chosen from respect for the Native American Indian tribe of Apache (Ind?), well-known for their superior skills in warfare strategy and their inexhaustible endurance. For more information on the Apache Nation, we suggest searching Google, Northernlight, or AllTheWeb.

Secondarily, and more popularly (though incorrectly) accepted, it’s a considered cute name which stuck. Apache is “A PATCHy server.” It was based on some existing code and a series of “patch files.”

This isn’t necessarily the case. The answer to this question has been the subject of considerable revisionism. Sometime between October 27, 2002, and January 24, 2003, the answer to this FAQ assumed the current form. On October 27, 2002, however, it read as follows:

3. Why the name “Apache”?

A cute name which stuck. Apache is “A PATCHy server.” It was based on some existing code and a series of “patch files.”

For many developers it is also a reverent connotation to the Native American Indian tribe of Apache, well-known for their superior skills in warfare strategy and inexhaustible endurance. For more information on the Apache Nation, we suggest searching Google, Northernlight, or AllTheWeb (<http://web.archive.org/web/20021027211348/http://httpd.apache.org/docs/misc/FAQ.html>).

This earlier form of the FAQ had been largely unchanged in form and content since at least late 2000. For example, see <http://web.archive.org/web/20001109083500/http://httpd.apache.org/docs/misc/FAQ.html#name>.

many other Web servers, including commercial servers. Apache is also both easy to configure and easy to reconfigure. Its configuration information resides in plain-text files and uses simple English-language directives. Reconfiguring a running server is a simple matter of changing the appropriate configuration directive and reloading the configuration file.

Few seriously question Apache’s stability, even among its competitors. Sites receiving millions of hits each day report no problems. Moreover, while no software product of any complexity can be considered bug-free, Apache is beset with fewer (known) bugs than other Web servers, particularly closed source servers. Many factors contribute to Apache’s stability, but the two most important are eyeballs and axle grease.

I hear you saying to yourself, “Huh? Eyeballs and axle grease? What is he talking about?” Read on for the answer:

- More eyeballs means fewer bugs. Apache’s source code is freely available, so hundreds of developers have looked at it, found bugs, fixed them, and submitted their fixes for inclusion in the code base. Similarly, Apache’s enormous user base means that virtually every segment of its code has been tested in real-world environments, uncovering most major bugs and many minor ones.
- The squeaky wheel gets the grease. Combine Apache’s widespread usage with Web site administrators who quickly complain about server bugs and insist on the absolutely vital requirement to maintain a secure Web server, and you can easily understand why the Apache team consistently and frequently releases patches for confirmed bugs and security holes, often within just a few hours of their discovery.

The balance of Apache’s features is the result of developer and user input. Apache is an open-source software project (indeed, Apache is a poster child of successful open-source software projects), so anyone can contribute code for inclusion in the server, although whether such code is accepted is up to members of the core Apache team. User feedback drives Apache’s development and defines its feature set. A very short list of such features includes, in no particular order:

- Apache is easily extensible using Dynamic Shared Objects (DSOs), more commonly known as modules. Modules extend Apache’s capabilities and new features without requiring recompilation because they can be loaded and unloaded at runtime, just as shared program libraries can be dynamically loaded and unloaded at runtime.
- Apache has a rich set of access controls and gives Web site administrators considerable flexibility in choosing authentication mechanisms. You can use a simple text file; a binary database format that supports very large numbers of users without becoming bogged down executing authentication requests; third-party databases such as MySQL, PostgreSQL, or Oracle; and even sitewide authentication methods such as LDAP.
- Apache supports virtual hosts, also known as multihomed servers, which enables a single machine to provide Web services for multiple domains or IP addresses (or hostnames).
- Apache enables administrators to define multiple directory index files, the default page to display when a Web client requests a directory URL. So, for example, the server can return `index.html`, `index.htm`, `index.php`, or execute a script named `index.cgi` when a client requests a directory URL, depending on what Apache finds in the requested directory.

- Another boon for Web server administrators is Apache's rich support for server logging. You can define custom log file formats and control the level of detail contained in each log entry. Apache can send log file output to named pipes (FIFOs) on systems that support named pipes, primarily Linux, UNIX, and similarly designed operating systems. This feature enables any arbitrary log manipulation that can be accomplished using a named pipe. In fact, Apache can be configured to generate a unique identifier that distinguishes one hit from every other hit, although there are some restrictions that apply.
- Within limits, Apache automatically adjusts to the capabilities of connected Web clients, one instance of a more general process called content negotiation. If a Web client incompletely or improperly implements HTTP standards, is broken in a way that Apache can determine, or does not support a given HTML specification (or, at least, the specification Apache supports), Apache sends Web pages modified to give the best representation of the requested information based on what the client can process. For more information about content negotiation, see the sidebar titled "What Is Content Negotiation?"

You'll learn more about these and other Apache features as you read the chapter.

WHAT IS CONTENT NEGOTIATION?

Content negotiation refers to the technique Web clients and servers use to select how to present a resource, such as a document, that is available in several different formats. For example, suppose that a Web page is available in different languages. One way to select the proper language is to give the user an index page from which she chooses the desired language. Content negotiation enables the server to choose the preferred language automatically based on information a Web browser sends, indicating what representations it prefers. For example, a browser could indicate that it would like to see information in French, if possible, or else in English if French is not available. Browsers indicate their preferences by transmitting specific data in each HTTP request's header. To request only French representations, the browser would send:

```
Accept-Language: fr
```

The next request shows a request that accepts both French and English, but prefers French:

```
Accept-Language: fr; q=1.0, en; q=0.5
```

These preferences apply only when there is a choice of representations and when the choices vary by language.

Changes in Apache 2

When Apache 2 was released, it included some fundamental architectural changes. One of the most far-reaching changes made in the Apache 2 affected how `httpd` servers are created. Apache 1.x followed a standard master/child server creation model. In the 1.x model, single master server process spawned a number of child server processes to perform the actual work of serving Web pages. As HTTP traffic increased, the master server would spawn additional child server processes to meet the demand, up to a limit set by the `MaxClients` directive. As HTTP traffic decreased, the master server would kill some of the child server processes so that the number of idle child `httpd` server processes stayed between the values set by `MinSpareServers` and `MaxSpareServers` directives.

While this model worked well, it imposed significant overhead and limited the scalability of Web servers. The overhead resulted from the time involved in and system resources consumed by creating, maintaining, and killing large numbers of server processes. The scalability issue resulted from having to recompile Apache to override the hard-coded limit of 256 clients (for the `MaxClients` directive).

Apache 2 redesigned the server generation process, replacing the monotonic process model with multiprocessing modules, or MPMs. Each MPM implements a different server control and process generation method, giving system administrators greater flexibility in deciding how Apache will create and destroy child server processes. Some MPMs exist only to take advantage of the underlying operating system's particularities. Examples of OS-specific MPMs include `beos`, `mpm_netware`, `mpm_os2`, and `mpm_winnt`, for BeOS, Netware, OS/2, and Windows NT, respectively. You cannot use these MPMs on Linux systems, so we have nothing more to say about them. The other MPMs implement various approaches to process creation and control. These MPMs include:

- **perchild** — Provides a hybrid server that is both a multiprocess and multithreaded server, meaning that the Apache master server starts multiple child server processes, and each child server process can spawn a number of server threads. The advantage of this model is that threads are much faster and consume very few resources compared to full-fledged processes. An additional feature makes it possible to assign different user IDs to the child processes.
- **prefork** — Provides an unthreaded, process-based server that resembles the server creation model from Apache 1.3x in which the master server maintains a pool of idle spare servers that are started before they are needed (that is, are preforked).

- **worker** — Provides a hybrid server that is both multiprocess and multithreaded. Like the `perchild` MPM, the `worker` MPM starts multiple child server processes, and each child server process can spawn a number of server threads. The difference between the `worker` and `perchild` MPMs is that the `worker` MPM cannot assign child processes to multiple user IDs.

NOTE There are two other MPMs, `leader` and `threadpool`. They are experimental variations of the `worker` MPM. Because they are experimental, they won't be discussed further. For more information about the Apache MPMs, including the `leader` and `threadpool` MPMs, see the Apache Web site at <http://httpd.apache.org/docs-2.0/mod>.

Why does this matter to you? The effect of certain serverwide configuration directives (described in the section “Configuring Apache”) is very different, depending on the MPM in use.

Other notable changes in Apache 2 include the following:

- The build system was rewritten to use the standard GNU build tools, namely, `autoconf` and `libtool`.
- A new API for modules removes most requirements for loading modules in a certain order.
- Apache 2 supports IPv6 on those systems that have IPv6. Rather, if the Apache runtime supports IPv6 on a given OS, Apache 2 listens on IPv6 ports for HTTP connections.
- Apache's regular expression support has been replaced using Perl-Compatible Regular Expressions (PCRE).
- HTTP error messages generated by the server can be displayed in multiple languages using server-side includes (SSI).
- Filters to modify the `httpd` input or output stream can be implemented as modules, significantly improving the performance of those filters.
- The configuration language has been simplified and rationalized. For example, confusing directives such as `Port` and `BindAddress` have been replaced by `Listen`.

How Web Servers Work

To understand Apache, its configuration, and how to fine-tune it for your own environment, you should understand how Web servers work in general. Otherwise, lacking this context, Apache's behavior and configuration might

seem arbitrary. Figure 23-1 shows the general process that takes place when a Web browser requests a page and the Apache Web server responds.

This simplified illustration disregards the browser cache, content accelerators such as Inktomi and Akamai, and the existence of proxy servers between the user's browser and the Apache Web server. The Web client (a browser in this case) first performs a DNS lookup on the server name specified in the URL, obtains the IP address of the server, and then connects to port 80 at that IP address (or another port if the server is not using the default HTTP port). If you specify an IP address directly, the DNS lookup doesn't occur. When the connection is established, the client sends an HTTP GET request for the document in the URL, which could be, among other possibilities, a specific HTML document, an image, or a script.

After the server receives the request, it translates the document URL into a filename on the local system. For example, the document URL `http://www.example.com/news.html` might become `/var/www/html/news.html`. Next, Apache evaluates whether the requested document is subject to some sort of access control. If no access control is required, Apache satisfies the request as described in the next paragraph. If access control is in effect, Apache requests a username and password from the client or rejects the request outright, depending on the type of access control in place.

If the requested URL specifies a directory (that is, the URL ends in `/`) rather than a specific document, Apache looks for the directory index page, `index.html` by default, and returns that document to the client. If the directory index page does not exist, Apache might send a directory listing in HTML format back to the client or send an error message, depending on how the server is configured. The document can also be a specially written script, a Common Gateway Interface (CGI) script. In this case, Apache executes the script, if permitted to do so, and sends the results back to the client. Finally, after Apache has transmitted the requested document and the client receives it, the client closes the connection and Apache writes an entry in one or more log files describing the request in varying levels of detail.

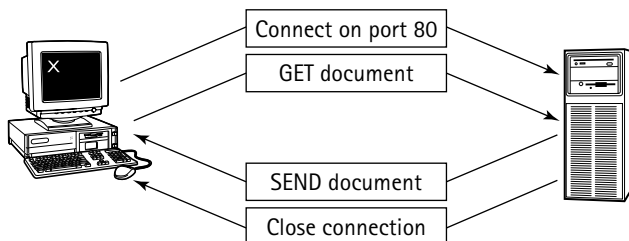


Figure 23-1 Apache transmits a document according to a client's request.

Depending on how the page is written and what it contains, additional processing takes place during the transfer. For example, embedded scripts or Java applets are transferred to and execute on the client side of the connection; server-side includes (discussed in the section titled “Implementing SSI”), however, are processed on the server side, as are CGI scripts, database access, and so forth.

Configuring Apache

Configuring Apache, especially for new or inexperienced administrators, seems a daunting task at first glance. The terminology is confusing, the concepts unfamiliar, and the configuration file intimidating. Moreover, there are a lot of configuration directives. The happy news is that many of Apache’s configuration options cover corner cases you might never encounter (but, if you do, you’ll be glad to have them). It’s the old 80/20 rule in reverse: 20 percent of Apache’s configuration options do 80 percent of the work. This section defines the terms, explains the concepts, and removes or at least reduces the intimidation. If you read carefully and eat your spinach, by the time you reach the end of the section, you will see that Apache is easy to configure, in large part because its developers and maintainers have taken the trouble to define sensible defaults for most configuration items. It also helps that Apache has a pleasantly complete documentation set, thanks to the efforts of the developers and of a dedicated user community.

Using the default configuration as a starting point is the easiest way to learn how to configure Apache. You will learn about groups of related Apache configuration directives and analyze the corresponding entries in the default configuration provided in Fedora Core and RHEL. Finally, after evaluating a variety of changes and additions to the default configuration, you will learn how to use a GUI, the HTTP Configuration tool (`system-config-httpd` for CLI addicts) to maintain the Web server configuration.

Depending on the installation profile you chose when you installed Fedora Core or RHEL, Apache might or might not be installed. To find out, use the `rpmquery` command shown in the following example. If the output is similar (the version numbers might be slightly different by the time you read this book), Apache is installed:

```
$ rpmquery httpd{,-{manual,devel }}
httpd-2.0.54-7
httpd-manual-2.0.54-7
httpd-devel-2.0.54-7
```

If Apache is installed, you’re ready to go. Otherwise, install the packages shown in the listing before proceeding.

Apache's Startup Process

When Apache starts, either at boot time or afterward, a single master process starts. The master server reads and processes `/etc/httpd/conf/httpd.conf`. The default location for this file is compiled into Apache but can be overridden using the `-f` option discussed later in the chapter. Apache also reads `/etc/mime.types` to configure the MIME subsystem, which maps file-name extensions to content types. In current versions of Apache, `httpd.conf` is the primary configuration file.

NOTE Pre 2.x versions of Apache included two additional configuration files, `srn.conf` and `access.conf`. These files existed for backward compatibility with earlier versions of Apache and the NCSA's `httpd` daemon. Support for these auxilliary files was removed from Apache 2.0. All configuration directives can and should be placed in `httpd.conf` or included from other files specified using the `Include` directive.

NOTE A more complete and far more amusing explanation of the reason for three configuration files is available on the Web at Apache's home page: <http://httpd.apache.org/info/three-config-files.html>.

After processing these configuration files, the master server performs some other sanity checks and opens its log files. Next, the master server, which is owned by root in most cases, launches one or more child processes (known as child servers) as defined by the process creation directives discussed shortly. These child servers might be standalone processes or lighter-weight threads, depending on how the master server was compiled and how its runtime behavior is configured. Regardless, the child servers are owned by a less-privileged user and group (`apache`, on Fedora Core and RHEL systems). These child servers do the yeoman's work of listening for requests on the HTTP port (80, by default) and answering them.

Apache is highly configurable, a fact reflected in its configuration language, which contained, at the time this paragraph was written, 350 distinct directives. Some directives are operating system specific, some provide backward compatibility for deprecated features, and some are or should be used in only very specific situations. To keep the discussion manageable and to hold your attention, the configuration directives discussed in this chapter are limited to those that affect basic server configuration, those that reflect Fedora Core and RHEL configuration options, and those that permit you to implement common configuration customizations.

The following sections follow the basic layout of the primary Apache configuration file, `/etc/httpd/conf/httpd.conf`, which is organized into

three sections. The first section configures Apache's global characteristics, the second section configures the primary or default server, and the third section configures virtual hosts.

Configuring Global Behavior

The first group of configuration directives you examine, shown in Table 23-1, control the behavior of the server as a whole.

Table 23-1 Global Configuration Directives

DIRECTIVE	DESCRIPTION
<code>Include conf.d/*.conf</code>	Includes the contents of the files in <code>conf.d/</code> whose names end in <code>.conf</code>
<code>KeepAlive Off</code>	If set to <code>On</code> , maintains an open connection to a remote client in the absence of direct contact for the time specified by <code>KeepAliveTimeout</code>
<code>KeepAliveTimeout 15</code>	Sets the number of seconds permitted to elapse between direct requests from the same client on the same connection before the server will close the connection (applies if <code>KeepAlive</code> is <code>On</code>)
<code>Listen [ipaddress:]80</code>	Determines the combination of IP address and port on which Apache listens for connections; multiple <code>Listen</code> directives may be used
<code>LoadModule modname filename</code>	Links the module or library <code>filename</code> into the server and adds it to the list of active modules using the name <code>modname</code>
<code>MaxClients 256</code>	Sets the maximum number of simultaneous connections supported
<code>MaxKeepAliveRequests 100</code>	Sets the number of requests permitted per connection
<code>MaxRequestsPerChild 4000</code>	Sets the maximum number of requests each child server fills before terminating
<code>MaxSpareServers 20</code>	Defines the maximum number of spare (idle) child servers the master server spawns
<code>MinSpareServers 5</code>	Defines the minimum number of spare (idle) child servers permitted
<code>PidFile run/httpd.pid</code>	Defines the file containing the PID of the master server process, relative to <code>ServerRoot</code>

(continued)

Table 23-1 (continued)

DIRECTIVE	DESCRIPTION
ServerLimit 256	Specifies the upper limit on the number of server processes or threads running simultaneously
ServerRoot /etc/httpd	Defines the top-level directory for Apache's configuration files and log files
ServerTokens OS	Defines the contents of the server's HTTP response header
StartServers 8	Defines the number of child servers created when Apache starts
Timeout 120	Defines the maximum time in seconds Apache waits for packet send and receive operations to complete

Some of the items in Table 23-1 bear additional discussion. When specifying the names of log files or additional configuration files, these names are appended to `ServerRoot` unless they begin with `/`. That is, if `ServerRoot` is `/etc/httpd` and a log file is specified as `logs/mylog.log`, the complete name is taken as `/etc/httpd/logs/mylog.log`, whereas `/logs/mylog.log` is interpreted as an absolute pathname.

For most sites, the default values for the configuration directives in Table 23-1 should be sufficient. Listing 23-1 shows the corresponding entries from the default Apache configuration provided by Fedora Core and RHEL, except for the `LoadModule` directives. Comments have been removed to save space and simplify the presentation.

```

ServerTokens OS
ServerRoot "/etc/httpd"
PidFile run/httpd.pid
Timeout 120
KeepAlive Off
MaxKeepAliveRequests 100
KeepAliveTimeout 15
<IfModule prefork.c>
    StartServers 8
    MinSpareServers 5
    MaxSpareServers 20
    ServerLimit 256
    MaxClients 256
    MaxRequestsPerChild 4000
</IfModule>

```

Listing 23-1 Fedora Core's global configuration directives.

```
<IfModule worker.c>
    StartServers 2
    MaxClients 150
    MinSpareThreads 25
    MaxSpareThreads 75
    ThreadsPerChild 25
    MaxRequestsPerChild 0
</IfModule>
Listen 80
Include conf.d/*.conf
```

Listing 23-1 *(continued)*

Nothing in Listing 23-1 necessarily needs to be changed, but you might consider reducing the `MaxSpareServers` value to 10 and the `StartServers` value to 5, the default values given these directives by the Apache Group. Then, as you develop a usage profile for your Web server, adjust them if necessary. Similarly, specifying `KeepAlive On` results in significant performance improvements because it eliminates the overhead involved in initiating new HTTP connections between clients and the Web server. On the other hand, if `KeepAlive` is on and you see performance problems in heavily used servers, reducing `KeepAliveTimeout` to 5 is usually preferable to simply turning `KeepAlive` off entirely.

The `MinSpareServers` and `MaxSpareServers` directives enable Apache to self-regulate, adding and deleting child processes as Web server usage fluctuates. When more than `MaxClients` attempt to connect, each connection request is put onto a queue (in particular, a FIFO, or first in, first out, queue) and serviced in the order received as current connections close. If visitors to the Web site have to wait too long, however, they'll either send another request or disconnect entirely. Busy Web sites might need to adjust this value to accommodate heavy traffic.

The `Listen 80` directive indicates that the server listens on port 80 for incoming requests. Because port 80 is the default port, this does not need to be specified, but explicitly setting the port in this way makes the configuration slightly more transparent. When using a port below 1024, one of the privileged ports, the master server must be started by root. If you wish, you can configure Apache to listen on an unprivileged port (for example, with a `Listen 8080` directive), in which case, the master server does not have to be run by the root user and group.

How and when child servers are created depends on the MPM in use, which in turn affects the behavior of some of Apache's configuration directives. For example, `MaxClients` only limits simultaneous connections as described in unthreaded servers. In threaded servers, `MaxClients` limits the number of

simultaneous connection threads per process. Requests greater than this number are typically queued, not discarded. Similarly, the `ServerLimit` directive refers to *processes*, not *threads*. Sharing a limit between processes and threads would eliminate the value and point of implementing subservers' threads.

Server control directives have different values, depending on whether the `prefork` or the `worker` MPM is loaded. If you compare the settings for the `prefork` MPM to the server process control settings for Apache 1.3x, you will see that the configuration is nearly identical. The `worker` MPM, in contrast, requires a very different configuration. For example, in the `worker` MPM, processes are started for only two servers, but each server can have as many as 25 threads (`ThreadsPerChild`). To handle load spikes, you can either increase the `ThreadsPerChild` value or increase the number of child servers (`StartServers`). For the `worker` MPM, `MaxRequestsPerChild` is 0, which means that child processes will not be killed and respawned.

Configuring the Default Server

Apache has a notion of a *default* or *primary server*, which refers to the `httpd` server that responds to any HTTP requests not handled by virtual hosts, also known as virtual servers. Without going into detail yet, a *virtual server* or *virtual host* is an `httpd` process running on the same machine as the default server, which is distinguished from the primary server by the hostname or IP address assigned to the virtual host. Despite the distinction between primary and virtual servers, configuration directives defined for the primary server also apply to virtual servers unless specifically overridden.

CROSS-REFERENCE The next section, “Configuring Virtual Servers,” discusses configuration directives specific to virtual servers and discusses virtual servers in general.

Table 23-2 lists directives used to configure the default server and the default values for each directive, where applicable.

Table 23-2 Default Server Configuration Directives

DIRECTIVE	DESCRIPTION
<code>AccessFileName .htaccess</code>	Lists one or more filenames in the complete path to the requested document that define and control access to documents in each directory or subdirectory at the same level as or below the topmost directory where the file(s) specified by <code>AccessFileName</code> (if any) is found

Table 23-2 (continued)

DIRECTIVE	DESCRIPTION
AddCharset charset	Adds the character set <code>charset</code> to Apache's list of supported character sets
AddDefaultCharset UTF-8	Sets the default output character set Apache uses if no character set is requested by browsers or specified in HTTP request headers
AddHandler handler mimetype	Associates <code>handler</code> with the MIME type <code>mimetype</code>
AddIcon icon name	Maps <code>icon</code> as the icon to display next to files ending with <code>name</code> ; used with the <code>FancyIndexing</code> directive
AddIconByEncoding icon mimeencoding	Associates <code>icon</code> with files whose MIME encoding is <code>mimeencoding</code> ; used with the <code>FancyIndexing</code> directive
AddIconByType icon mimetype	Sets <code>icon</code> as the icon to display next to files with the MIME type of <code>mimetype</code> ; used with the <code>FancyIndexing</code> directive
AddLanguage mime-lang name	Maps the filename extension specified by <code>name</code> to the MIME language <code>mime-lang</code> , overriding previous mappings for <code>name</code>
AddOutputFilter output mimetype	Associates an output filter for server output of type <code>output</code> with the MIME type <code>mimetype</code>
AddType mimetype name	Adds the specified <code>mimetype</code> for files ending in <code>name</code> to the list of MIME types read from the <code>TypeConfig</code> file
Alias urlpath dirpath	Maps the directory <code>urlpath</code> , specified relative to <code>DocumentRoot</code> , to the file system directory <code>dirpath</code> enabling Apache to access a directory otherwise inaccessible because all content must be under <code>DocumentRoot</code>
CustomLog logs/ access_log combined	Defines the name of Apache's access log and the log format used when logging requests to the server
DefaultIcon /icons/ unknown.gif	Sets the default icon displayed next to files whose MIME or content type cannot be determined; used with the <code>FancyIndexing</code> directive

(continued)

Table 23-2 (continued)

DIRECTIVE	DESCRIPTION
<code>DefaultType text/plain</code>	Defines the default MIME type when a requested document's MIME type cannot be determined using the <code>TypesConfig</code> or <code>AddType</code> directives
<code><Directory dirname> directives </Directory></code>	Delimits a set of configuration directives that apply to <code>dirname</code> and its subdirectories, if any
<code>DirectoryIndex filename</code>	Specifies one or more filenames that can serve as a directory index when a request does not specify a particular file or document
<code>DocumentRoot "/var /www/html"</code>	Sets the base directory from which all requested documents will be served; document URLs (filenames) are interpreted relative to <code>DocumentRoot</code> ; see also <code>UserDir</code>
<code>ErrorLog logs/error_log</code>	Defines the name of Apache's error log, relative to <code>ServerRoot</code> if the filename does not begin with <code>/</code>
<code>ForceLanguagePriority</code>	Uses the values specified in <code>LanguagePriority</code> to return a valid result. <code>Fallback</code> returns a valid choice from those listed in <code>LanguagePriority</code> ; <code>Prefer</code> returns the first matching value from acceptable languages specified by the client
<code>Group [#]apache</code>	Specifies the group name or, if prefixed with <code>#</code> , the GID under which the child servers execute
<code>HeaderName HEADER.html</code>	Defines <code>HEADER.html</code> as the file whose contents will be inserted at the top of a directory listing
<code>HostnameLookups Off</code>	Controls whether Apache performs DNS lookups on connecting hosts in order to log hostnames
<code>IndexIgnore filespec</code>	Defines one more filename patterns <code>filespec</code> that the directory indexer will ignore
<code>IndexOptions opts</code>	Sets the options controlling the behavior of Apache's directory indexing feature
<code>LanguagePriority spec</code>	Specifies a list of language codes in <code>spec</code> that indicates the order in which languages codes will be applied during content negotiation

Table 23-2 (continued)

DIRECTIVE	DESCRIPTION
<code>LogFormat str name</code>	Defines a log file format <code>str</code> named <code>name</code> , which Apache uses for messages it logs in the access log (see also <code>TransferLog</code> and <code>CustomLog</code>)
<code>LogLevel warn</code>	Controls the amount and detail of information Apache records in its error log
<code>ReadmeName README.html</code>	Defines <code>README.html</code> as the file whose contents will be appended to the end of a directory listing
<code>ScriptAlias urlpath dirpath</code>	Functions exactly like the <code>Alias</code> directive and also indicates that <code>dirpath</code> contains executable CGI scripts
<code>ServerAdmin root@localhost</code>	Defines the email address included in error messages displayed to client connections
<code>ServerName</code>	Specifies an alternative name for the server, such as <code>www.mydomain.com</code> , that is different than the host's actual name (<code>webbeast.mydomain.com</code>)
<code>ServerSignature On</code>	Directs Apache to append the <code>ServerName</code> and version number as a footer to generated documents, such as error message, FTP file listings, and so forth
<code>TypesConfig /etc /mime.types</code>	Sets the filename of the MIME types configuration file (relative to <code>ServerRoot</code> if the filename does not begin with <code>/</code>), which maps filename extensions to content types (see also <code>AddType</code>)
<code>User [#]apache</code>	Specifies the user name or, if prefixed with <code>#</code> , the UID under which the child servers execute
<code>UserDir public_html</code>	Defines the subdirectory in a user's home directory that is used when clients request documents belonging to a specific user

To facilitate further discussion of the directives in Table 23-2, Listing 23-2 shows the corresponding entries from Red Hat's default Apache configuration. The listing excludes comments to conserve space and simplify the discussion (ellipses indicate deleted directives).


```

User apache
Group apache
ServerAdmin root@localhost
DocumentRoot "/var/www/html"
<Directory />
    Options FollowSymLinks
    AllowOverride None
</Directory>
<Directory "/var/www/html">
    Options Indexes FollowSymLinks
    AllowOverride All
    AcceptPathInfo On
    Order allow,deny
    Allow from all
</Directory>
<IfModule mod_userdir.c>
    UserDir disable
    #UserDir public_html
</IfModule>
DirectoryIndex index.php index.html index.html.var
AccessFileName .htaccess
<Files ~ "^\.ht">
    Order allow,deny
    Deny from all
</Files>
TypesConfig /etc/mime.types
DefaultType text/plain
<IfModule mod_mime_magic.c>
    MIMEMagicFile conf/magic
</IfModule>
HostnameLookups Off
ErrorLog logs/error_log
LogLevel warn
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\""
    combined
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent
CustomLog logs/access_log combined
ServerSignature On
Alias /icons/ "/var/www/icons/"
<Directory "/var/www/icons">
    Options Indexes MultiViews
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>
<IfModule mod_dav_fs.c>
    # Location of the WebDAV lock database.

```

Listing 23-2 Default server configuration.

```

    DAVLockDB /var/lib/dav/lockdb
</IfModule>
ScriptAlias /cgi-bin/ "/var/www/cgi-bin/"
<Directory "/var/www/cgi-bin">
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all
</Directory>
AddIconByEncoding (CMP,/icons/compressed.gif) x-compress x-gzip
AddIconByType (TXT,/icons/text.gif) text/*
AddIconByType (IMG,/icons/image2.gif) image/*
AddIconByType (SND,/icons/sound2.gif) audio/*
AddIconByType (VID,/icons/movie.gif) video/*
AddIcon /icons/binary.gif .bin .exe
...
AddIcon /icons/back.gif ..
AddIcon /icons/hand.right.gif README
AddIcon /icons/folder.gif ^^DIRECTORY^^
AddIcon /icons/blank.gif ^^BLANKICON^^
DefaultIcon /icons/unknown.gif
ReadmeName README.html
HeaderName HEADER.html
IndexIgnore .??* *~ *# HEADER* README* RCS CVS *,v *,t
AddLanguage ca .ca
...
AddLanguage zh-TW .zh-tw
LanguagePriority en ca cs da de el eo es et fr he hr it ja ko ltz nl nn
    no pl pt pt-BR ru sv zh-CN zh-TW
ForceLanguagePriority Prefer Fallback
AddDefaultCharset UTF-8
AddCharset ISO-8859-1 .iso8859-1 .latin1
...
AddCharset shift_jis .sjis
AddType application/x-compress .Z
AddType application/x-gzip .gz .tgz
AddHandler imap-file map
AddHandler type-map var
AddType text/html .shtml
AddOutputFilter INCLUDES .shtml
Alias /error/ "/var/www/error/"
<IfModule mod_negotiation.c>
    <IfModule mod_include.c>
        <Directory "/var/www/error">
            AllowOverride None
            Options IncludesNoExec
            AddOutputFilter Includes html
            AddHandler type-map var
            Order allow,deny

```

Listing 23-2 (continued)

```

        Allow from all
        LanguagePriority en es de fr
        ForceLanguagePriority Prefer Fallback
    </Directory>
</IfModule>
</IfModule>
BrowserMatch "Mozilla/2" nokeepalive
BrowserMatch "MSIE 4\.0b2;" nokeepalive downgrade-1.0 force-response-1.0
BrowserMatch "RealPlayer 4\.0" force-response-1.0
BrowserMatch "Java/1\.0" force-response-1.0
BrowserMatch "JDK/1\.0" redirect-carefully
BrowserMatch "Microsoft Data Access Internet Publishing Provider" redirect-carefully
BrowserMatch "^WebDrive" redirect-carefully
BrowserMatch "^WebDAVFS/1.[012]" redirect-carefully
BrowserMatch "^gnome-vfs" redirect-carefully

```

Listing 23-2 *(continued)*

The `User` and `Group` directives indicate that the child servers are owned by the user and group `apache`, which are safer because they do not have the same privileges as the root user does. To run the child servers under less privileged users in this fashion, the master server must be started by the root user. Why? Only processes running with root permissions can change their UID and GID at runtime in order to maintain the Linux security model.

The `ServerName` directive defines the name returned to clients if it is different from the host server's actual name. For example, if the server's DNS name is `webbeast.mydomain.com`, you can specify `ServerName www.mydomain.com` for the server to respond to requests sent to `http://www.mydomain.com/`.

The `DocumentRoot /var/www/html` directive sets the server's base document directory to `/var/www/html`, meaning that all URLs are served relative to this directory. For example, if the server is named `www.example.com`, then given a client request for the URL `http://www.example.com/index.html`, the server would return the file `/var/www/html/index.html` to the client.

Each `<Directory></Directory>` block configures access information for the named directory (or directories) and its subdirectories. The first block sets the default permissions for all directories:

```

<Directory />
    Options FollowSymLinks
    AllowOverride None
</Directory>

```

In this case the applicable directory is the server's root directory, /. The access information specified for the server's root directory applies to all subdirectories and is fairly restrictive. Accordingly, if you want looser access to certain subdirectories or want to set additional options of certain subdirectories or document trees, you have to configure the subdirectories specifically. Otherwise, the defaults set for the server root will apply. Other `<Directory></Directory>` blocks include `/var/www/html`, `/var/www/icons`, and `/var/www/error`.

The `Options` directive specifies the server features that apply to the named directory. Values for the `Options` directive can be placed in a space-delimited list of one or more of the following:

- **All** — Enables all options except `MultiViews`. `All` is the default option.
- **ExecCGI** — Enables execution of CGI scripts.
- **FollowSymLinks** — Enables the server to follow symbolic links in this directory.
- **Includes** — Enables SSI (server-side includes).
- **IncludesNOEXEC** — Enables SSI but disables the SSI `#exec` command and the use of `#include` for CGI scripts.
- **Indexes** — Instructs the server to return a formatted listing of a directory for which no directory index as determined by the `DirectoryIndex` directive exists.
- **MultiViews** — Enables `MultiView` searches. If the server receives a request for a resource that does not exist, for example, `/docs/resource`, the server scans the directory for all files named `resource.*`, if any, assigns them the same media types and content encodings they would have had if the client had asked for one of them by name, chooses the best match to the client's requirements, and returns that document. Using `MultiViews` can cause an serious performance hit, and multiple pages point to a resource that doesn't exist but whose name has many close matches in that directory.
- **None** — Disables all special directory features in the associated directory and its subdirectories.
- **SymLinksIfOwnerMatch** — Instructs the server to follow only those symbolic links for which the target file or directory has the same UID as the link.

Options preceded by a `+` are added to `Options` currently in force, and those preceded by a `-` are removed from `Options` currently in force. If multiple `Options` could apply to a directory, then the most specific one is applied.

However, if all of the options in an `Options` directive are prefixed with `+` or `-`, the options are merged.

So, the only option enabled for all of the directories under the server root (`/`) is the `FollowSymLinks` option. From this point forward, any divergence from this default must be specified, as in the following directory block:

```
<Directory "/var/www/html">
    Options Indexes FollowSymLinks
    AllowOverride All
    AcceptPathInfo On
    Order allow,deny
    Allow from all
</Directory>
```

The `AllowOverride` directive tells the server which groups of directives can be overridden by per-directory configuration files. Possible values are:

- **All** — The same as specifying `AuthConfig FileInfo Limit`
- **AuthConfig** — User access directives
- **FileInfo** — Document type directives
- **Limit** — Host access directives
- **None** — No directives can be overridden on a per-directive basis

If set to `None`, the server ignores per-directory configuration files. If set to `All`, any directive valid in the current context is enabled. For the server's root directory and its subdirectories, therefore, the server ignores directives files unless `AllowOverride Something` is specifically set for a given directory under the server root, where `Something` is not `None`.

`AcceptPathInfo` controls whether trailing path information in a request will be honored or rejected. Given a file named `/mystuff/somefile.html`. If the browser requests `/mystuff/somefile.html/dirname`, the trailing path information, is `/dirname` (which will be stored in the server environment variable `PATH_INFO`). The main use of `AcceptPathInfo` is to override the default disposition of path information, especially for those content handlers that use input or output filters, such as SSI files. Such handlers often use `PATH_INFO` to generate content dynamically. If `AcceptPathInfo` is `Default`, the server uses the default policy set by the handler; if it is `On`, the server accepts the trailing path information regardless of what the handler's policy is and the path information points to a resource that exists; if it is `Off`, the server rejects the trailing path information, again disregarding the handler's default policy.

The `Order` directive controls the default access policy for various resources, such as files and directories, and the order in which the server evaluates `Allow` and `Deny` directives for those resources. `Order` can be one of the following:

- **Order Deny, Allow** — Evaluates Deny directives before Allow directives and enables access by default. Clients that do not match a Deny directive or that do match an Allow directive will be permitted access.
- **Order Allow, Deny** — Evaluates Allow directives before Deny directives and denies access by default. Clients not matching an Allow directive or matching a Deny directive are denied access.

For the server root directory, for example, all clients are denied access to it, and Allow directives are evaluated before Deny directives. For instance, consider the following snippet from Listing 23-2:

```
<Files ~ "^\.ht">
    Order allow,deny
    Deny from all
</Files>
```

The `<Files></Files>` block specifies that access to all files beginning with `.ht` is denied unconditionally to all clients, preventing clients from viewing their contents, an important measure considering that access files can contain security-sensitive information.

All directives inside an `<IfModule></IfModule>` block are evaluated only if the indicated module is loaded. The default configuration file has a number of such blocks. For example, consider the following `IfModule` directive:

```
<IfModule mod_mime_magic.c>
    MIMEMagicFile conf/magic
</IfModule>
```

If the `mod_mime_magic` module is loaded, the `MIMEMagicFile` directive causes Apache to read the contents of the configuration file named `/etc/httpd/conf/magic`. This file enables Apache to determine the contents of files and thus to determine their MIME types by reading the first few bytes of a file. `mod_mime_magic` works much like the Linux `file` command works and is, in fact, based on an older version of the `file` command. The `MIMEMagicFile` directive complements and extends the MIME typing provided by the `TypesConfig /etc/mime.types` directive.

The `IfModule` block for `mod_userdir` marks a departure from previous practice vis-à-vis users' personal Web pages. `UserDir` specifies the directory in users' home directories that contain personal Web pages. User-specific Web pages are disabled by default because they can be used to test the presence of a username on the system. To enable access to `public_html` in each user's home directory, comment out `UserDir disable` in `httpd.conf` file, and uncomment the line that reads:

```
#UserDir public_html
```

The `DirectoryIndex` lines specifies the files Apache looks for when passed a URL that does not specify a filename, that is, when the URL specifies a directory, such as `http://www.example.com/products/`, instead of a file, such as `http://www.example.com/products/prodlist.html`. Apache searches for the files in the order listed and the first match wins. In the `httpd.conf` file shipped, the order is:

1. `index.php`
2. `index.html`
3. `index.html.var`

The `TypesConfig`, `DefaultType`, and `MIMEMagicFile` directives work together to help Apache determine file types. `TypesConfig` tells Apache where to find a list or database of MIME types (`/etc/mime.types` in this case). This file maps MIME types to filename extensions. Consider the following two entries from `/etc/mime.types`, for example:

```
text/html      html htm
video/mpeg     mpeg mpg mpe
```

The first line means that files ending with the extensions `html` and `htm` (rather, `.html` and `.htm`) should be considered to be standard HTML files. Similarly, the second line indicates that files having the filename extensions `.mpeg`, `.mpg`, and `.mpe` have the MIME type `video/mpeg`. These mappings are important because Apache sends them the MIME types to Web clients, which use the information to determine how to display the associated content. Files with a MIME type of `text/html` will ordinarily be displayed as a normal Web page, while Web browsers might start an MPEG player to display files that have a MIME type of `video/mpeg`.

`DefaultType text/plain` provides a default MIME type (plain text) for any file that Apache serves for which a MIME type is not defined. `MIMEMagicFile` supplements the MIME typing system, making it possible for Apache to determine a file's MIME type by examining the contents of a file.

The logging directives control the level and format of Apache's log output. The directive `ErrorLog logs/error_log` specifies the error log Apache uses. The four `LogFormat` directives define log formats named `combined`, `common`, `referer`, and `agent` (yes, `referer` is misspelled). These format names can then be used in other log-related directives to identify the output format. For example, the directive `CustomLog logs/access_log combined` uses the combined format defined. The `CustomLog` directive indicates the file used to log all requests sent to the server. Access log entries might resemble the following:

```

127.0.0.1 - - [26/Mar/2005:09:47:33 -0500] "GET /manual/images/up.gif HTTP/1.1" 200 57 "http://localhost/manual/mod/core.html" "Mozilla/5.0 (X11; \
U; Linux i686; en-US; rv:1.7.6) Gecko/20050302 Firefox/1.0.1 Fedora/1.0\
.1-1.3.2"
127.0.0.1 - - [26/Mar/2005:09:47:39 -0500] "GET /manual/mod/quickreference.html HTTP/1.1" 200 95342 "http://localhost/manual/mod/directives.html" \
"Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.7.6) Gecko/20050302 Firefox\
x/1.0.1 Fedora/1.0.1-1.3.2"
127.0.0.1 - - [26/Mar/2005:10:19:53 -0500] "GET /manual/mod/mod_access.html HTTP/1.1" 200 18557 "http://localhost/manual/mod/quickreference.html" \
"Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.7.6) Gecko/20050302 Firefox\
x/1.0.1 Fedora/1.0.1-1.3.2"

```

The log entries don't break as shown in the text. These access log entries record the requests the server processed when opening a Web page named `ssitest.html` (see Listing 23-3).

The long series of `AddIconByEncoding`, `AddIconByType`, and `AddIcon` directives define the various icons displayed next to files with a given icon. The directive `AddIcon /icons/binary.gif .bin .exe`, for example, indicates that files ending with `.bin` and `.exe` should have the image `/icons/binary.gif` displayed next to them. Note that the directory `/icons/` was aliased to `/var/www/icons/` using the `Alias` directive `Alias /icons/ "/var/www/icons/"` earlier in the `httpd.conf` file, so the file system path to `binary.gif` is expanded to `/var/www/icons/binary.gif`. As a fall-back measure, the directive `DefaultIcon /icons/unknown.gif` defines the default icon Apache displays if it cannot determine the file type of a given file based on the definitions given by the `TypesConfig` and `MIMEModMagic` directives and additional types appended to the MIME type listing using `AddType` directives.

To handle non-HTML document types (MIME types), Apache uses the `AddType` and `AddHandler` directives. `AddType` is used for MIME types that are not specified in MIME configuration files (as set with the `MIMEMagicFile` directive) or for predefined MIME types that you want to override. It works by associating filename extensions with specific content handlers. Its general syntax is:

```
AddType mimetype extension [...]
```

This directive associates one or more filename endings specified by `extension` with the MIME type specified by `mimetype`. For example, the following directive assigns the MIME type `text/html` (a normal HTML file) with files whose filename extension is `.shtml`:

```
AddType text/html .shtml
```


AddHandler's purpose is to define a content handler for specific MIME types. Its syntax is comparable to AddType's:

```
AddHandler handler extension [...]
```

This directive associates any file that has the extension *extension* with the content handler named by *handler*. The following statement, for instance, instructs Apache to use the image map handler *imap-file* with files whose name ends with *map*:

```
AddHandler imap-file map
```

The AddLanguage directives map filenames to language encodings. So, for example, files ending with *.en* are treated as English documents, and files ending with *.en.gz* or *.en.tgz* are treated as gzip compressed English documents. The LanguagePriority directive, similarly, determines which file the server returns if the browser does not indicate a preference. For example, if the files *index.en.html* and *index.fr.html* both exist and a client does not specify a preferred content language, the server returns *index.en.html*.

AddDefaultCharset and AddCharset load Apache's support for various character sets. AddDefaultCharset specifies the default character set Apache uses to serve content if a browser does not specifically request one. The default character set in Fedora Core and RHEL is UTF-8.

The BrowserMatch directives set environment variables that can be used in CGI scripts and SSI based on the information in the User-Agent HTTP request header field. The first argument is the text to match from the request header. The second and subsequent arguments name the variables to set and the value or values to which to set the variables. The variable assignments can take one of the following forms:

- **varname** — Sets *varname* to 1
- **!varname** — Removes (unsets) *varname* if it was already set
- **varname=value** — Assigns *value* to *varname*

If a User-Agent header matches multiple BrowserMatch strings, Apache merges the matching strings. Entries are processed in the order in which they appear, and later entries can override earlier ones.

TIP If you do not know or cannot be sure of the case of the request header you want to match, you can use the `BrowserMatchNoCase` directive, which matches regardless of case.

Configuring Virtual Servers

Virtual servers (also referred to as *virtual hosts*) are primarily used to run Web servers for multiple domains on a single physical system. Virtual servers can also be used to allow multiple workgroups or departments that share the same network or subnet to maintain independent Web pages without requiring individual dedicated departmental servers. Virtual hosts fall into two categories: IP-based virtual hosts and name-based virtual hosts.

IP-based virtual hosts refer to Web servers that have different IP addresses. In some cases, the different IP addresses correspond to different Ethernet interfaces, such as `eth0` and `eth1` (machines that have multiple Ethernet interfaces are called *multihomed hosts*). In other cases, a single Ethernet interface is assigned multiple IP addresses using aliases. In this configuration, a system might have a single Ethernet interface, say, `eth0`, and a single primary IP addresses, for example, `192.168.0.1`. The aliases have IP addresses of the form `192.168.0.1:n`, where *n* is a digit such as 1, 2, or 3. Regardless of the method you use to implement virtual hosts, end users will not realize that the Web sites they are viewing reside on the same physical server. Table 23-3 shows the Apache configuration directives that control the configuration and behavior of virtual servers.

Table 23-3 Virtual Server Configuration Directives

DIRECTIVE	DESCRIPTION
<code><VirtualHost ipaddr[:port]></code> directives <code></VirtualHost></code>	Defines a virtual host whose IP address is <code>addr</code> (listening on <code>port</code> , if specified); <code>directives</code> are one or more of the directives listed previously and override the directives listed for the default server.
<code>NameVirtualHost ipaddr[:port]</code>	Defines the IP address <code>addr</code> (listening on <code>port</code> , if specified) for a name-based virtual host.
<code>ServerAlias altname</code>	Enables the virtual server to respond to one or more alternate hostnames <code>altname</code> when used with name-based virtual hosts. A single <code>VirtualHost</code> directive can have multiple <code>ServerAlias</code> statements.
<code>ServerName fqdn</code>	Sets the name of the virtual server to the FQDN <code>fqdn</code> .

A bare-bones set of name-based virtual server definitions might resemble the following:

```
Listen 80
[...]
NameVirtualHost *:80

<VirtualHost *:80>
    ServerName webbeast.example.com
    DocumentRoot /var/www/webbeast
    # other directives
</VirtualHost>

<VirtualHost *:80>
    ServerName www.mydomain.com
    DocumentRoot /var/www/mydomain
    # other directives
</VirtualHost>

<VirtualHost *:80>
    ServerName www.yourdomain.org
    DocumentRoot /var/www/yourdomain
    # other directives
</VirtualHost>
```

This example shows three virtual hosts, `webbeast.example.com`, `www.mydomain.com`, and `www.yourdomain.org`, all of which have the same IP address. For the purpose of this example, the actual IP address doesn't matter because the asterisks match all IP addresses, but suppose that the address is 192.168.0.2. One of the side effects of using the asterisk is that the main server won't answer any requests. Apache will pass all requests to the appropriate virtual host, depending on the name specified in the request. Furthermore, the virtual host `webbeast.example.com` is the default or primary server because it is the first listed host. As a result, it will answer any request that isn't answered by one of the other virtual hosts.

As stated earlier in the chapter, configuration directives for the default server also apply to virtual servers unless specifically overridden in a `<VirtualHost>` block. Therefore, if your virtual hosts require special configuration needs not provided or explicitly disabled or disallowed in the default server, you must specify these custom configuration directives inside the appropriate `<VirtualHost>` block.

If you add a name-based virtual host to an existing Web server, you must also add a virtual host for the existing Web server. Moreover, the directives in the virtual host you create for the original, single-site server must match those you specified for the original single-site server. In addition, the virtual host

you create for the existing server should appear before any other virtual hosts so that it will act as the default server. If you fail to add a virtual host for the existing server, requests that should be answered by the existing Web server will be answered by the added virtual host. Why? When a request comes in, Apache first looks to see if the requested name matches an address specified for a `NameVirtualHost`. Because all of the IP addresses in a name-based virtual host are the same, Apache routes the request to the *first matching virtual host*, bypassing the default server.

Starting and Stopping Apache

To start and stop Apache, the preferred method is to use the `httpd` initialization script and the `service` utility, as shown in the following examples:

```
# service httpd start
Starting httpd:                [ OK ]
# service httpd stop
Stopping httpd:                [ OK ]
```

There are some additional arguments you might want to use, such as `restart`, `reload`, and `configtest`. As you might guess, the `restart` argument stops and starts Apache. The `reload` argument signals Apache to reload its configuration files and is a good way to refresh a running server's configuration without restarting it and closing all active connections. The `configtest` argument causes Apache to parse its configuration files. If it detects an error, it will display an error message indicating what went wrong and where in the configuration file it found the error. This is a very easy way to test configuration changes without causing havoc for yourself or users of your Web site.

(NOT) USING THE APACHE CONFIGURATION TOOL

If you prefer graphical configuration tools, you can configure Apache's basic functionality using HTTP Configuration Tool. HTTP Configuration Tool enables you to edit the `/etc/httpd/conf/httpd.conf` configuration file for the Apache HTTP server. Using the graphical interface, you can configure directives such as virtual hosts, logging attributes, and server control parameters. To start HTTP Configuration Tool, type `system-config-httpd` at a command prompt or select Red Hat ⇨ System Settings ⇨ Server Settings ⇨ HTTP Server. However, we do not recommend using HTTP Configuration Tool on your systems because it has the annoying habit of overwriting changes made outside of the tool and it does not recognize manually installed Web servers that don't store their configuration information in `/etc/httpd/conf` or `/etc/httpd/conf.d`.

Implementing SSI

Server-side includes (SSI) are specially formatted statements placed in HTML documents and evaluated by the server before the server sends the document to a client. SSI lets you add dynamically generated content to an existing HTML page without needing to generate the entire page using CGI or another dynamic page generation technique. SSI is best used to add small amounts of dynamically generated content to otherwise static documents. Server-side includes are also a great way to specify standard, static header and footer info on Web pages. SSI content doesn't have to be dynamic. For simple sites, it's a great alternative to PHP, Perl, and other fuller-featured approaches for including headers, footers, style sheets, and so forth in Web pages.

The stock Fedora Core and RHEL configuration includes support for SSI using the statements:

```
AddType text/html .shtml
AddOutputFilter INCLUDES .shtml
```

The first line adds the file extension `.shtml` to the `text/html` MIME type. The `AddOutputFilter` directive tells Apache that files with an `.shtml` extension should be processed using `mod_include`, the module that provides Apache's SSI support (the default Red Hat `httpd.conf` file should contain these directives).

TIP If, for some reason, you have to add the `AddType text/html .shtml` and `AddOutputFilter INCLUDES .shtml` directives to the `httpd.conf` file, the server must be restarted to make them take effect. You can use one of the following commands to force Apache to reread its configuration file:

```
# service httpd restart
# service httpd reload
```

The first command stops and restarts the server. The second one sends Apache the `SIGHUP` signal, which causes it to reread `httpd.conf`. The effect is the same regardless of which command you use.

However, you need to tell Apache which directories contain content it should parse for SSI content. To do so, add the `Includes` argument to the `Options` statement for the directory in which you want SSI to work. For example, suppose that you create a directory name `/var/www/html/tests` and want to enable SSI for this directory. Add a file named `tests.conf` to `/etc/httpd/conf.d` that contains the following `<Directory>` block:

```

<Directory "/var/www/html/tests">
    Options Indexes FollowSymLinks Includes
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>

```

The `Options Includes` directive instructs Apache to parse files it serves from this directory for SSI directives. Next, create the Web page shown in Listing 23-3, naming it `ssitest.shtml` and placing it in `/var/www/html/tests`:

```

<html>
<head>
    <title>SSI Test Page</title>
    <link rel="stylesheet" type="text/css" href="rhlnsa3.css">
</head>

<body>

<h1>SSI Test Page</h1>
<div id="content">
<pre>
<!--#exec cmd="ls -lh /var/www" -->
</pre>
</div> <!-- content -->
<!--#include virtual="footer.html" -->

</body>
</html>

```

Listing 23-3 An SSI test page.

SSI directives look like HTML comments. They take the following general form:

```

<!--#element attribute=value ... -->

```

Because SSI directives look like comments, if SSI is improperly configured on the server, the browser ignores the contents. Otherwise, the server creates properly formatted HTML output that Web browsers render properly. In Listing 23-3, the first SSI directive is `<!--#exec cmd="ls -lh /var/www" -->`, which uses the built-in `exec` command to execute `ls -lh /var/www`, embedding the output of this command in `<pre></pre>` tags to maintain the appropriate formatting. The second SSI directive, `include virtual=/includes/footer.html`, includes a standard footer file. Finally, open the document in your

Web browser, using the URL `http://localhost/tests/ssitest.shtml` if accessing the server locally or `http://your.server.name/tests/ssitest.shtml` if accessing the server remotely, replacing your `.server.name` with the name of your Web server. Figure 23-2 shows how the page appears in the Firefox Web browser.

As you can see in Figure 23-2, the SSI statement shows output of the `ls -lh` command. For comparison purposes, `ls -lh` executed in a terminal window might resemble the following:

```
$ ls -lh /var/www
total 28K
drwxr-xr-x  2 root      root   33 May 19 02:07 cgi-bin
drwxr-xr-x  3 root      root  4.0K May 19 01:05 error
drwxr-xr-x  4 root      root   33 May 22 00:04 html
drwxr-xr-x  3 root      root  8.0K May 19 01:47 icons
drwxr-xr-x 14 root      root  8.0K May 19 01:05 manual
drwxr-xr-x  2 root      root  162 May 19 01:52 mrtg
drwxr-xr-x  2 root      root   61 May 19 02:09 nut-cgi-bin
drwxr-xr-x  2 webalizer root   43 May 19 01:05 usage
```

After confirming that SSI is properly configured using the test page, the SSI configuration is complete.

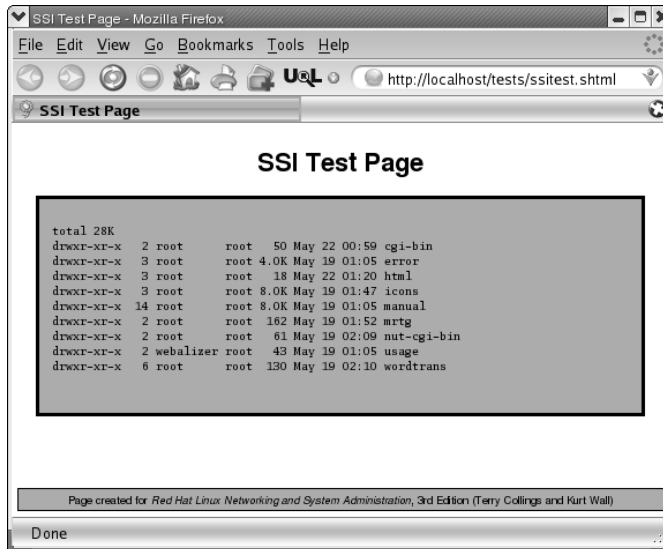


Figure 23-2 Viewing `ssitest.html` in Firefox.

Enabling CGI

CGI, the Common Gateway Interface, is a protocol that defines a standard method enabling Web servers to communicate with external programs. These programs are known as *CGI scripts*, *CGI programs*, or, more colloquially, just *CGIs*. Many programming and scripting languages provide CGI implementations so that you can use them in Web pages. Perl is the Big Daddy in the Linux world, but it is far from the only option: Python, Ruby, and even Bash can be used to create CGIs. CGI scripts are commonly used to create or update Web pages or parts of Web pages dynamically. In this respect, CGIs are much like SSI, but CGI is far more flexible than SSI and provides additional functionality that SSI cannot. For example, CGI scripts can be used for user authentication, to create a user interface on a Web page, and, within limits, in any situation in which a Web-based interface is used to execute programs and display the results in a near real-time environment. This section briefly explains Apache configuration directives and procedures that enable CGI.

As you might suspect by this point, your first task is to ensure that Apache's configuration permits CGI script execution. The `ScriptAlias` directive associates a directory name with a file system path, which means that Apache treats every file in that directory as a script. If not present, add the following directive to `httpd.conf`:

```
ScriptAlias /cgi-bin/ "/var/www/cgi-bin"
```

This directive tells Apache that any URL beginning with `/cgi-bin/` should be served from `/var/www/cgi-bin`. Thus, given a URL of `http://localhost/cgi-bin/cgiscript.pl` or `http://your.server.name/cgi-bin/cgiscript.pl`, Apache reads and executes the script `/var/www/cgi-bin/cgiscript.pl`. If necessary, modify the configuration file to include the `ScriptAlias` directive shown, and restart Apache as explained previously. Then use the script in Listing 23-4 to test the configuration.

```
#!/usr/bin/perl

print 'Content-type: text/html\r\n\r\n';
print '<html>\n';
print '<head>\n';
print '<title>CGI Test Page</title>\n';
print '<link rel="stylesheet" type="text/css" \
      href="/tests/rhlnsa3.css">\n';
```

Listing 23-4 A CGI test script. (*continued*)


```

print '</head>\n';
print '<body>\n';
print '<h1>CGI Test Page</h1>\n';
print '<div id='content'>\n';
system 'ls -lh /var/www';
print '</div> <!-- content -->\n';
print '<div id='footer'>\n';
system 'cat /var/www/html/tests/footer.html';
print '</div> <!-- footer -->\n';
print '</body>\n';
print '</html>\n';

```

Listing 23-4 (continued)

Save this script as `cgitest.pl`, make it executable (**`chmod 755 cgitest.pl`**), and then put it in `/var/www/cgi-bin`. Finally, open the URL `http://localhost/cgi-bin/cgitest.pl` if accessing the server locally or `http://your.server.name/cgi-bin/cgitest.pl` if accessing the server remotely, replacing `your.server.name` with the name of your Web server. Figure 23-3 shows sample output from the CGI test script.

If you see similar output, your server's CGI configuration works. If you enable CGI execution for other directories, make sure to test those configuration options as well before putting the server into production.

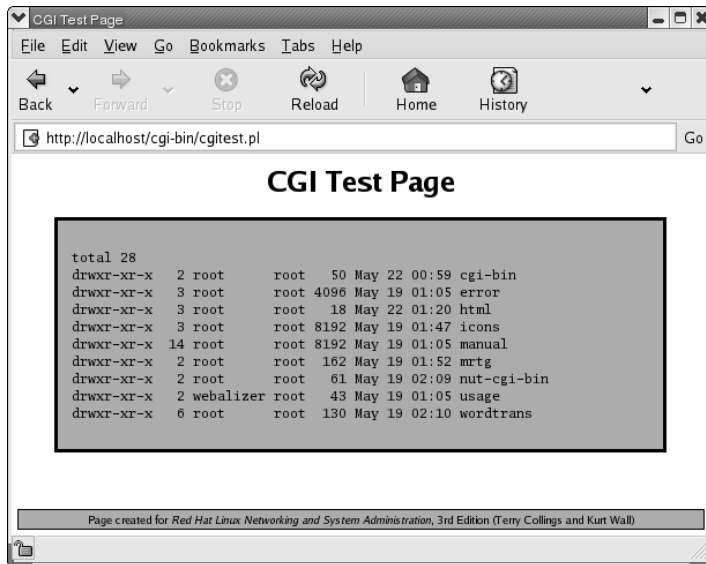


Figure 23-3 Viewing the CGI test script in the Epiphany Web browser.

Enabling PHP

PHP is an extremely popular and capable HTML scripting language. As shipped in Fedora Core and RHEL, PHP is enabled and ready to run, so this section simply presents a short PHP script you can use to make sure that PHP is working properly. Create the PHP script shown in Listing 23-5, and save it as `/var/www/html/tests/phptest.php`.

```
<html>
<head>
  <title>PHP Test Page</title>
  <link rel="stylesheet" type="text/css" href="rhlnsa3.css">
</head>

<body>

<h1>PHP Test Page</h1>
<div id="content">
<pre>
<?php
system("ls -lh /var/www");
?>
</pre>
</div> <!-- content -->
<?php include("footer.html"); ?>
</body>
</html>
```

Listing 23-5 A PHP test script.

Open the document in your Web browser, using the URL `http://localhost/tests/phptest.shtml` if accessing the server locally or `http://your.server.name/tests/phptest.php` if accessing the server remotely, replacing `your.server.name` with the name of your Web server. Figure 23-4 shows how the page appears in the Konqueror Web browser.

If you see comparable output, PHP is working correctly. By way of explanation, the PHP script uses the `system()` function to invoke `ls -lh /var/www`, which in turn displays the file listing shown in Figure 23-4.

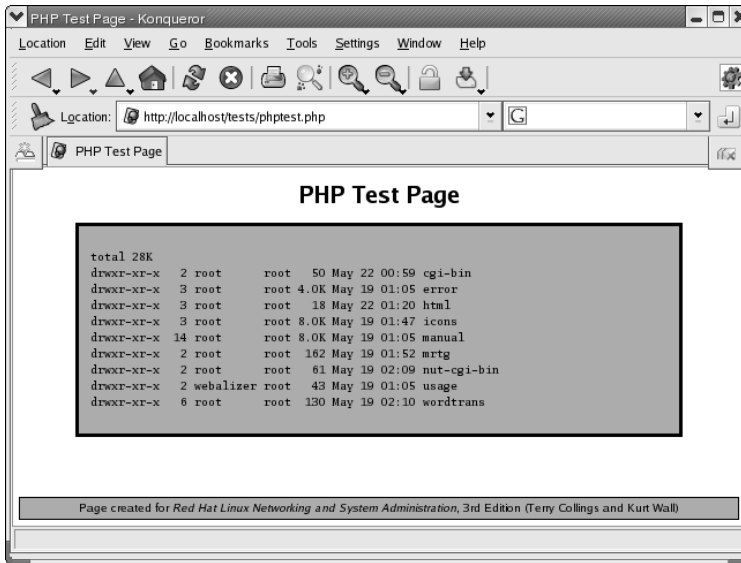


Figure 23-4 Viewing the PHP test script in the Konqueror Web browser.

Creating a Secure Server with SSL

Lamentably, the Internet is a much less secure place than it used to be. If the Web site you administer will be used for electronic commerce or for exchanging any type of information that needs to be kept private, these transactions need to be secure. SSL-enabled Web sites use a different URL prefix, `https`, to indicate that HTTP protocol request and document transfers are encrypted. You've probably visited SSL-enabled Web sites yourself. This section describes how to create a secure Web server using the Secure Sockets Layer (SSL) to encrypt communications between your Web server and Web clients. It gives an overview of SSL, describes how digital certificates fit into the security picture, and how to create a self-signed certificate. A final section discusses obtaining a digital certificate from a recognized certificate authority and lists a number of certificate authorities from which you can obtain a valid certificate.

For more information about SSL and certificate creation, the following online resources will prove helpful:

- Building a Secure RedHat Apache Server HOWTO (www.tldp.org/HOWTO/SSL-RedHat-HOWTO.html)
- SSL Certificates HOWTO (www.tldp.org/HOWTO/SSL-Certificates-HOWTO/index.html)
- OpenSSL Web site (www.openssl.org)

Understanding SSL and Server Certificates

It isn't necessary to go into the gory details of SSL encryption to understand how SSL and server certificates work together to create a secure Web server. SSL uses key pairs to encrypt and decrypt data. One key is public, accessible to everyone; the other key is private, so only you or another authorized person can access it. Either key can be used to encrypt or decrypt data. The public key is part of the certificate, which is how the certificate is used to verify data sent to and received from the server.

If a key is public, if (theoretically) everyone knows the public key, how can it be used for secure communication? The idea is remarkably simple. Data encrypted with the public key can be decrypted only with the private key, which only you know. So, anyone can send you data encrypted with the public key but only you will be able to decrypt it because only you know the private key. Likewise, data encrypted with your private key can be decrypted only by the public key. If only you know the private key, recipients of encrypted data can be confident that a message or other data has come from you and not from someone impersonating you.

Digital certificates work on two simple principles, *encryption* and *trust*:

1. SSL encrypts the communication between a Web server and a Web client to ensure that the data exchange has not been altered during transmission and to make it more difficult to steal sensitive data if the data exchange is intercepted. Encryption increases the difficulty of deciphering a captured data stream. A message digest created from the contents of the data stream serves as a fingerprint, verifying that the data stream hasn't been accidentally or deliberately altered while in transit between the server and the client.
2. Digital certificates provide a certain level of assurance, or trust, that the identities behind a Web server and a Web client are genuine, that is, that a Web server or client is not being operated by an impostor. Depending on the type of certificate in use, a digital certificate issued by a recognized and trusted certificate authority (CA) means that the CA has taken steps to verify the identity of the organization or entity operating a Web site. As a result, a digital certificate provides a reasonable degree of certainty that a Web site is in fact operated by the organization or entity that claims to operate it.

A certificate contains information about the certificate owner, including the following:

- The owner's email address
- The owner's name

- How the certificate can be used
- How long the certificate is valid
- The address of the Web site for which the certificate has been issued
- The public key associated with the certificate
- A message digest (also known as *hash*) to use to confirm that the certificate has not been altered since it was issued

The certificate also contains the certificate ID of the person or entity that issued the certificate and that certified (signed) the information provided in the certificate. Accordingly, you have to trust the issuer of the certificate, the certificate authority (CA). A CA's certificate is referred to as a *root certificate* because it forms the basis, or root, of a tree of trust: if you trust the CA's root certificate, you trust the certificates issued and *signed* by that CA. (Certificates are not valid until they are signed by a CA.) Most browsers come preloaded with the root certificates of several recognized CAs. Figure 23-5 shows the root certificates preloaded in Firefox. To view this list, start Firefox and select Edit ⇨ Preferences ⇨ Advanced, click the Manage Certificates button, and then click the Authorities tab.

As you can see, there are quite a few root certificates. You can also import new certificates, a capability you will need when you create a self-signed certificate. (See the section titled "Creating a Self-Signed Certificate.")

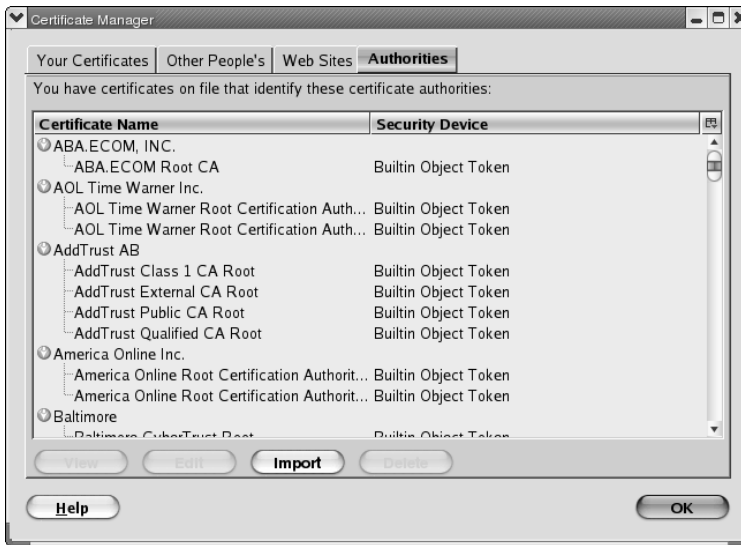


Figure 23-5 Viewing Firefox's preloaded root certificates.

Digital certificates come in two flavors: those signed by CAs and self-signed certificates. A self-signed certificate is one that has been signed using the certificate itself. The primary use of self-signed certificates is to create secure servers that do not participate in the larger trust trees provided by formal CAs. For example, you might use a self-signed certificate on an intranet Web server that handles payroll processing or sensitive personnel information. As a rule, you don't use self-signed certificates on publicly accessible Web sites because most Web browsers won't automatically accept them and also because self-signed certificates do not provide the necessary level of trust.

Creating a Self-Signed Certificate

Creating a self-signed digital certificate on Fedora Core and RHEL systems is simple and straightforward. Use the following procedure (as the root user):

1. Change directories to `/etc/pki/tls/certs`:

```
# cd /etc/pki/tls/certs
```

2. Create a key pair:

```
# make genkey
umask 77 ; \
/usr/bin/openssl genrsa -des3 1024 >
/etc/pki/tls/private/localhost.key
Generating RSA private key, 1024 bit long modulus
.....++++++
.....++++++
e is 65537 (0x10001)
Enter pass phrase:
Verifying - Enter pass phrase:
```

This step creates public and private keys that will be used to encrypt and decrypt data streams sent to and from the server. Use a pass phrase that will be easy to remember but difficult for others to guess. The generated key file will be `/etc/pki/tls/private/localhost.key`.

3. If you are going to obtain a digital certification from a CA, you need to create a certificate signing request (this step is optional for self-signed certificates):

```
# make certreq
umask 77 ; \
/usr/bin/openssl req -new -key /etc/pki/tls/private/localhost.key -
out /etc/pki/tls/certs/localhost.csr
Enter pass phrase for /etc/pki/tls/private/localhost.key:
You are about to be asked to enter information that will be
incorporated
```

into your certificate request.

What you are about to enter is what is called a Distinguished Name or a DN.

There are quite a few fields but you can leave some blank

For some fields there will be a default value,

If you enter '.', the field will be left blank.

Country Name (2 letter code) [GB]:**US**

State or Province Name (full name) [Berkshire]:**Pennsylvania**

Locality Name (eg, city) [Newbury]:**Pittsburgh**

Organization Name (eg, company) [My Company Ltd]:**Possum Holler Enterprises**

Organizational Unit Name (eg, section) []:**Coondog Division**

Common Name (eg, your name or your server's hostname)

[]):**webbeast.example.com**

Email Address []:**webdude@example.com**

Please enter the following 'extra' attributes
to be sent with your certificate request

A challenge password []:**bubba**

An optional company name []:

The resulting file will be created in `/etc/pki/tls/certs/localhost.csr`. A *certificate signing request* (CSR) is used to send a request to a CA to generate a CA-signed certificate for you to install on your Web server. Its contents include the locality domain name information you entered, server administrator's email address, and the public key you created in Step 2. Again, this step is unnecessary if you will only be using a self-signed certificate.

4. Create the self-signed certificate:

```
# make testcert
```

```
umask 77 ; \
```

```
/usr/bin/openssl req -new -key etc/pki/tls/private/localhost.key -out  
/etc/pki/tls/certs/localhost.crt
```

Enter pass phrase for `/etc/pki/tls/private/localhost.key`:

You are about to be asked to enter information that will be
incorporated

into your certificate request.

What you are about to enter is what is called a Distinguished Name or a DN.

There are quite a few fields but you can leave some blank

For some fields there will be a default value,

If you enter '.', the field will be left blank.

Country Name (2 letter code) [GB]:**US**

```

State or Province Name (full name) [Berkshire]:Pennsylvania
Locality Name (eg, city) [Newbury]:Pittsburgh
Organization Name (eg, company) [My Company Ltd]:Possum Holler
Enterprises
Organizational Unit Name (eg, section) []:Coondog Division
Common Name (eg, your name or your server's hostname)
[]:webbeast.example.com
Email Address []:webdude@example.com

```

You can find the newly created certificate in `/etc/pki/tls/certs/localhost.crt`. The Apache configuration looks for the certificate in this location, so there is no need to install the certificate. Nonetheless, it is helpful to know where the certificate is stored. Be sure to make a copy of the key file and the certificate file and store them in a secure location.

At this point, the secure server has been configured, and you are ready to test it. First, restart Apache so it will load the certificate:

```

# service httpd restart
Starting httpd: Apache/2.0.52 mod_ssl/2.0.52 (Pass Phrase Dialog)
Some of your private key files are encrypted for security reasons.
In order to read them you have to provide the pass phrases.

Server luther.kurtwerks.com:443 (RSA)
Enter pass phrase:

OK: Pass Phrase Dialog successful.

```

[OK]

When prompted to enter a pass phrase, enter the pass phrase you used in Step 2. This step is necessary because the key file has been encrypted. You will learn how to skip this dialog box shortly. After restarting the server, point your Web browser at `https://localhost` if you are running the server locally or `https://www.example.com` if you are accessing a remote server. Notice that you need to use the `https` URI to connect to the secure server, which operates on port 443 rather than port 80.

When you first connect, you will see the dialog box shown in Figure 23-6.

This dialog box indicates that the Web server has presented an unverified server certificate. In this case, the “problem” is that the certificate is self-signed, and the browser does not recognize the certificate signer (you) as a valid CA. If you are curious, click the Examine Certificate button to see what the certificate looks like. (See Figure 23-7.)

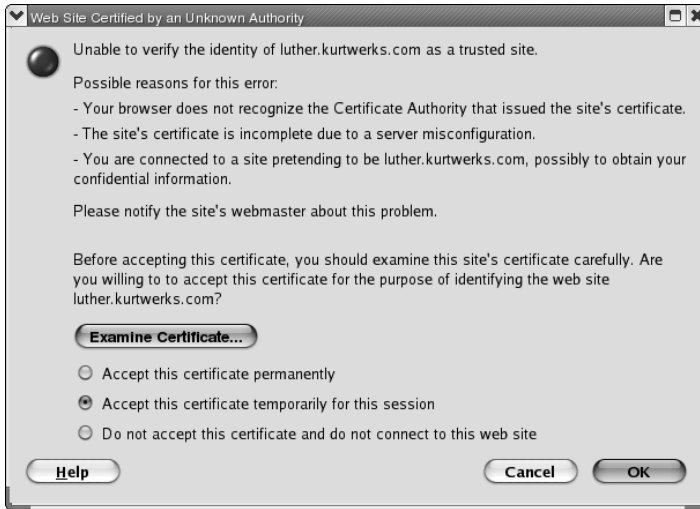


Figure 23-6 Viewing a certificate verification warning.

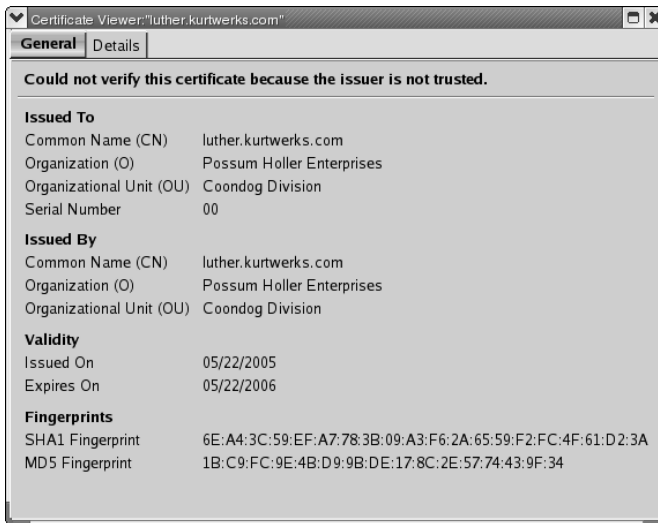


Figure 23-7 Viewing the self-signed certificate.

Otherwise, since you know you can trust this particular certificate, check the Accept this certificate permanently radio button and click OK. If you accept the certificate permanently and are using Firefox, it will import the certificate you can view it later using Firefox's Certificate Manager on Web Sites tab, as shown in Figure 23-8.



Figure 23-8 Viewing the self-signed certificate in Firefox's Certificate Manager.

Figure 23-9 shows the default home Fedora Core home page, as served through the secure server.

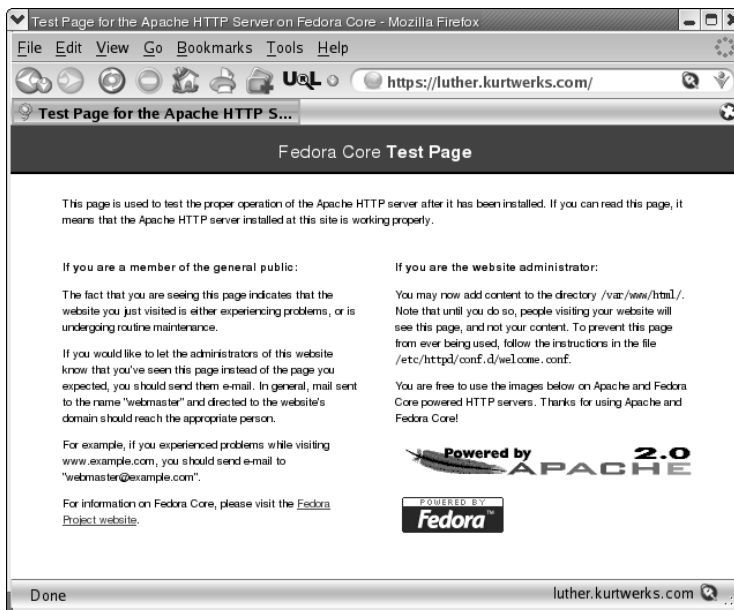


Figure 23-9 Viewing the default Fedora Core home page served via SSL.

The only significant difference between the standard and secure home page is that most browsers usually provide some sort of visual indication that you are connected to a secure page. Firefox, for example, shows a padlock icon in the lower-right corner of the screen.

Obtaining a Certificate from a Certification Authority

To obtain a digital certificate from a recognized CA, you must create a CSR, as described in the previous section, and submit it to a CA. You also have to pay for the certificate. You can choose from a number of CAs, some of which are shown in the following list. The list is not complete, but should provide a starting point for you:

- Cybertrust (betrusted.com/products/ssl/shop/index.asp)
- Entrust (entrust.com/certificate_services/)
- GeoTrust (geotrust.com/web_security/index.htm)
- GlobalSign (globalsign.net/digital_certificate/serversign/index.cfm)
- GoDaddy (godaddyssl.com)
- Thawte Consulting (.thawte.com/ssl123)
- Verisign (verisign.com/products-services/security-services/ssl/buy-ssl-certificates/index.html)

If you would like to use a free or open source CA, the two best known are:

- CAcert (cacert.org)
- StartCom Free SSL Certificate Project (<http://cert.startcom.org>)

Each CA has its own procedures, requirements, and fees for issuing signed certificates, so it isn't possible to describe them in this space.

Summary

This chapter introduced you to Apache Web server configuration. The stock Fedora Core and RHEL configuration files served as the basis to review some of Apache's configuration options. Most Web servers serve dynamic content, so you also need to know how to enable and test Apache's support SSI, CGI, and PHP. Creating a secure server is a vital step if your Web site will be used for electronic commerce or to conduct any type of transaction that must be secured, so you also learned how to create a secure server using SSL. The features you learned in this chapter provide the basis for most of the Web services you learn how to set up in Chapter 24.

Providing Web Services

IN THIS CHAPTER

- Creating Mailing Lists
- Setting Up Web-Based Email
- Configuring an RSS Feed
- Adding Search Functionality

Users of Internet-based services are considerably more sophisticated (or perhaps more demanding) today than they used to be. It is rarely sufficient to put up a Web server that “merely” displays information. Depending on the enterprise or the Web site, users might want mailing lists or online forums so that they can kibbutz with other users. Another typical request (or requirement) is for some sort of Web-based access to email, especially for remote users or road warriors. News sites and community-oriented Web sites (think Slashdot or Groklaw) often provide RSS feeds to enable people to track the latest news without having to visit the Web site constantly. Few Web sites larger than 10 pages or so can get by without robust search features. This chapter shows you how to provide all of these services using packages that are Fedora Core- or RHEL-ready.

Creating Mailing Lists

Mailing lists are an easy, low-maintenance way to allow people who share a common interest or goal to communicate with each other. One of the most popular mailing list managers right now is Mailman, the GNU Mailing List Manager. There are several reasons for its popularity, but perhaps the chief reason is that, aside from its initial installation and setup, Mailman can be

administered using a browser-based interface, which makes it ideal for use by groups whose members or participants are geographically dispersed. Mailman is also rich in built-in functionality that other mailing list manager software (such as the venerable Majordomo) requires add-in software to support, such as:

- Automatic bounce processing
- Automatic message archiving and hooks for third-party archival solutions
- Web-based message archive access
- Content filtering
- Digest creation, maintenance, and delivery
- Excellent tools for individual and mass membership management
- Integrated Usenet gateway
- Intelligent detection of autoresponse message loops
- Password-based authentication
- Passwordless handling of certain user tasks
- Per-list and per-user-per-list configurability
- Spam filtering
- Strong moderation and privacy controls
- Subscription blacklists
- Support for approximately two dozen languages in its Web pages and email notices

You can find out more about Mailman by reading the documentation installed in `/usr/share/doc/mailman-2.1.5`, the README files in `/usr/lib/mailman`, and by visiting the project Web site at <http://www.list.org>.

Completing the Initial Mailman Configuration

The obvious prerequisite for Mailman configuration is having the software installed. The following command will show you if Mailman is installed:

```
$ rpmquery mailman
mailman-2.1.5-35.fc4
```

If you don't see similar output, install Mailman before proceeding. Otherwise, the following procedure walks you through completing the initial configuration:

1. Edit `/etc/httpd/conf.d/mailman.conf`, setting the domain for the Web server appropriately. At the bottom of this file, you'll find a commented-out line that reads:

```
# RedirectMatch ^/mailman[/]*$
http://www.example.com/mailman/listinfo
```

Uncomment it and change `www.example.com` to the domain for your Web server. For example:

```
RedirectMatch ^/mailman[/]*$ http://www.wiley.com/mailman/listinfo
```

2. Restart Apache:

```
# service httpd restart
```

3. Create the site password:

```
# /usr/lib/mailman/bin/mmsitepass sekritword
Password changed.
```

4. Optionally, create a password for the site moderator:

```
# /usr/lib/mailman/bin/mmsitepass -c sekritword
Password changed.
```

The *site moderator* is usually someone other than the system administrator who has the authority to create, manage, and delete mailing lists and administer the Mailman installation, while not having any other system administrative rights or responsibilities. You don't have to have a site moderator, and at small sites you probably won't have the personnel to assign a dedicated site moderator.

5. If your mail host and Web server do not reside on the same system, edit `/usr/lib/mailman/Mailman/mm_cfg.py` and set `DEFAULT_EMAIL_HOST` and `DEFAULT_URL_HOST` appropriately. This example assumes that the Web and mail servers are the same system. If you need to edit this file, however, be sure to make a copy of it before editing it just in case something goes wrong.
6. Create a mailing list named `mailman`. This list will send out password reminders and other list manager messages and follow the prompts to complete the list creation process:

```
# /usr/lib/mailman/bin/newlist mailman
Enter the email of the person running the list: listdude@example.com
Initial mailman password:
To finish creating your mailing list, you must edit your /etc/aliases
(or
equivalent) file by adding the following lines, and possibly running
the 'newaliases' program:
```

```
## mailman mailing list
mailman:          "|/usr/lib/mailman/mail/mailman post mailman"
mailman-admin:    "|/usr/lib/mailman/mail/mailman admin mailman"
mailman-bounces:  "|/usr/lib/mailman/mail/mailman bounces mailman"
mailman-confirm:  "|/usr/lib/mailman/mail/mailman confirm mailman"
mailman-join:     "|/usr/lib/mailman/mail/mailman join mailman"
mailman-leave:    "|/usr/lib/mailman/mail/mailman leave mailman"
mailman-owner:    "|/usr/lib/mailman/mail/mailman owner mailman"
mailman-request:  "|/usr/lib/mailman/mail/mailman request mailman"
mailman-subscribe: "|/usr/lib/mailman/mail/mailman subscribe mailman"
an"
mailman-unsubscribe: "|/usr/lib/mailman/mail/mailman unsubscribe mailman"
```

Hit enter to notify mailman owner...

The display wraps due to this book's formatting constraints.

7. Edit `/etc/aliases` and add the aliases that `newlist` displayed in Step 6. Mailman uses these aliases to route email for a specific list to the subscribed members. You will also need to comment out the two aliases that already exist for mailman. They are near the bottom of `/etc/aliases` and look like the following:

```
# mailman aliases
mailman:          postmaster
mailman-owner:    mailman
```

8. After you've added the aliases, uses the `newaliases` utility to update the binary aliases database.

```
# newaliases
```

9. Restart your MTA. If you are using Sendmail, use the following command:

```
# service sendmail restart
```

If you are using Postfix, the command to use is:

```
# service postfix restart
```

10. Start the mailman service:

```
# service mailman start
```

The mailman service starts Mailman's `qrunner` daemon, which handles the bulk of the mailing list processing duties.

11. If you want Mailman to start at boot time, use the `chkconfig` utility to configure the boot scripts accordingly:

```
# chkconfig --levels 0123456 mailman off
# chkconfig --levels 345 mailman on
```

With the initial Mailman configuration complete, you are ready to create a mailing list and test the installation.

Creating a Mailing List

To create a mailing list using the browser-based interface, point a Web browser to the URL `http://www.example.com/mailman/create/`, replacing `www.example.com` with the server on which Mailman is running. You should see a screen resembling Figure 24-1.

When you have filled in the various fields, click the Create List button at the bottom of the page to create the list. If the attempt succeeds, you should see a results page resembling Figure 24-2.

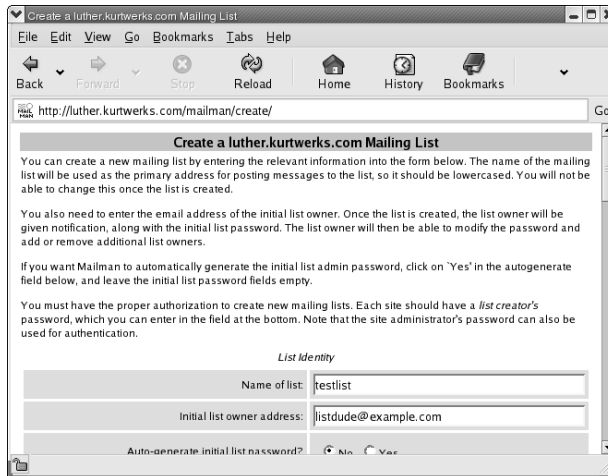


Figure 24-1 Viewing Mailman's list creation interface.

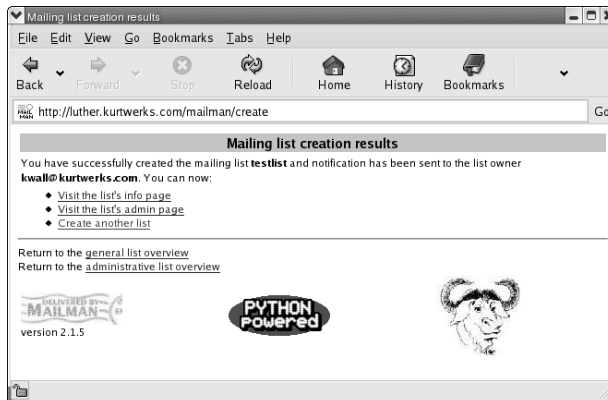


Figure 24-2 Viewing Mailman's list creation results page.

Modifying a Mailing List's Configuration

To modify the configuration of mailing list, open the URL `www.example.com/mailman/admin/listname`, replacing `listname` with the name of list you want to configure. For example, to change the configuration of the `testlist` mailing list created in the previous section, go to `www.example.com/mailman/admin/testlist`. After you enter the list administrator's password, the administration screen you see should resemble Figure 24-3.

Mailman's mailing list configuration is divided into a number of categories that make it possible to configure the behavior and characteristics of a mailing list at a remarkably fine-grained level. Table 24-1 describes each category.

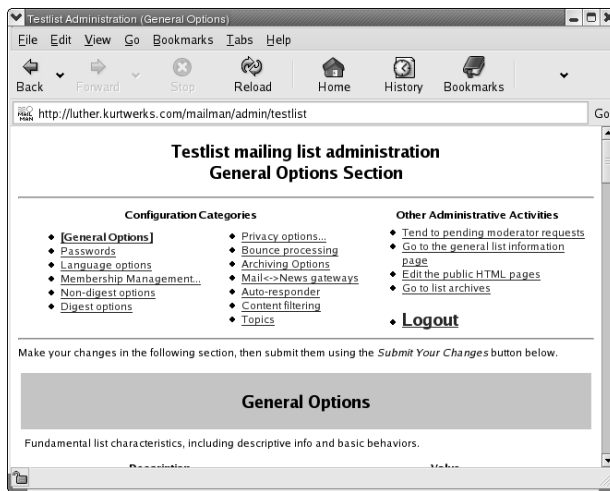


Figure 24-3 Viewing the testlist administrative interface.

Table 24-1 Mailman List Configuration Categories

CATEGORY	DESCRIPTION
General Options	Controls how the list interacts with the Web server, how users interact with the list, and overall list characteristics
Passwords	Changes list moderator and Mailman administrator's passwords
Language options	Defines the languages the list supports
Membership Management	Supports adding, removing, and modifying users and customizing user settings (individually and en masse)

Table 24-1 (continued)

CATEGORY	DESCRIPTION
Nondigest options	Sets policies for all mailing list traffic that is delivered immediately
Digest options	Sets policies for all mailing list traffic that is delivered in digest or batch form
Privacy options	Modifies exposure of the list's membership and whether the list is public
Bounce processing	Controls how Mailman processes bounced messages for this list
Archiving Options	Changes list's archive creation parameters
Mail<->News gateways	Defines the list's gateway services (if any) for mail-to-news and news-to-mail
Auto-responder	Sets policies for the list's autoresponses and how to handle autoresponders from list members
Content filtering	Manages the list's content filters for accepted/rejected messages
Topics	Enables topic filtering, which creates topic keywords based on a set of regular expressions

There isn't space to go into all the configuration options. Almost all the options have associated help text accessible by clicking a hyperlinked term. For example, on the General Options page shown in Figure 24-3, clicking the Details for `real_name` hyperlink opens a configuration help screen that explains the details and implications of changing the list's real name. Feel free to click around and read the help text. We won't insult you by repeating what you can read for yourself. Keep in mind that the settings you configure on these screens apply to only a single list. Configuration parameters that apply to one list do not *necessarily* apply to another unless the Mailman administrator (not a list or site moderator) changes the defaults for *all* Mailman-served mailing lists.

Performing Common Mailman Administrative Tasks

On a freshly installed Mailman site or with a newly create list, there are a number of small tasks that most administrators might want or need to do. Key tasks include the following:

- Presubscribing a list of people
- Hiding a list from casual browsers of the mailman interface
- Restricting archives access to group members

Mailman's browser-based interface makes it ridiculously simple to perform all of these tasks. We'll show you how to perform some of them in the following subsections.

TIP The general form of the URL for getting to any specific Mailman list administration page is `www.example.com/mailman/admin/listname`, where **listname** is the name of the mailing list. Likewise, for general information about a given Mailman-hosted mailing list, the general form of the URL is `www.example.com/mailman/listinfo/listname`.

Adding Multiple List Members

After creating a mailing list, you often want to subscribe a multiple preselected people at once. Mailman's interface for this feature resides in the Membership Management configuration category. From your mailing list's main administration page, click Membership Management ⇄ Mass Subscription. On the resulting page, add the email addresses in the top text area, one address per line.

To subscribe these email addresses immediately, make sure that the Subscribe radio button is selected. It's usually good form to let people know when they've been subscribed to a mailing list, so make sure the Send Welcome messages to new subscribers radio button is also set to yes. When you're done adding email addresses, click the Submit Your Changes button.

Hiding a Mailing List

In the context of Mailman, *hiding a mailing list* means to keep a mailing list from showing up in the list of advertised mailing lists when a Web site visitor visits `www.example.com/mailman/listinfo`. To hide a mailing list, click Privacy Options ⇄ Subscription Rules from a mailing list's main administration page. On the resulting page, select the No radio button next to Advertise this list when people ask what lists are on this machine?

As a general rule, especially for publicly accessible mailing lists, you should also set the following two options:

- Who can view subscription list? — List admin only
- Show member addresses so they're not directly recognizable as email addresses? — Yes

Hiding a list in this way does not prevent people from subscribing or finding it by accident. Rather, it simply prevents showing the list on the site's general list information pages.

Restricting Archives Access

If you don't want nonmembers to be able to access your email archives, Mailman makes it easy to do. One common reason to restrict archive access is to foil the attempts of spambots to harvest email addresses by crawling your Web site, which provides the archive access. To block access, from a list's primary administrative page, click Archiving Options and select the private radio button next to Is archive file source for public or private archival? Click Submit Your Changes to save your changes.

Setting Up Web-Based Email

For road warriors and remote employees that cannot access LAN-based services using VNC or VPN, perhaps the most valuable service you can provide is Web-based or browser-based email access. Fedora Core and RHEL ship with SquirrelMail, a popular and capable browser-based email package that uses IMAP for mail handling, Apache for Web services, and PHP to glue everything together. As with Mailman, most of the work has already been done for you after you install the SquirrelMail RPM. To see if it is installed, use the following command:

```
# rpmquery squirrelmail
squirrelmail-1.4.4-2
```

If SquirrelMail isn't installed, install it before proceeding.

As you might suspect, you need Apache and an IMAP server installed to use SquirrelMail. Refer to Chapter 23 if you haven't already gotten Apache installed and running. Pay particular attention to the section that describes verifying that PHP works, because SquirrelMail uses PHP to create the Web mail interface. If you don't have an IMAP server installed, or don't know if you do, please refer to Chapter 21, which describes how to configure an email server and how to set up IMAP services. Make sure that Apache, your email server, and your IMAP server are running before proceeding. The discussion that follows assumes that you are using the Postfix MTA and that you are using the Cyrus IMAP daemon that is part of the standard Fedora Core and RHEL installation.

Connecting to SquirrelMail

To connect to the Web mail server, browse to `http://localhost/webmail/`. If necessary, replace `localhost` with the name of the Web server you will be using to provide access. (You do not have to run SquirrelMail on the same system that hosts the IMAP server.) You should see a page that resembles Figure 24-4.

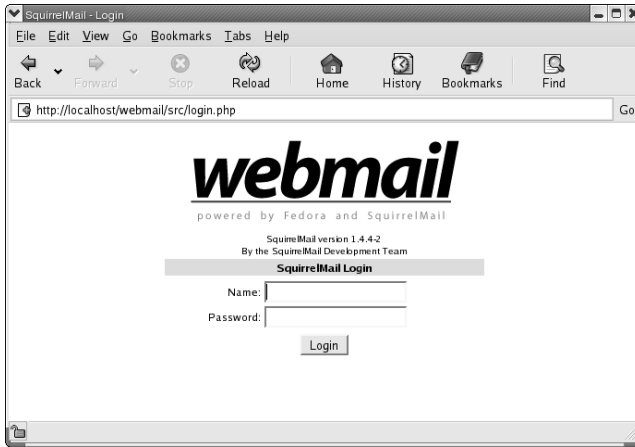


Figure 24-4 Viewing the SquirrelMail login screen.

Login using a user account you know has email. The user's home SquirrelMail page should resemble Figure 24-5.

If your IMAP server does not reside on the same system and the Web server hosting SquirrelMail, edit SquirrelMail's configuration file, `/usr/share/squirrelmail/config/config.php`, and modify the line that begins with `$imapServerAddress` (near line 29). By default, it reads

```
$imapServerAddress = 'localhost';
```

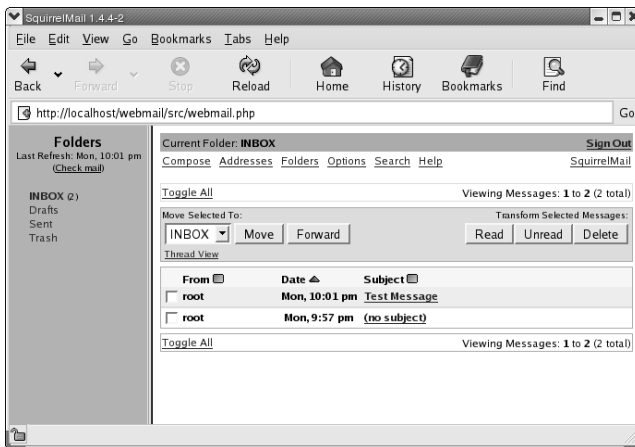


Figure 24-5 Viewing a user's SquirrelMail home page.

Change 'localhost' to reflect the name of the IMAP server. For example, if the IMAP server is named `imap.example.com`, the modified line should read:

```
$imapServerAddress = 'imap.example.com';
```

Reload Apache's configuration to make the change take effect:

```
# service httpd reload
```

Reconfiguring SquirrelMail

SquirrelMail provides an alternate (and perhaps preferable) method for configuring its behavior — a Perl-based configuration script named `/usr/share/squirrelmail/config/config.conf.pl`. The configuration script allows you to reconfigure SquirrelMail's behavior and default settings. It is a command line-based script (no pretty GUI), but it is easy to use and spares you having to dig around in the bowels of SquirrelMail's PHP code. To start the configuration tool, execute the script:

```
# /usr/share/squirrelmail/config/conf.pl
```

The resulting screen should resemble Figure 24-6.

TIP If you can't see what you're typing when you first start the SquirrelMail configuration tool, type `c` or `C` and press Enter to turn off the color display.

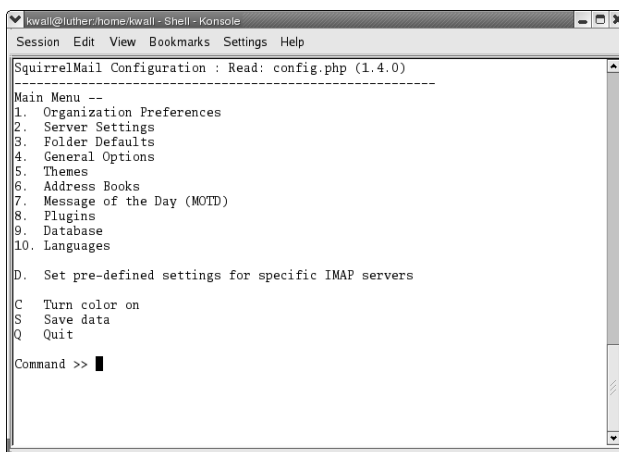


Figure 24-6 Viewing the SquirrelMail configuration tool's main menu.

The key groups of settings you might want to change are the Organization Preferences, which enable you to modify SquirrelMail's overall appearance; Server Settings, which is where you set the IMAP server name and the default domain name to use; and Folder Defaults, which define the names, locations, and behavior of the standard IMAP mail folders.

To select a menu item, type the number or letter next to the item you want to change and press Enter. So, to change Server Settings, type **2** and press Enter. To change the IMAP server SquirrelMail uses, type **a** or **A**, press Enter, and then type **4** and press Enter. At the prompt (see Figure 24-7), type the name of the IMAP server you want to use and then press Enter.

At (almost) any time, you can type **s** or **S** and press Enter (on most screens) to save your changes immediately or type **r** or **R** and press Enter to return to the main menu. If you prefer, you can make all of your changes and then type **q** or **Q** to exit, at which point the configuration tool prompts you to save you changes. If you don't save your changes at this point, any changes that have not already been saved will be discarded.

Many of the settings you make using SquirrelMail's configuration tool are global and cannot be changed. The color scheme, however, what SquirrelMail calls a *theme*, and the default stylesheet (as defined by a CSS file) can be set globally but overridden on a per-user basis.

If you make any significant configuration change, especially to server-related settings, you should view the configuration test page to make sure that the changes worked and had the desired effect. The configuration test page can significantly simplify tracking down configuration mistakes and conveniently includes a login hyperlink at the bottom. The test page is <http://localhost/webmail/src/configtest.php>. Figure 24-8 shows a sample page.

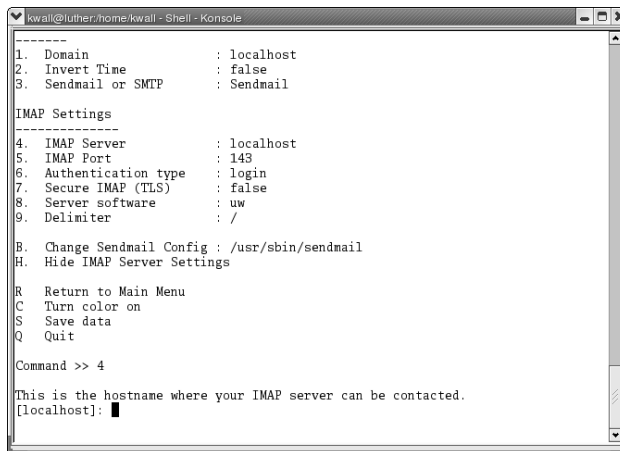


Figure 24-7 Changing the IMAP server SquirrelMail uses.

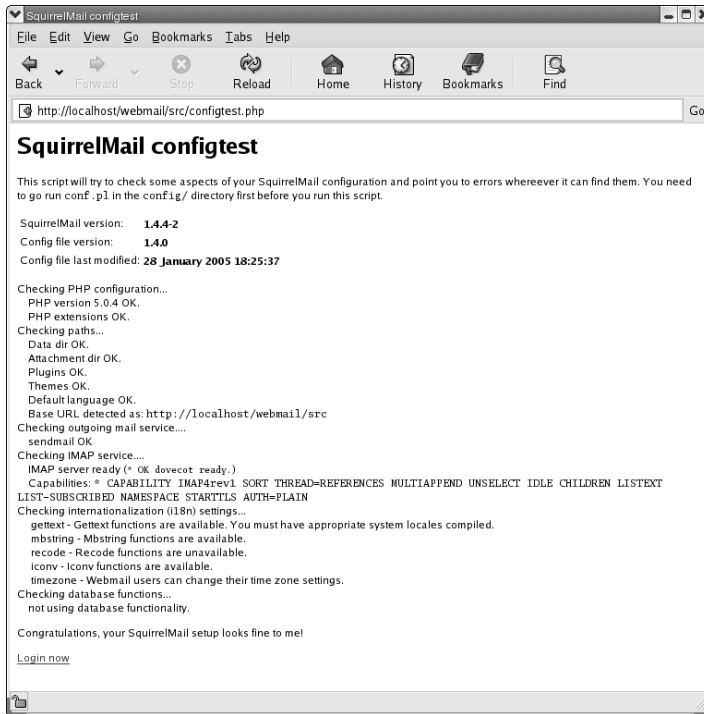


Figure 24-8 Testing SquirrelMail configuration changes.

SquirrelMail's interface provides all the features you would expect in a browser-based email client and should keep your mobile users happy. If you need more information about SquirrelMail, visit its project page on the Internet, www.squirrelmail.org.

Configuring an RSS Feed

What's an RSS feed? RSS is an acronym for *Really Simply Syndication*, *Rich Site Summary*, or *RDF Site Summary*, depending on which version of the RSS specification you follow. Regardless of the version you use, RSS defines and implements an XML format for distributing news headlines over the Web, a process known as *syndication*. To express it more simply and generally, RSS makes it possible to distribute a variety of summary information across the Web in a news-headline style format. The headline information includes a URL that links to more information. That URL, naturally, brings people to your Web site. By way of example, Figure 24-9 shows the BBC's front page RSS news.

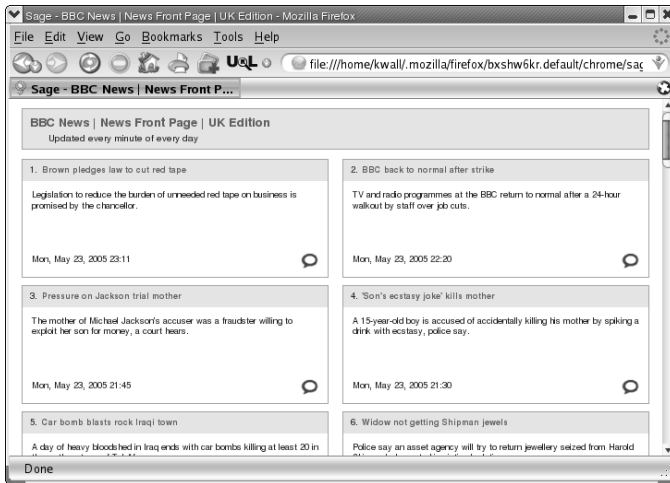


Figure 24-9 Viewing one of the BBC's RSS feeds.

The canonical use of RSS is to provide news headlines in a compact format. Most major news sites provide this type of summary information. Some open-source software projects use RSS to inform subscribers of significant events occurring in the project, such as releases, updates, and meetings. Popular blog sites use RSS to notify people of the latest blog entries. If you or your users have any sort of activity to publicize or frequently updated information to distribute, one way to do so is to provide an RSS feed on your Web site. This section shows you how to set up a simple, no-frills RSS feed that follows the 0.91 RSS specification. (See the sidebar “Sorting out RSS Versions” for a discussion of the competing RSS versions.)

NOTE For more information about RSS and RDF, see the home pages for the **original RSS specification**, <http://purl.org/rss/> and the **W3C RDF activity pages** at <http://www.w3.org/RDF/>.

If you're providing an RSS feed, you might be curious how your Web site visitors might use it. Many people track news and other RSS feeds using an RSS aggregator. An *aggregator* is an application or browser extension that collects (or *aggregates*) RSS feeds from a list of sites that you specify and presents all of them in a single interface. Most aggregators can understand both plain vanilla RSS feeds and the more feature-rich Atom feeds. One of our favorite feed aggregators is the Firefox extension called Sage (see Figure 24-10).

SORTING OUT RSS VERSIONS

There are different versions of the RSS specifications, at this point in time, three versions, 0.9x, 1.0, and 2.0. The original version, RSS 0.91, was designed by Netscape and UserLand Software's Dave Winer. The current iteration of the 0.9x specification is 0.94. The 0.9x spec is the simplest to understand and the easiest to use, so it is generally referred to as *Really Simple Syndication*. Dave Winer maintains control of this version of the RSS specification.

RSS 1.0, referred to as *RDF Site Summary*, where RDF stands for *Resource Description Framework*, is a version of RSS promoted by the W3C. It is not necessarily an improvement over RSS 0.9x. Rather, it is a version of RSS that can be parsed by any reader that understands RDF. Accordingly, any RDF-capable reader can handle an RSS 1.0 feed without having to understand anything about RSS itself. Unfortunately, proponents of the simpler 0.9x specification and the more standardized 1.0 specification were unable to come to a compromise, which resulted in the 1.0 branch morphing into a version known as Atom.

Meanwhile, in reaction to the emergence of Atom, proponents of the 0.9x branch started working on RSS 2.0. RSS 2.0 is the successor to RSS 0.9x. Like 0.9x, RSS 2.0 development is led by Dave Winer but, partially in response to criticism that he owned the copyright on RSS 0.9x, Winer donated copyright on 2.0 to Harvard University and removed himself as the final judge of RSS 2.0 extensions or usage.

As the matter stands, then, you can write Atom-compliant RSS feeds or 0.9x/2.0-compliant feeds. Choosing which one is likely to come down to a matter of what your users want and whether you prefer the simplicity of the 0.9x/2.0 branch or the alleged "standards compliance" of the Atom branch.

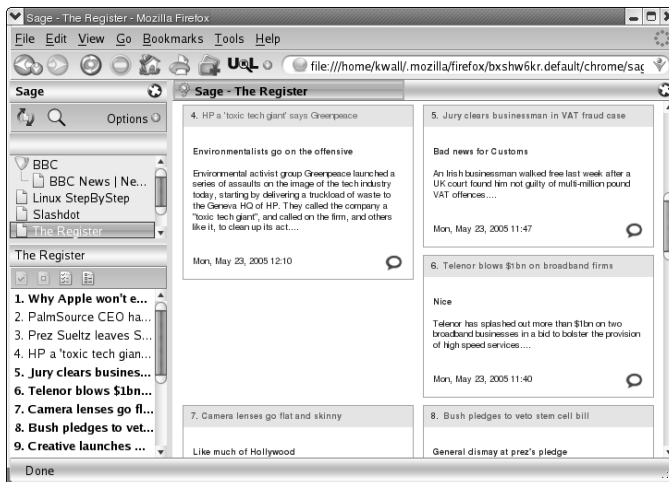


Figure 24-10 Using the Sage RSS aggregator in Firefox.

On the right side of the browser screen, Sage shows article summaries. You can click these summaries to view the entire article. Notice that the left side of the screen contains the Sage sidebar. The sidebar is always present (unless you close it), which makes it easy to jump to the news or other RSS feed item that interests you. The upper portion of the sidebar lists each individual feed that you track. The lower portion of the Sage sidebar lists each individual feed item available from the feed that is currently selected in the upper portion of the sidebar. For example, in Figure 24-10, the selected feed is from The Register, which had 15 different feed headlines. Clicking a feed headline in the list loads it into the browser window on the right.

Selecting Content for an RSS Feed

What kind of content might be appropriate to include in an RSS feed? Structurally, any sort of list-oriented information, that is, information that can be organized into a list of hyperlinks *and* that contains information people will likely find useful are potential candidates for inclusion in an RSS feed. In terms of content, you might include the following types of information:

- News and announcements about products, events, press releases, or whitepapers
- If your Web site (rather, the Web site you maintain) frequently updates documents, you might consider providing an RSS feed that lists new or updated documents (or individual pages)
- Calendars of events, such as company appearances at trade shows, user group meetings, or listings of training sessions
- Listings of available jobs

As a final suggestion, RSS feeds can be even more useful on a company intranet than they are on an extranet. For example, a human relations department might use an RSS feed to notify people of new or updated personnel forms. A payroll team might use an RSS feed to let people know when paychecks can be picked up or to remind employees to fill out important paperwork.

Creating the Feed File

Listing 24-1 shows a minimal RSS feed file. You can type this as a model for your own file, type in the listing yourself, or use the included `feed.rss` file from this chapter's code directory on the CD-ROM.

```

<?xml version="1.0"?>
<rss version="0.91">
  <channel>
    <title>RHLNSA3 Channel</title>
    <link>http://localhost/</link>
    <description>Updates for RHLNSA3</description>
    <language>en-us</language>
    <image>
      <title>RHLNSA3 Channel</title>
      <url>http://localhost/favicon.png</url>
      <link>http://localhost/rhlnsa3/</link>
    </image>
    <item>
      <title>RHLNSA3 News: April 5, 2005</title>
      <link>http://localhost/rhlnsa3/20050405.html</link>
      <description>RSS feeds material nearly complete!</description>
    </item>
  </channel>
</rss>

```

Listing 24-1 A bare-bones RSS feed.

This file contains the key tags you need to create an RSS feed, which Table 24-2 describes. The `<?xml version="1.0"?>` line is required by the XML specification, and it *must* be the first line in the file.

Table 24-2 Minimum Required Elements in an RSS Feed

TAG	DESCRIPTION
channel	Delimits the contents of a single channel
description	Describes the channel or lists the headline for the syndication item
image	Describes an icon or image that represents the channel
item	Delimits a single syndication item
language	Informs readers of the language in which the feed is written
link	Contains a link to the channel home page or an individual syndication item
rss	Defines the content as RSS data and specifies the version (0.91)
title	Identifies the channel or individual syndication item

Required tags are shown in boldface. As an XML file, all of the tags in an RSS file must be terminated with matching `</>` tags (such as `<channel></channel>` and `<link></link>`), and the tags have to be lower case. The version attribute of the `<rss>` tag is required because it enables RSS readers (usually called feed aggregators) to know which version of RSS to support and how to interpret the contents of the RSS file.

The meat of an RSS feed appears in `<item>` tags. The information in a feed item's `<title>` tag serves as the headline, so it should be catchier than the ho-hum `<title>` shown in Listing 24-1. Each item's `<link>` contains the URL of the document containing the full scoop or other content you are trying to publicize using RSS. The text in the `<description>` tag might be shown as text under the headline, as pop-up text that appears if a mouse cursor hovers over the headline link, or it might be totally ignored, depending on the RSS reader in use.

Turning on an RSS Feed

Naturally, Web browsers and feed aggregators need to know that your Web site has an RSS feed and where to find it. To do this, you need to add some metadata to the headers of your Web pages. Use the HTML `<link>` tag to do so. The following code snippet shows a template you can use for HTML pages:

```
<link rel="alternate" type="appliation/rss+xml" href="rssfile.rss"
title="descriptive text">
```

If your Web pages are XHTML, the `<link>` tag must use the implicit end tag marker, as shown in the following snippet:

```
<link rel="alternate" type="appliation/rss+xml" href="rssfile.rss"
title="descriptive text" />
```

Replace `rssfile` and `descriptive text` with the name of your RSS feed file and an appropriate title, respectively. For the RSS feed file shown in Listing 24-1, and for HTML-based pages, you could use the following `<link>` tag:

```
<link rel="alternate" type="application/rss+xml" href="feed.rss"
title="RHLNSA3 Updates">
```

After you have added this text, RSS-capable applications will be aware that you provide an RSS feed. For example, if you load the page containing this text in an RSS-capable Web browser, such as Firefox, and you'll see a small icon in the lower-right corner of the window that signals an RSS feed is available. (See Figure 24-11.)

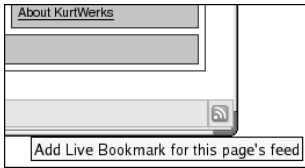


Figure 24-11 Viewing Firefox's icon indicating a Web page has an RSS feed.

Interested readers can see a slightly modified example of this feed in action at <http://www.kurtwerks.com/pubs/rhlnsa3/>.

Creating a simple RSS feed like the one in this section is a relatively low impact activity. It would quickly grow to become a labor-intensive undertaking if you had to do it manually. Fortunately, there are a variety of tools that automate the creation of RSS feeds, and some content management systems even include tools to create RSS feeds automatically. Other tools exist that you can use to validate your RSS feeds for correct format. This section ends with a list of RSS creation and validation tools that you might find useful:

- *Online RSS 0.9x Validator* (<http://aggregator.userland.com/validator/>) checks 0.9x feeds.
- *Online RSS 1.0 Validator* (ldodds.com/rss_validator/1.0/validator.html) checks 1.0 RSS feeds.
- *Orchard RSS* (<http://orchard.sourceforge.net/>) creates feeds using Python, Perl, or C.
- *RSS Editor* (<http://rsseditor.mozdev.org/>) is a Firefox extensions for creating/updating RSS feeds.
- *RSS.py* (mnot.net/python/RSS.py) uses the Python scripting language to generate and parse RSS.
- *XML::RSS* (<http://search.cpan.org/author/EISEN/XML-RSS/>) is Perl module for creating and parsing RSS.
- *xpath2rss* (mnot.net/xpath2rss/) uses XPath expressions to “scrape” Web sites and create RSS feeds.

If you would like additional tutorial information about RSS, see Reuven Lerner's tutorial on RSS syndication, “At the Forge — Syndication with RSS,” which appeared in the print version of *Linux Journal* in September 2004 and is also available on the Web at *Linux Journal's* Web site at www.linuxjournal.com/article/7670. Another excellent tutorial is Mark Nottingham's “RSS Tutorial for Content Publishers and Webmasters,” available on the Web at mnot.net/rss/tutorial/. An excellent book on the subject is *Hacking RSS and Atom*, written by Leslie Orchard (Wiley, ISBN 0-7645-9758-2).

Adding Search Functionality

If you have more than a few pages of content on your Web site, you will need some sort of search capability to help people find the information for which they're looking. While a site map might suffice for small sites, anything larger than a dozen pages or so needs to be searchable. Fedora Core and RHEL ship with the `ht://Dig` search engine installed and ready to go. This section describes how to get it going.

Getting Started with `ht://Dig`

`ht://Dig` is a complete document searching and indexing system designed for a single domain or an intranet. It is not meant to replace the big global search engines like Google, Yahoo!, or Excite. Rather, it is intended for use on single sites and domains and is especially well suited for intranets, primarily because `ht://Dig` was initially developed for campus use at San Diego State University. Although `ht://Dig` is intended for use on a small scale, the word "small" is relative; it is quite capable of searching sites or domains that comprise multiple servers and thousands of documents. `ht://Dig` can handle sites or domains that consist of multiple servers because it has a built-in Web spider that can traverse a site and index all the documents it encounters. `ht://Dig` handles thousands of documents because it uses a static search index that is very fast.

Other `ht://Dig` features include the following:

- **Character set collation** — SGML entities such as `´`; and ISO-Latin-1 characters can be indexed and searched.
- **Content exclusion** — Support for excluding content from indexing using a standard `robots.txt` file, which defines files and filename patterns to exclude from searches.
- **Depth limiting** — Queries can be limited to match only those documents that are given number of links or clicks away from the initial search document.
- **Expiration notification** — Maintainers of documents can be notified when a document expires by placing special meta-information inside an HTML document (using the HTML `<meta>` tag) that `ht://Dig` notices and uses to generate document expiration notices.
- **Fuzzy searching** — `ht://Dig` can perform searches using a number of well-known search algorithms. Algorithms can be combined. The currently supported search methods include the following:
 - *Accent stripping* — Removes diacritical marks from ISO-Latin-1 characters so that, for example, `e`, `ē`, `ě`, `è`, `ę`, and `ě` are considered the same letter (`e`) for search purposes.

- *Exact match* — Returns results containing exact matches for the query term entered.
 - *Metaphones* — Searches for terms that sound like the query term but based on an awareness of the rules of English pronunciation.
 - *Prefixes* — Searches for terms that have a matching prefix, so, for example, searching for the prefix *dia* matches *diameter*, *diacritical*, *dialogue*, *diabolical*, and *diadem*.
 - *Soundex* — Searches for terms that sound like the query term.
 - *Stem searches* — Searches for variants of a search term that use the same root word but different stems.
 - *Substrings* — Searches for terms that begin with a specified substring, so searching for *phon** will match *phone*, *phonetic*, and *phonics* but not *telephone*.
 - *Synonyms* — Searches for words that mean the same thing as the query term, causing *ht://Dig* to perform return results that include synonyms.
- **Keyword optimization** — You can add keywords to HTML documents to assist the search engine using the HTML `<meta>` tag.
 - **Output customization** — Search results can be tailored and customized using HTML templates.
 - **Pattern matching** — You can limit a search to specific parts of the search database by creating a query that returns only those documents whose URLs match a given pattern.
 - **Privacy protection** — A protected server can be indexed by instructing *ht://Dig* to use a given username and password when indexing protected servers or protected areas of public servers.

As you can see, *ht://Dig* is a full-featured search engine. It is also easy to use and maintain, so try it and see if it will meet your needs.

To get started, you need to create the initial search database and then create some customized indexes to facilitate fast searching. Fortunately, you do not have to do this manually. *ht://Dig* uses a script named *rundig* to automate database creation and index maintenance. So, as root, execute the following command:

```
# /usr/bin/rundig
```

rundig works by reading the *ht://Dig* configuration file, */etc/htdig/htdig.conf*, and spidering the site specified by the *start_url* variable. In the stock installation, *start_url* is *http://localhost*. *rundig* finds and

follows each hyperlink specified in HTML files (that is, files with the extensions `.html`, `.htm`, or `.shtml`) that point to documents in the domain it is indexing, creating a database of words in each document and another database of the URLs. Additional steps massage these databases into a searchable format and, optionally, create various indexes for so-called *fuzzy searches*, such as soundex and metaphone searches (searches for words that sound like other words) and prefix matches (searches for words that contain specified prefixes).

If you have a lot of content in files that are not straight HTML, such as text files, files created with SSI, and so forth, `start_url` can also point to a file that contains a list of URLs to check. For example, consider the following `start_url` statement in `/etc/htdig/htdig.conf`:

```
start_url:      http://www.example.com/digme.html
```

This directive tells `ht://Dig` to start indexing at the URL `http://www.example.com/digme.html`. `digme.html` looks like this:

```
<html>
<head>
  <title>Issue Tracker Source Files</title>
  <link rel="stylesheet" type="text/css" media="screen"
        href="/styles/default.css">
</head>
<body>
<a href="5007.txt">5007</a>
<a href="5041.txt">5041</a>
<a href="5042.txt">5042</a>
<a href="5043.txt">5043</a>
<a href="5044.txt">5044</a>
<a href="5045.txt">5045</a>
...
</body>
</html>
```

`ht://Dig` will index the contents of each file linked in `digme.html`.

If you have a lot of content in your Web document trees, the database creation and indexing process can take a while. Otherwise, when the command prompt returns, you can test out the search engine. To do so, point your Web browser at `http://localhost/htdig/` (replace `localhost` with the name of your server if you are not accessing it locally). You should see a screen that resembles Figure 24-12.

Type in a search term (try **documentroot**) and press Enter or click the Search button. The search results should resemble Figure 24-13.



Figure 24-12 Viewing the ht://Dig search page.

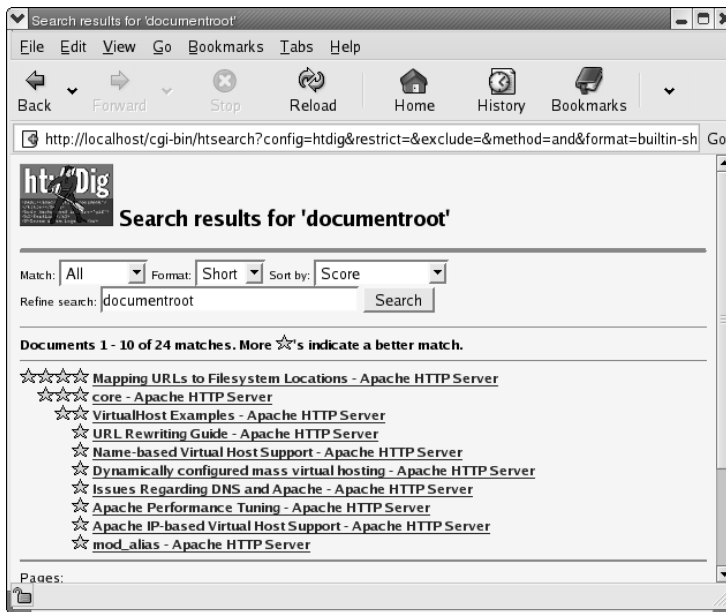


Figure 24-13 ht://Dig search results for the word “documentroot.”

The search results shown in a short format so you can see the number of matches. Notice that the search results are case-insensitive.

After you satisfy yourself that the search engine is working, you will want or need to update the search databases as content is added to the server. The easiest way to accomplish this is to execute `rundig` on a periodic basis using `cron`. How often you update the indexes is up to you, but it should reflect the frequency with which content on the server changes. Listing 24-2 shows a sample script you can use, `rundig.cron`.

```
#!/bin/sh
# rundig.cron - Update ht://Dig search database
/usr/bin/rundig -s
```

Listing 24-2 Cron script to execute `rundig`.

The `-s` option causes `rundig` to display some runtime and database statistics when it is finished. The output might resemble the following:

```
htdig: Run complete
htdig: 1 server seen:
htdig:      localhost:80 2 documents

HTTP statistics
=====
Persistent connections      : Yes
HEAD call before GET       : Yes
Connections opened         : 2
Connections closed         : 1
Changes of server          : 0
HTTP Requests              : 5
HTTP KBytes requested      : 5.68164
```

Make the script executable (**`chmod 755 rundig.cron`**) and place in `/etc/cron.daily` if you want to run it every day; `/etc/cron.weekly` if you want to run it once a week, or `/etc/cron.monthly` if you want to run it on a monthly basis.

After you have a cron job in place to update `ht://Dig`'s database and search indexes, you are done. Check the output of the cron job periodically (it will be mailed to the root user) to make sure that the index is being updated properly. Beyond that, `ht://Dig` takes care of itself, which is just the arrangement a busy system administrator likes.

NOTE For more information about `ht://Dig`, visit its project Web page,

<http://www.htdig.org/>.

Summary

The days in which you could just slap together a simple Web server and cross the “Install Web server” task off your project list are *long* gone. You will likely be asked to add additional features that build on the capabilities provided by your Web server, such as providing mailing list services or creating a browser-based interface for email. GNU Mailman makes it child’s play to provide mailing list services, and SquirrelMail is a popular Fedora Core- and RHEL-ready browser-based email solution. Creating an RSS feed for your Web site is simple to do if you follow the instructions in this chapter. Fedora Core and RHEL also come with ready-to-run Web site search engine features; you just need to know what they are, where to find them, and how to enable them.

Optimizing Internet Services

IN THIS CHAPTER

- Optimizing LDAP Services
- Optimizing DNS Services
- Optimizing Mail Services
- Optimizing FTP Services
- Optimizing Web Services

This chapter offers some optimization techniques you can apply to the servers and services described in the previous chapters. Alas, we can't offer sure-fire, foolproof methods for turning your server into, say, a mail-serving speed daemon. We're fresh out of pixie dust and magic potions. Indeed, listen with a healthy dose of skepticism to anyone who claims to have the One True Optimization Method. Server optimization requires analysis to narrow the problem domain, diagnosis to identify the performance problem, and experimentation to evaluate the effectiveness of your optimization. Naturally, though, it helps to know what kinds of tweaks and changes are most appropriate for a given application or a particular situation. In the case of LDAP, for example, the directory layout can have a dramatic impact on overall LDAP performance. DNS can be optimized by having LAN clients run local caching servers, by configuring multiple slave servers, by directing local lookups to internal servers first, by zone file tweaks, and so on. Mail servers are sensitive to I/O binding, and Web servers respond especially well to additional memory.

In general, anything that improves overall system performance will improve Internet services performance as a side effect. Disabling unnecessary services is a standard technique and hopefully one that you have already implemented. Centralizing Internet services on a machine that is not used by users is another general performance tweak for servers. Also, getting a fatter pipe or simply

using Gigabit Ethernet (GigE) whenever possible will improve Internet server/service performance. Finally, using a higher-performance filesystem on the directories that hold email can substantially improve performance. For example, ext3 does a great job on mail partitions because short-lived mail files might not even need to be created in the filesystem if they pass through the filesystem journal quickly enough. ReiserFS can be a good idea for mail partitions on which hundreds of thousands of tiny files live in the same directories.

Optimizing LDAP Services

Several items in the `slapd` configuration can be tweaked to give better server performance. The items shown in the following list show configuration directives that can be modified for performance reasons:

- **Cache size modification** — You can increase the size of the cache using the `cachesize` directive in `slapd.conf`. For example, the following directive sets the number of LDAP entries stored in the cache to 100,000:

```
cachesize 100000
```

- **Disk subsystem** — As suggested elsewhere in this chapter, replace IDE disks with SCSI disks, and replace or augment SCSI disks with FibreChannel. If SCSI and FibreChannel are too rich for your budget, using Serial ATA (SATA) drives (and a SATA controller, of course) hits a good middle ground because SATA is faster than IDE and less expensive than SCSI and FibreChannel. If you have multiple LDAP data stores, situate each store on its own disk and, if possible, its own dedicated I/O controller in order to minimize I/O contention with other processes.
- **Filesystem tuning** — On filesystems that support it, disable updating file access and modification timestamps, which will decrease the number of file operations that have to be performed by two-thirds. Fewer CPU cycles spent on bookkeeping means more CPU cycles spent doing actual LDAP-related work.
- **Indexing** — With limits, indexes increase performance, but at the cost of additional memory, disk, and CPU usage. Accordingly, don't index data you don't (often) search. By way of guidelines, index only heavily used parts of your schema.
- **Logging** — If you are persuaded that excessive message logging is hampering the performance of your LDAP server, add the following entry to `slapd.conf`:

```
loglevel 0
```

This entry disables logging via the system log.

- **System memory** — In addition to adding more physical memory, increase the size of OpenLDAP's cache to use more RAM.

For more details about these performance tuning tips, have a look at the OpenLDAP FAQ, available on the Web at openldap.org/faq/data/cache/190.html.

Optimizing DNS Services

Optimizing DNS services centers on reducing the latency involved in making DNS queries. For client programs, that is, for applications requesting DNS services, the best all around performance enhancement is to maintain a local cache of DNS information. You get the most bang for your performance buck by reducing the number of DNS queries that have to go to a remote server, even if that server is inside the subnet to which you are connected. The typical approach is to run a caching-only name server on client machines. On the server side, you have a much wider range of options, as discussed in the section titled “Tweaking DNS Servers,” later in this chapter.

Improving the Performance of DNS Clients

To increase the performance (and security) of your caching-only servers on the DNS clients, several options can be modified in the `/etc/named.conf` file created during the installation of BIND. The `/etc/named.conf` file is shown in Listing 25-1.

```
// generated by named-bootconf.pl
options {
    directory "/var/named";
    /*
     * If there is a firewall between you and nameservers you want
     * to talk to, you might need to uncomment the query-source
     * directive below. Previous versions of BIND always asked
     * questions using port 53, but BIND 8.1 uses an unprivileged
     * port by default.
     */
    // query-source address * port 53;
};

//
// a caching only nameserver config
//
controls {
    inet 127.0.0.1 allow { localhost; } keys { rndckey; };
```

Listing 25-1 The `/etc/named.conf` file. (*continued*)


```
};  
zone "." { type hint; file "named.ca"; };  
zone "localhost" {  
    type master;  
    file "localhost.zone";  
    allow-update { none; };  
};  
  
zone "0.0.127.in-addr.arpa" {  
    type master;  
    file "named.local";  
allow-update { none; };  
};  
  
include "/etc/rndc.key";
```

Listing 25-1 *(continued)*

The section of the file in which you are interested is the `options` section. Since this is a caching-only server, you can safely disable functions that are not necessary for this type of server. You can also add options that do apply to a caching-only server.

By default, BIND allows zone transfers to all hosts. However, zone transfers are necessary only for master and slave servers; they aren't necessary for DNS clients. You can disable zone transfers by adding the following line to `/etc/named.conf`:

```
allow-transfer {  
    none;  
};
```

You can also configure your caching-only server to respond to regular queries only from specific hosts. BIND's default setting is to allow queries from any host. Typically, you want to allow queries only from hosts inside your firewall. You can add the following line to the `options` section in `/etc/named.conf`, replacing `www.xxx.yyy.zzz` with your network IP number:

```
allow-query {  
    www.xxx.yyy.zzz;  
    localhost;  
};
```

You can also configure your caching-only server to respond to recursive queries from only specific hosts. BIND's default setting is to allow queries

from any host. Typically, you want to allow queries only from hosts inside of your firewall. You can add the following line to the `options` section of `/etc/named.conf`, again replacing `www.xxx.yyy.zzz` with your network IP number:

```
allow-recursion {  
    www.xxx.yyy.zzz;  
    localhost;  
};
```

Typically, a caching-only server does not have direct access to the Internet, so it creates a cache file to hold dns information. This is the purpose of a caching-only server, and it boosts performance by eliminating the need to send queries to external servers. But if the server does not have the information it needs in its local cache, it needs to send a request to other servers. You can specify the IP address of the servers to which you want to forward requests. Add the following line to the `options` section of `/etc/named.conf`, replacing `www.xxx.yyy.zzz` with the IP address of the name server to which requests should be forwarded:

```
forwarders {  
    www.xxx.yyy.zzz;  
};
```

What happens if the servers you are forwarding to are down? Your server tries to forward the request to other servers. You can prevent this from happening by adding this line:

```
forward only;
```

Tweaking DNS Servers

To increase the performance and security of your master domain server on the DNS clients, several options can be modified in the `/etc/named.conf` file created during the installation of BIND. Make changes to the `options` section of `/etc/named.conf`.

By default, BIND allows zone transfers to all hosts. Zone transfers are necessary only between the master and slave servers, so you can specify the IP address of your slave server by adding the following entry (replace `www.xxx.yyy.zzz` with the IP address of your slave server):

```
allow-transfer {  
    www.xxx.yyy.zzz;  
};
```

You can also configure your master server to respond to regular queries only from specific hosts. The default setting in BIND is to allow queries from any host. Typically, you want to allow other types of queries only from hosts inside your firewall. You can add the following line to the `options` section:

```
allow-query {  
    www.xxx.yyy.zzz;  
    localhost;  
};
```

Replace `www.xxx.yyy.zzz` with your internal network IP number.

You can also configure your caching-only server to respond to recursive queries only from specific hosts. The default setting in BIND is to allow queries from any host. Typically, you want to allow queries only from hosts inside of your firewall. You can add the following line to the `options` section:

```
allow-recursion {  
    www.xxx.yyy.zzz;  
    localhost;  
};
```

Again, replace `www.xxx.yyy.zzz` with your internal network IP number.

Logging

You can configure your caching-only slave and master servers to automatically rotate your `/var/log/named.log` file to prevent your filesystem from filling up with old information. The file `/etc/logrotate.d/named` should have been created during the installation of BIND and should be similar to Listing 25-2.

```
/var/log/named.log {  
    missingok  
    create 0644 named named  
    postrotate  
        /sbin/service named reload 2> /dev/null || true  
    endscript  
}
```

Listing 25-2 The `/etc/logrotate.d/named` file controls log rotation of BIND's log files.

If you do not have `/etc/logrotate.d/named` on your system, create it by copying the file as shown in the listing. Then make the following changes.

Remove this line:

```
create 0644 named named
```

Change the line:

```
/sbin/service named reload 2> /dev/null || true
```

to:

```
/bin/kill -HUP `cat /var/run/named.pid 2> /dev/null` 2> /dev/null ||  
true
```

Be sure to save your changes and restart the named server.

Another tweak you can make is to disable logging for LAME servers. These are servers that appear to be name servers for a domain but are not. This reduces system-resource use. Add the following line to your `/etc/named.conf` file:

```
logging {  
    category lame-servers { null; };  
};
```

Optimizing Mail Services

To improve the speed of your mail services, you can take one of several approaches. Busy sites often use multiple mail servers in order to spread the mail processing load across a number of systems. This reduces the demand on any single system. Another common performance enhancement is to replace Sendmail with another mail server, such as Postfix. If your mail server supports a number of mailing lists, you might consider handling list traffic on one server and regular (nonlist) mail traffic on another server. If you are not in a position to buy a beefier mail server, to buy more mail servers, or to replace Sendmail with Postfix, the section titled “Getting More from Sendmail,” later in this chapter, offers a number of tips and hints to help you squeeze more speed out of it. “Getting More from Postfix,” also later in this chapter, gives you a number of methods you can try to get better performance from Postfix.

Getting More from Sendmail

The `/etc/mail/sendmail.cf` file contains many options that can be tweaked to give better performance and increase the efficiency of your mail server. The most common Sendmail tweak is to change the frequency with which it runs the queue. You can modify this by editing `/etc/sysconfig/sendmail` and changing the line that reads `QUEUE=1h`, which causes Sendmail to process the mail queue every hour, to, for example, `QUEUE=15m`, which runs the queue every

15 minutes. If you modify this file, you have to restart Sendmail. A good source for some performance-tuning tips can be found at the Sendmail Web site at sendmail.org/~ca/email/doc8.12/TUNING.

Getting More from Postfix

If you have a lot of mail that just seems to sit in Postfix's outbound queue, you may be trying to deliver mail to a site that is quite busy. One way to work around this problem is to create a transport map entry for such a site that enables multiple parallel connections and then to give each connection to that site a shorter timeout. Suppose, for example, that you know that `slowsite.com` and `reallyslowsite.net` are busy sites, and you have trouble getting email to them in a timely manner. First, create entries resembling the following in `/etc/postfix/transport`:

```
slowsite.com          deadbeats:
reallyslowsite.net    deadbeats:
```

These entries in `/etc/postfix/transport` add `slowsite.com` and `reallyslowsite.net` to an alias named `deadbeats`. Next, create a corresponding entry in `/etc/postfix/main.cf` that increases the number of simultaneous connections to 50, which allows more mail to `slowsite.com` and `reallyslowsite.net` to be transmitted at once.

```
transport_maps = hash:/etc/postfix/transport
deadbeats_destination_concurrency_limit = 50
```

The key directive here is `deadbeats_destination_concurrency_limit = 50`, which increases the number of parallel connections to `slowsite.com` and `reallyslowsite.net` to 50.

```
deadbeats      unix      -      -      n      -      -      smtp
-o smtp_connect_timeout=5 -o smtp_helo_timeout=5
```

These entries from `/etc/postfix/master.cf` tell Postfix that SMTP connections to sites in the `deadbeats` alias should timeout after 5 seconds and that, similarly, an SMTP transaction must commence within 5 seconds of the HELO command, or the connection will be closed.

If incoming mail seems to queue up while outbound mail gets delivered, then outgoing mail is crowding out incoming mail. Postfix can waste a great deal of time waiting for connections to time out, so, again, the solution is to reduce the connection timeout for incoming email, modifying `/etc/postfix/master.cf` as follows:

```
relay      unix      -      -      n      -      -      smtp
-o smtp_connect_timeout=2 -o smtp_helo_timeout=2
```

If you see that Postfix pegs disk I/O when processing incoming mail, the *real* solution is to get faster disks or to allocate one disk for logging, one disk for the mail queue, and a third disk for user mailboxes. Postfix-caused disk saturation is especially a problem if you are serving multiple virtual hosts on a single system. One workaround is to configure multiple IP addresses for the machine and to run a Postfix instance for each IP address, where each Postfix instance writes to a different disk. It is easier to configure than it might seem at first glance. The key is starting each Postfix instance with a different configuration directory:

```
# postfix -c conf_dir start
```

Replace `conf_dir` with the name of the configuration directory assigned to each IP address. Within each `conf_dir`, `main.cf` has a different `$myhostname`, `$queue_directory`, and `$inet_interfaces` setting, depending on the interface or IP it handles. For example, if you have two virtual hosts, `first.vhost.com` and `second.vhost.com`, you might arrange it like this: For `first.vhost.com`, suppose that the Postfix's configuration directory is `/first/postfix`. Thus, the configuration file `/first/postfix/main.cf` might have the following entries:

```
queue_directory = /first/queue/dir
myhostname = first.vhost.com
inet_interfaces = $myhostname
```

Accordingly, start the Postfix instance for `first.vhost.com` this way:

```
postfix -c /first/postfix start
```

Make a similar arrangement for `second.vhost.com`. If the configuration directory is `/second/postfix`, the configuration file is `/second/postfix/main.cf`, which has the following entries for `second.vhost.com`:

```
queue_directory = /second/queue/dir
myhostname = second.vhost.com
inet_interfaces = $myhostname
```

The proper Postfix invocation is:

```
postfix -c /second/postfix start
```

If Postfix responds too slowly to incoming SMTP connections but POP or IMAP connections are acceptably fast, you need to run more SMTP server processes. Edit the `smtpd` entry in the `master.cf` file and increase the process limit. Alternatively, increase the `default_process_limit` setting in the `main.cf` file.

TIP Anytime you edit one of Postfix's configuration files, be sure to use the `postfix reload` command to activate the changes.

Optimizing FTP Services

Out of the box, `vsftpd` is pretty darn fast and makes lightweight demands on a system's memory and CPU resources. If its speed fails to suit you, the following tips, adapted from the `vsftpd` documentation, might help:

- If possible, disable the NIS and NIS+ (`nis` and `nisplus`) for `passwd`, `shadow`, and `group` lookups in `/etc/nsswitch.conf`. The idea with this tip is to avoid loading unnecessary runtime libraries into the `vsftpd`'s memory space and to avoid using NIS for lookups that can be resolved more quickly by resorting to file-based lookups.
- Break directories with more than a few hundred entries into smaller directories. Many file systems, such as `ext2` and `ext3`, do not handle such cases efficiently at all, and the process of creating listings of large directories (with, for example, the `ls` or `dir` commands) causes `vsftpd` to use moderate amounts of memory and CPU. If you are stuck with large directories, use a file system, such as `XFS`, `JFS`, or `ReiserFS`, designed to work with large directory structures.
- Limit the number of simultaneous connections to the FTP server.
- More drastically, if the load on your FTP server is bogging down the system, you could disable anonymous FTP altogether or dedicate a machine to providing FTP services.
- Take advantage of `vsftpd`'s bandwidth throttling features to limit the network bandwidth consumed by any one connection or connection classes.

Optimizing Web Services

Chapter 23 briefly touched on Apache configuration settings you can modify that affect Apache's performance. The settings mentioned in that section are

good starting points for fine-tuning Apache, but they do not exhaust the possibilities. To recap that discussion:

- Increasing the `MaxClients` setting (to a maximum of 256) increases the maximum number of simultaneous client connections to the `httpd` server before the server starts refusing additional connections. The default value is 150 (clients). One generally-accepted rule-of-thumb formula is:

$$\text{MaxClients} = \frac{\text{Physical RAM} - 128 \text{ MB} + \text{Size of Active Pages}}{\text{Nonshared Memory per httpd Process}}$$

The theory is that you should use physical RAM for system resources and caching active pages. Leftover RAM should be used by `httpd` processes serving up active pages. If you have more clients, you will end up swapping, which degrades performance. If you have fewer clients, you will not be maximizing the available system resources. In practice, you will have to decide what constitutes an active page. One way to go about this is to use the server logs to evaluate which pages are served more than once every `Timeout` period, which defaults to 300 seconds (5 minutes).

- The `Timeout` directive controls how long the server waits between protocol messages before it closes a connection. The longer the `Timeout` directive, the longer a client connection will be tied up and, thus, unavailable to another client. The default value is 300 (seconds).
- The `MaxRequestsPerChild` setting controls how many HTTP requests an `httpd` child process will service before a new child process starts. The default value is 100, but setting it to 0, for unlimited requests, will work just fine on a Red Hat system.
- `MaxKeepAliveRequests`, 100 by default, sets the upper limit on the total number of requests from the same client on the same connection.

The following tips and suggestions appear in no particular order. Your mileage may vary, and if breaks, you get to keep both pieces. Some of the following might work better than others; others ideas might fail miserably. If your server is running a lot of CGI scripts or using PHP markup, you should look into resources that discuss Apache tuning in depth. The overhead requirements of PHP and CGI scripts involve creating new processes rather than merely additional RAM, network, or disk I/O.

- Set `HostnameLookups` to `Off`. Each resolver call impairs performance. If you need to resolve IP addresses to hostnames, you can use Apache's `logresolve` program or one of the resolver programs available in the log reporting and analysis packages.

- Similarly, use IP addresses instead of host names in `Allow from domain` and `Deny from domain` directives. Each such query, when *domain* is a name, performs a reverse DNS query followed by a forward query to make sure that the reverse query is not being spoofed. Using IP addresses avoids having to resolve names to IP numbers before performing the reverse and forward queries.
- If you do not use `Options FollowSymLinks`, or if you *do* use `Options SymLinksIfOwnerMatch`, Apache performs extra system calls to check symbolic links. For example, suppose you have the following configuration:

```
DocumentRoot /var/www/htdocs
<Directory />
    Options SymLinksIfOwnerMatch
</Directory>
```

If a client then requests `/index.html`, Apache performs an `lstat()` system call on `/var`, `/var/www`, `/var/www/htdocs`, and `/var/www/htdocs/index.html` to check the owner matching of the symbolic link. The overhead of these `lstat()` system calls occurs for *each* request, and Apache does not cache the results of the system calls. For the best performance (and, unfortunately, the least security against rogue symlinks), set `Options FollowSymLinks` for all directories and never set `Options SymLinksIfOwnerMatch`.

- A similar performance problem occurs when you use `.htaccess` files to override directory settings. In this case, Apache attempts to open `.htaccess` for each component of a requested filename. For the best performance use `AllowOverride None` everywhere in the Web space Apache is serving.
- Unless you rely on the `MultiView` option, turn it off. It is perhaps the single biggest performance hit you can throw at an Apache server.
- Do not use NFS mounted file systems to store files that Apache serves unless absolutely necessary. Not only is the read performance of NFS slower than the read performance of a local file but also the file being served via NFS might disappear or change, causing NFS cache consistency problems. Moreover, if the Apache server is somehow compromised, the NFS mount will be vulnerable.
- If you must use NFS-mounted file systems, mount them as read-only. Read-only NFS mounts are significantly faster than read/write mounts. Not only will this improve performance, disabling write access adds another barrier to bad guys who might compromise the system.

- The single most important system resource that Apache uses is RAM. As far as Apache is concerned, more RAM is better because it improves Apache's ability to store frequently requested pages in its cache. You can also help by limiting the non-Apache processes to the absolute minimum required to boot the system and enable Apache to run — that is, run a dedicated Web server that doesn't need to share the CPU or memory with other processes. Naturally, a faster CPU, a high-speed Ethernet connection, and SCSI disks are preferable.

Summary

Fedora Core and RHEL systems are commonly deployed to provide Internet services, so this chapter mentioned some methods you can use to improve the performance of several key Internet services: LDAP, DNS, email, and Web services. We could list only some of the areas to consider when tuning LDAP, because LDAP performance tuning is a complex subject best addressed by the LDAP authorities. DNS is more easily tuned. A DNS client's performance can often be improved simply by running a caching nameserver, while there are several methods available for getting better query performance from a server. Mail servers are high-volume, heavy throughput systems requiring careful tuning, but sometimes, simply replacing Sendmail with Postfix can fix slow mail-processing times. We also mentioned a number of methods you can use to get faster page-serving behavior from Apache.

PART

Four

System Administration

- Chapter 26:** Keeping Your System Updated with up2date and the Red Hat Network
- Chapter 27:** Upgrading and Customizing the Kernel
- Chapter 28:** Configuring the System at the Command Line
- Chapter 29:** Administering Users and Groups
- Chapter 30:** Installing and Upgrading Software Packages
- Chapter 31:** Backing Up and Restoring the File System
- Chapter 32:** Performance Monitoring

Keeping Your System Updated with up2date and the Red Hat Network

IN THIS CHAPTER

- Using the RedHat up2date Agent
- Registering Your System with the Red Hat Network
- Accessing the Red Hat Network with a Web browser

The Red Hat Network up2date agent is a program that is installed by default when you install Fedora Core or Red Hat Enterprise Linux. The Red Hat Network up2date software will give you visual notification of the update right on your desktop. If you purchased Enterprise Linux, you are entitled to subscribe to the Red Hat Network, which gives you access to many additional services, such as registering your system, receiving email notifications and scheduling automatic system updates. As a registered Red Hat Network user, you can use a Web browser to access your account.

This might not sound like much at first, but think about the many steps involved in keeping your system up to date with the latest versions of the hundreds of packages that are installed on your system. The Red Hat Network practically eliminates the need for you to search for these packages because you can receive this information by email. As a registered Red Hat Network user, or a Fedora Core user you can also search for updates by using the up2date agent. With the Red Hat Network, you can now easily keep your system running reliably and securely. A few steps are involved in setting up the Red Hat Network, but they are well worth the effort. In this chapter, you can read how to configure the up2date agent, and then connect to look for updated files. If you are running Enterprise Linux you learn how to register your system with Red Hat.

Using the Red Hat up2date Agent

The up2date agent is a valuable tool for you because it helps you keep your system running the most current versions of the packages installed on your system. During the system installation, an Alert icon is placed on the right side of the top desktop panel that provides visual notification when your system needs to be updated. Figure 26-1 shows the location of the Alert icon on the panel.

The Alert icon is colored-coded, representing a different state of the update. The icons and their meaning are discussed in the following list:

- **Blue check mark** — The system is up to date.
- **Green double arrows** — The system is checking for updates.
- **Red exclamation point** — The system needs to be updated.
- **Gray question mark** — An error has occurred.

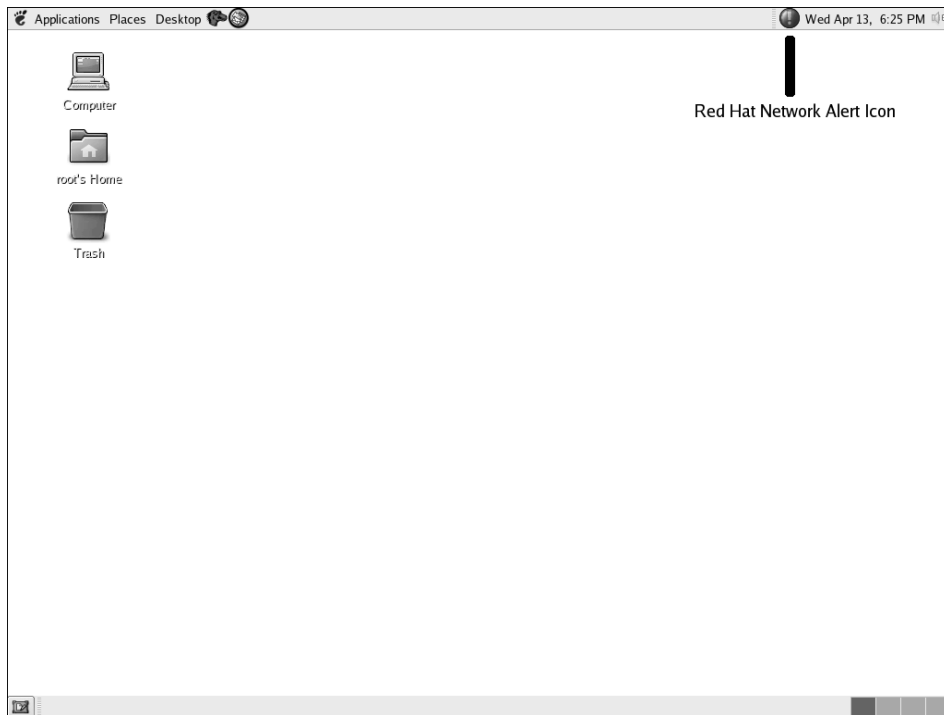


Figure 26-1 The Red Hat Network Alert icon.

Configuring the up2date Agent

Before you can use the Red Hat up2date agent, you need to configure it. The first time you start the up2date agent you will be prompted to install the GPG public key from Red Hat, which is used to verify the packages are signed and are from Red Hat. Be sure to install the key.

You can start the up2date Agent Configuration tool by doing any of the following:

- Open a terminal window, and from the command line, type **up2date-config**.
- Right-click the alert icon, and choose Configuration from the pop-up menu.

Choose Application ⇨ System Settings ⇨ Red Hat Network Configuration. The Red Hat Network Configuration dialog box, shown in Figure 26-2, opens.

This dialog box has three tabs — General, Retrieval/Installation, and Package Exceptions:

- **General** — The General tab is the tab shown by default when the dialog box opens. A server is already selected for you to use, and you should not need to change it. If you need to use a proxy server to connect to the Web, you can enter this information here by first selecting the Enable HTTP Proxy check box and then entering the name of your server in the field next to the check box. If you need to use authentication, you can enable it by selecting the Use Authentication check box and filling in the Username and the Password fields.

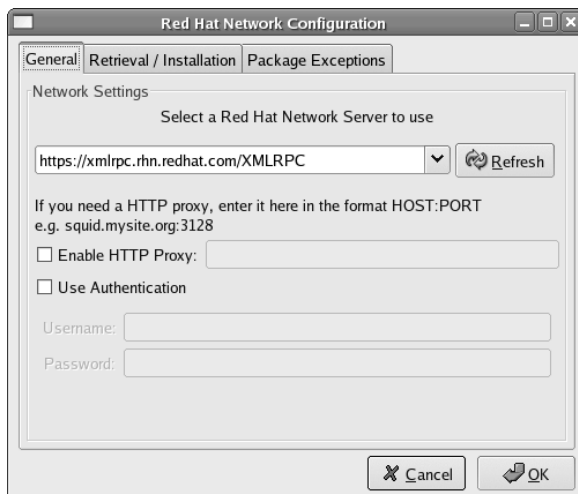


Figure 26-2 The Red Hat Network Configuration dialog box.

- **Retrieval/Installation** — From this tab (see Figure 26-3), you can choose options that affect how the packages are retrieved and subsequently installed.

The Package Retrieval Options are:

- **Do Not Install Packages After Retrieval** — By default, packages are automatically installed after they are retrieved. If you enable this option, packages are retrieved to the specified directory but not installed.
- **Do Not Upgrade Packages When Local Configuration File Has Been Modified** — If you have manually modified configuration files for packages on your system, these packages are not displayed by default. If you disable this option, the packages are displayed.
- **Retrieve Source RPM Along with Binary Package** — By default, the source Red Hat Package Manager (RPM) is not downloaded with the binary version of the package. By enabling this option, you also retrieve the source of the RPM.

The Package Verification Options has but one choice:

- **Use GPG to Verify Package Integrity** — By default, for security purposes, the packages are verified to ensure they contain the Red Hat GPG signature. If you disable this option, the security check is not performed.

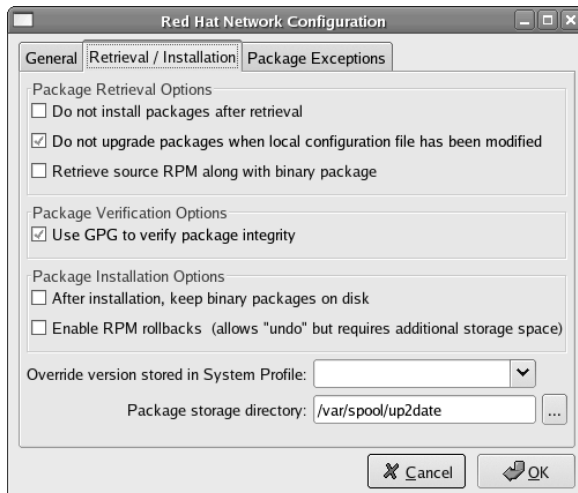


Figure 26-3 The Retrieval/Installation tab.

NOTE GPG stands for GNU Privacy Guard, which is the open-source replacement for PGP. PGP (Pretty Good Privacy) was developed in the mid-1990s by Phil Zimmerman to provide data encryption. GPG was developed to replace PGP because PGP contained a patented algorithm and its use was restricted. GPG can be freely used without concern for patented information.

The Package Installation Options are:

- *After Installation, Keep Binary Packages on Disk* — By default, the packages are removed from the local disk after they are installed. Enabling this option leaves a copy of the package in the specified directory.
- *Enable RPM rollbacks* — Choosing this option lets you restore your system to its condition before the RPM was installed.

The last two items on this tab are:

- *Override Version Stored in System Profile* — By filling in this field, you can override the version stored in your System Profile with the version in the field.
- *Package Storage Directory* — Here you can specify the storage location of the packages on your system.
- **Package Exceptions** — From this tab (see Figure 26-4), you can choose to exclude packages by the name of the package or by the name of the file.
 - *To exclude a package by package name* — Type the name of the package in the Add New field in the Package Names to Skip section, and then click the top Add button.
 - *To exclude a package by filename* — Type the name of the file in the Add New field in the File Names to Skip section, and then click the bottom Add button.

After you make any changes to the three tabs, click OK. Your configuration changes are saved, and you can now use the up2date agent.

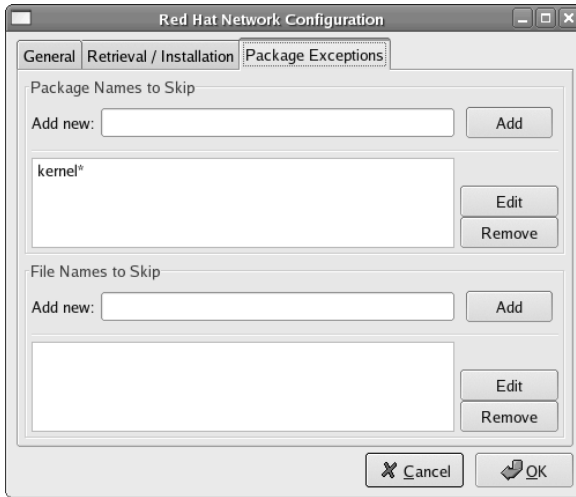


Figure 26-4 The Package Exceptions tab.

Updating Your System

Whenever your system needs to be updated, the Alert icon will appear as a red circle containing an exclamation point. You can roll the mouse over the Alert icon to view a small pop-up window that gives additional information. If your system needs to be updated the pop-up window will show the number of updates available.

There are multiple ways to start the up2date agent, here is one way. To start the up2date agent to update your system, do the following:

1. Right-click the Alert icon and select Launch Up2date from the contextual menu. You see the Red Hat Update Agent Welcome screen.

NOTE If you are not logged in as root, you will be prompted for the root password.

2. Click Forward to continue to the Channels dialog box, as shown in Figure 26-5.
3. The channels dialog box lists the channels that will be searched from which the updated packages will be obtained. You can think of the channels as file repositories on various servers in many locations. Click Forward to continue.

The program connects to the selected channel to search for package updates. Next you see the Skipped Packages dialog box, as shown in Figure 26-6. By default the kernel packages are not automatically updated and will be listed as packages to be skipped.

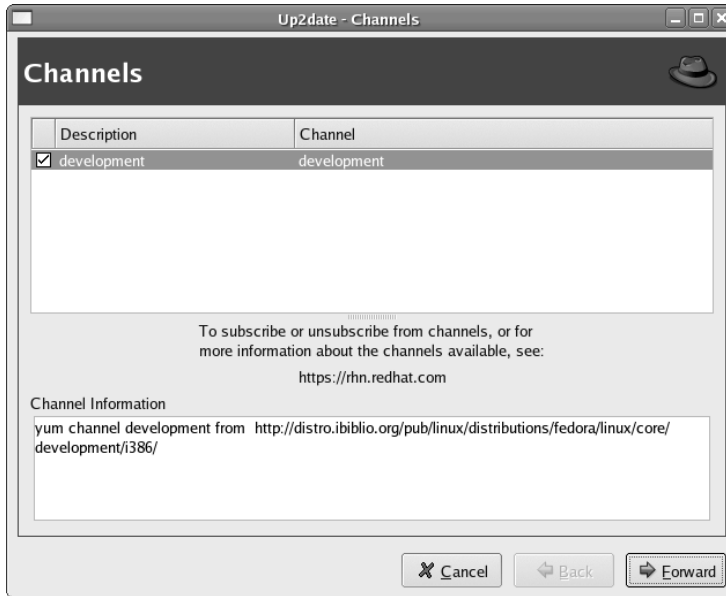


Figure 26-5 The Channels dialog box.

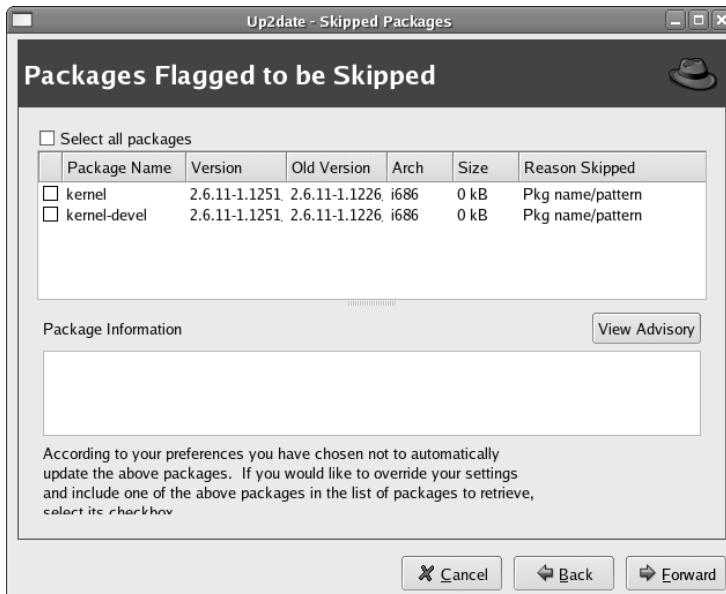


Figure 26-6 The Skipped Packages dialog box.

4. If you want to update the kernel packages check the appropriate box and then Click Forward to continue. You see the Package List dialog box with the available packages, as shown in Figure 26-7. If your system is updated, you won't see any packages listed.
5. You can select packages individually by selecting the check box in front of the package name, or you can mark the Select All Packages check box to select all packages. After you finish selecting packages, click Forward to begin package retrieval. You see the Package Retrieval dialog as shown in Figure 26-8.
6. The up2date program gets the packages and prompts you to continue after the packages have been retrieved. Click Forward to install the packages.
7. You see a progress dialog box during the package installation. After all the packages that you selected for installation are installed, you see a dialog box indicating the package installation has finished. Click Finish to complete the update process.

NOTE You can click the Alert icon to open a dialog box listing all available packages. You can also launch up2date from here.

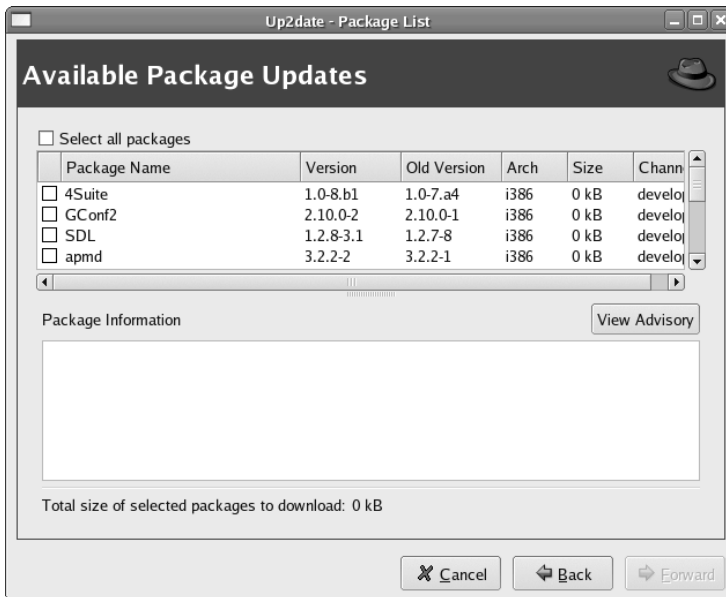


Figure 26-7 The Package List dialog box.



Figure 26-8 The Package Retrieval dialog box.

Registering Your System

NOTE If you installed Fedora Core on your system, you can skip this section because you do not need to register this version. If you installed any version of Enterprise Linux, you should register your system.

Before you can begin using the Red Hat Network, you must first register your system with Red Hat by using the Red Hat Network registration client.

NOTE You can also register for the Red Hat Network at the Red Hat Web site by using your Web browser to go to <http://rhn.redhat.com/network>, creating a login, and filling in the online registration form.

You must be logged in as root to perform the registration, and you must also have a connection to the Internet to be able to log on to the Red Hat Web site. If you aren't connected to the Internet, you receive an error message, and the program closes when you click OK.

You start and use the registration client by following these steps:

1. Open a terminal and enter the command **up2date -register** at the command line.

You see the Red Hat Network Configuration screen where you can set a proxy server and authentication information if you need to. Click OK to go on and click Yes to install the GPG key. The Red Hat Update Agent screen appears that provides a description and lists the benefits of the Red Hat Network. If you choose not to use the Red Hat Network, you can click Cancel to end the process.

2. Click Forward to continue to the Up2date Login Page dialog box, as shown in Figure 26-9.

You use this dialog box to create a new account or use an existing one if you've already created an account.

3. If you already have an account, check I have an existing Red Hat login radio button and enter the username and password in the appropriate text boxes. Go to Step 4.

If you don't have a Red Hat Network login, you need to create one. Check the I don't have radio button and click Forward. A new page opens where you can enter your information. (See Figure 26-10.) All fields with an asterisk are required fields.



Figure 26-9 The Up2date Login dialog box.

Figure 26-10 The Create Login dialog box.

The username must be at least five characters and cannot contain any spaces, tabs, line feeds, or reserved characters such as ' , & , + , or % . You also enter your email address here.

4. After you enter the appropriate information, click Forward to continue to the Up2date Activate dialog box, as shown in Figure 26-11.

On this page, you can enter your subscription number, if you have one, and set your system name. By default, your system hardware information and installed packages are sent to the Red Hat Network, but you can choose not to send them by deselecting the appropriate check box.

5. Click Forward to continue. Your system profile is sent to the Red Hat network.
6. Next you see the Channels dialog box. You don't need to do anything on this dialog box so click Forward. After the registration process is finished, you receive a confirmation message stating that you have successfully registered your system profile on the Red Hat Network. Now you are ready to use the up2date agent.

NOTE If you installed Fedora Core on your system, you can skip the next section. If you installed any version of Enterprise Linux, you can access the Red Hat Network using a Web browser.



Figure 26-11 The Activate dialog box.

Accessing the Red Hat Network with a Web Browser

NOTE If you installed Fedora Core on your system, you can skip this section. If you installed any version of Enterprise Linux, you can access the Red Hat Network using a Web browser.

You can access the Red Hat Network by using any Web browser. In fact, if you have multiple machines, you can manage them at the same time through the browser interface. All you need to do is go to the Red Hat Web site at <http://rhn.redhat.com> and log in with the username and password that you assigned to the account that you created earlier in this chapter.

After logging in, you see the Your RHN tabbed page for your system, as shown in Figure 26-12.

The Your RHN page shown in Figure 26-12 might not look exactly like your page. The information shown here is dependent on the preferences that you might have chosen if you already have an active account. Take a look at the significant areas of this page.

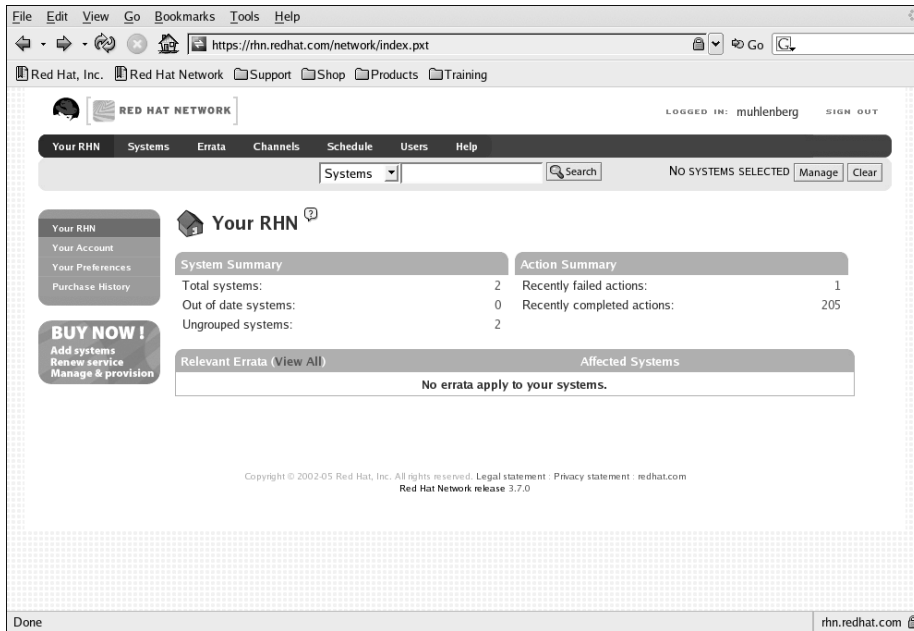


Figure 26-12 The Red Hat Network main page.

At the top of the page are seven links that can be used to take you to other pages where you can get more information about you systems and make configuration changes. The seven links are:

- **Your RHN** — This page opens by default after you log in to the Red Hat network. In the center of the page is a summary of your system (or systems, depending on how many you have configured). Along the left side of the page are links to information about your account and settings that you can change.

- **Systems** — The Systems link opens a page that provides a general overview of your systems, as shown in Figure 26-13.

You can click the link for your system that is shown in the body of the page to get information specific to that system. Along the left side of the page are links to pages that provide additional information about your systems.

- **Errata** — Clicking the Errata link opens the page shown in Figure 26-14, listing any errata that applies to your system. As with the other pages, there are more links along the left side of the page that give additional information related to the Errata.

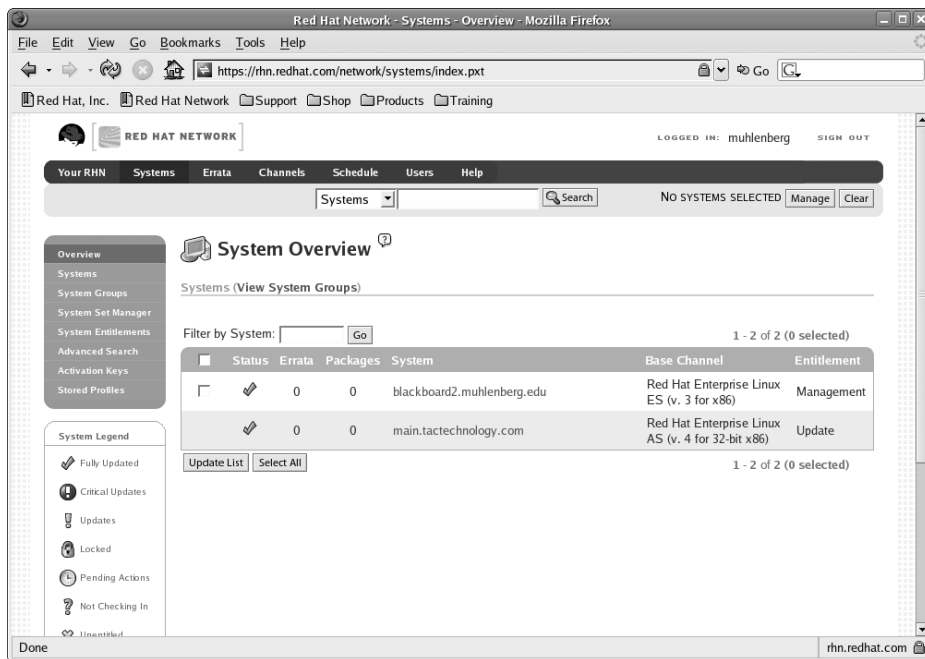


Figure 26-13 The Systems page shows general system information.

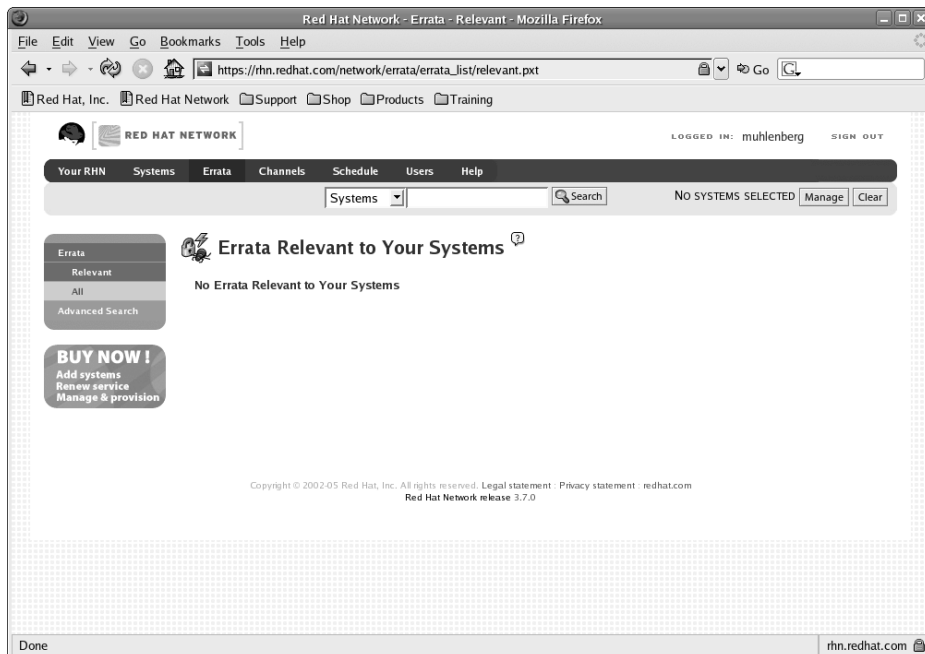


Figure 26-14 The Errata page shows system errata information.

- **Channels** — The Channels page, shown in Figure 26-15, shows the software channels to which you have subscribed. You can download software for your systems from channels to which you have subscribed. Like the other linked pages, there are additional links on the left side of the page for obtaining more information or services related to the Channels.

In the list on this page, you will see the first link for Red Hat Enterprise Linux AS shows a 1 in the system column. This means that there is one system subscribed to this channel. You can click this link, or any link that is shown underneath it, to go to the Channels detail page, as shown in Figure 26-16.

From the Channels details page you can click the Download link to download the files relevant to the link you chose to bring you to the Channels detail page.

Red Hat Network - Channels - Software Channels - Relevant - Mozilla Firefox

File Edit View Go Bookmarks Tools Help

https://rhn.redhat.com/network/software/index.pxt

Red Hat, Inc. Red Hat Network Support Shop Products Training

LOGGED IN: muhlenberg SIGN OUT

Your RHN Systems Errata Channels Schedule Users Help

Systems Search NO SYSTEMS SELECTED Manage Clear

Software Channels Overview

The software channels listed below are most relevant to your organization. You may also [view a list of all available supported software channels](#), as well as [retired channels](#).

Channel Name	Packages	Systems
Red Hat Enterprise Linux AS (v. 4 for 32-bit x86)	1464	1
↳ RHEL AS (v. 4 for x86) Beta	207	0
↳ RHEL AS (v. 4 for x86) Extras	20	0
↳ RHEL AS (v. 4 for x86) Extras Beta	20	0
↳ RHEL AS (v. 4 for x86) Hardware Certification	2	0
↳ RHEL AS (v. 4 for x86) SDK Beta	1	0
Red Hat Enterprise Linux ES (v. 4 for 32-bit x86)	1464	0
↳ RHEL ES (v. 4 for x86) Beta	207	0
↳ RHEL ES (v. 4 for x86) Extras	20	0
↳ RHEL ES (v. 4 for x86) Extras Beta	20	0
↳ RHEL ES (v. 4 for x86) SDK Beta	1	0

Done

rhns.redhat.com

Figure 26-15 The Channels page shows your subscription entitlements.

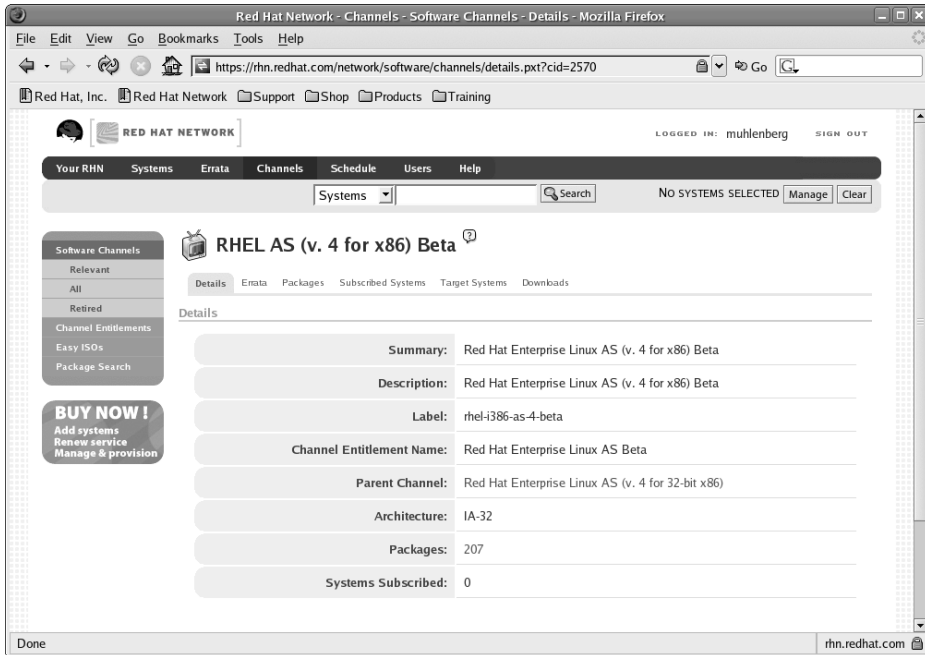


Figure 26-16 The Channels detail page gives you access to downloads for the channel.

- **Schedule** — The Schedule tab, shown in Figure 26-17, gives you information about pending actions for the selected system. The links on the left side of the page give additional information about completed, failed, and archived actions.
- **Users** — The Users tab, shown in Figure 26-18, shows the users who can access the Red Hat Network and their functions.
- **Help** — You already have a good idea what you'll find when you click this link. On this page, shown in Figure 26-19, you'll find links to many good sources of information about Enterprise Linux.

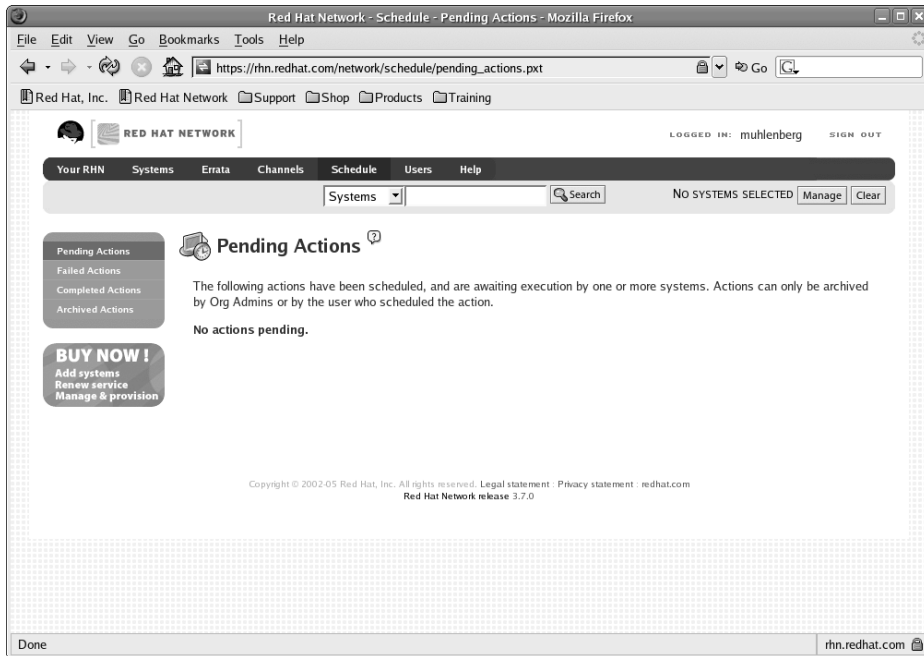


Figure 26-17 The Schedule page shows pending, failed, completed, and archived actions.

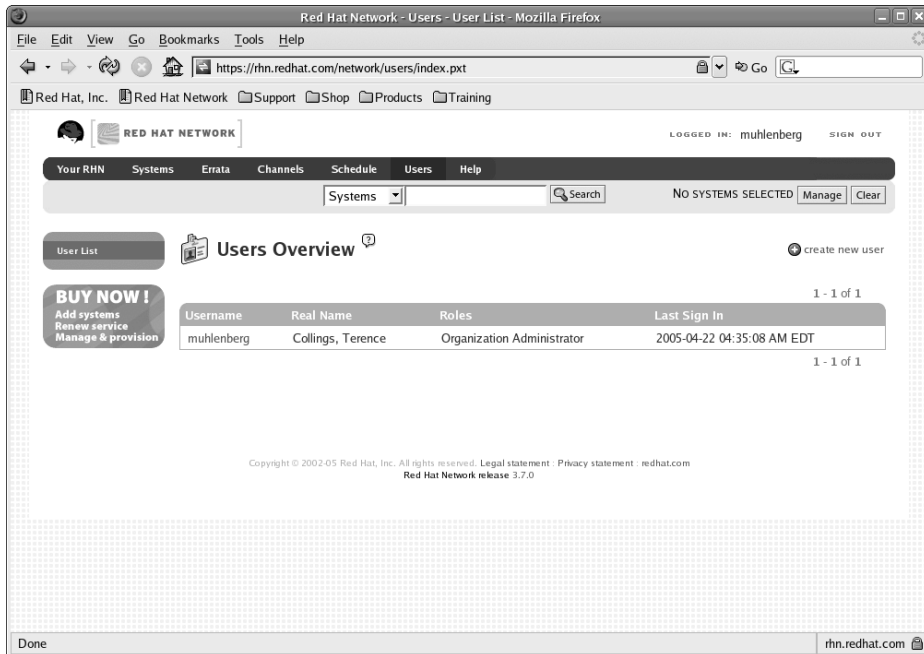


Figure 26-18 The Users page shows the administrators allowed to access the Red Hat Network account.

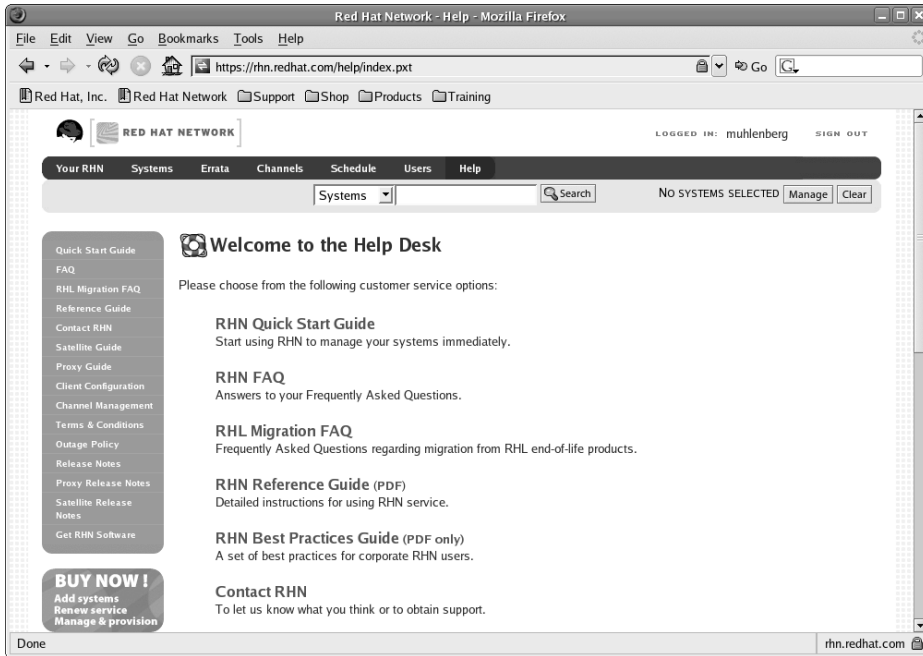


Figure 26-19 The Help page takes you to links to helpful information.

Summary

This chapter showed you how to keep your Fedora Core or Enterprise Linux system updated. You learned how to use the `up2date` program to find out about new updates for your system and to download and install them. You learned how to register your Enterprise Linux system with the Red Hat Network. Finally, you learned that you can access the Red Hat Network using a Web browser, and you looked at the Web interface pages to learn what you can do with them.

Upgrading and Customizing the Kernel

IN THIS CHAPTER

- Determining Whether to Upgrade to a New Kernel
- Upgrading versus Customizing
- Preparing to Upgrade
- Installing a Kernel RPM
- Getting the Kernel Source
- Configuring the Kernel
- Reviewing the Configuration Options
- Compiling the Kernel
- Installing the Kernel
- Updating GRUB

The Linux kernel has changed in a number of significant ways since the last edition of this book. The kernel itself is faster, more capable, and more flexible, and the build process has changed dramatically and for the better. The kernel is the core of the operating system and runs the CPU, manages system memory, controls access to disk drives, and contains device drivers that enable you to use the hardware and peripherals attached to the computer. The ability to update and customize the Linux kernel is one of the things that many people like best about Linux. Naturally, this feature appeals most to incorrigible tweekers and tinkerers, but it also appeals to system administrators who are responsible for wringing the most performance and benefit out of their existing hardware and software. In some cases, rebuilding the kernel is required to support new hardware that is not supported, or that is poorly supported, by your system's existing kernel.

Determining Whether to Upgrade to a New Kernel

Should you upgrade to a new kernel? Strictly speaking, no, it is rarely *necessary* to do so. The kernel provided with Fedora Core and RHEL is deliberately configured to support the widest possible array of existing PC hardware. Moreover, practically all of the functionality that most users need (both in home and hobbyist environments and in business and professional settings) is already available in the current kernel (version 2.6.10 at the time this paragraph was written). Many users, especially those coming from a predominantly Windows environment, are conditioned to downloading and installing the latest and greatest version of application X, regardless of whether or not they need the features and bug fixes it provides. The fact is that most users do not need to do this because they use perhaps 20 percent of the feature sets of existing software. Adding still more unused features contributes to software bloat and, potentially, to system instability.

When is it necessary to rebuild the kernel? Often as not, you rebuild the kernel usually to provide better support for the odd hardware device, to add support for new hardware you have added to an existing system, to add a driver for a device not supported by the existing kernel, to fix the occasional bug, or to close a security hole. The Fedora Project (for Fedora Core) and Red Hat (for RHEL) release a steady stream of patches to address kernel security issues, and these errata should be applied as quickly as possible, especially on systems exposed to the Internet. We personally consider a kernel customized for your specific hardware, usage profile, and personal preferences superior to the stock Red Hat kernel, although you will not go wrong using the standard Red Hat Linux kernel.

NOTE If you use Up2date (with Fedora Core) or the Red Hat Network (with RHEL), you system's kernel will be upgraded automatically. If you are using any third-party device drivers provided as loadable kernel modules (LKMs), such as VMWare, or binary-only drivers, such as the NVIDIA and ATI framebuffer drivers, it might be necessary to rebuild these modules after a kernel update. In this case, the kernel does not need to be rebuilt, but you *will* need a configured kernel source tree matching the newly installed kernel so that the LKMs can be rebuilt against that kernel.

Okay, that takes care of the practical reasons for and against upgrading to a new kernel. A significant portion of the Linux community, particularly home users, hobbyists, and developers, constantly upgrade and rebuild their kernels simply because they can or for the sheer satisfaction of the undertaking. For

many, including the authors of this book, Linux is a hobby, just as is stamp collecting or flying model airplanes — few people question the value of these more traditional hobbies or ask why people pursue them, so this new hobby should not be considered any different. “Because I can,” “because it’s there,” or “because I want to” are perfectly acceptable reasons. Don’t forget the geek factor: knowing how to roll your own kernel racks up serious geek points.

Chances are, however, that if you are reading this book, the previous paragraph is preaching to the choir. If not, the following list summarizes the most common reasons you might want or need to upgrade or customize the kernel on your Fedora Core or RHEL system:

- You can recompile the kernel to support your specific CPU, especially features that improve performance. The default Red Hat Linux installation installs a kernel configured to run on the widest possible variety of Intel CPUs. As a result, it does not take advantage of all of the features and improvements available in the newest CPUs or motherboard chipsets.
- Similarly, the default kernel often includes system features that you do not need or does not include features that you do need or want. Customizing and recompiling the kernel enables you to remove unnecessary or unwanted features and to add needed and desired features.
- The default kernel supports an enormous variety of the most common hardware, but no single system needs all of that support. You might want to create a new kernel that includes support for only the hardware actually installed on your system.
- If you have a system with hardware not supported when you installed Red Hat Linux or for which only experimental support was available, you can rebuild the kernel to include that support once it becomes available or to improve existing support.
- You’re dying to use the latest and greatest bleeding-edge kernel version.

Why should you not build a custom kernel? The compelling reason for RHEL users is that when you diverge from the Red Hat–blessed kernel or kernel sources, you lose official Red Hat support; if Red Hat support is important to you, stick with official Red Hat kernels. For Fedora Core users, the major (potential) problem with custom kernels is that a newly installed piece of hardware will not work because you failed to compile a driver. For users of either Fedora Core or RHEL, finally, one of the virtues of “official” Red Hat Linux kernels is that they reduce the likelihood of subtle compatibility problems with the rest of your installed software that can develop when you roll your own.

Upgrading versus Customizing

As used in this chapter, customizing the kernel and upgrading the kernel refer to two different procedures. *Customizing* the kernel refers to reconfiguring an existing kernel source code tree, recompiling it, installing the new kernel, and booting it. You can download either a complete source tree (now over 30 MB even when compressed) or one or more patches (described later in the section titled “Patching the Kernel”). Of the two options, downloading a series of patch files is faster than downloading an entire kernel source tree.

Upgrading the kernel means a new kernel image, probably from kernel RPMs provided by the Fedora Project or Red Hat Software (RHEL users). Whether you customize or upgrade, though, the end result is the same: a new kernel configured to your liking.

Preparing to Upgrade

Upgrading your kernel is a major change, so prudence and experience dictate taking some preventive measures first. Make sure that you have a working boot disk for your system in case a problem occurs. You will not be able to boot your system unless you have a boot disk that you know works. So, in this section, you’ll create a GRUB boot floppy.

1. Insert a floppy disk into the disk drive, and type the following command to perform a low-level format:

```
# fdformat /dev/fd0
Double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB.
Formatting ... done
Verifying ... done
```

2. Create an ext2 file system on the floppy disk:

```
# mke2fs /dev/fd0
mke2fs 1.36 (05-Feb-2005)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
184 inodes, 1440 blocks
72 blocks (5.00%) reserved for the super user
First data block=1
Maximum filesystem blocks=1572864
1 block group
1812 blocks per group, 8192 fragments per group
184 inodes per group
```

```
Writing inode tables: done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 39 mounts or
180 days, whichever comes first. Use tune2fs -c or -i to override.
```

3. If the mount point `/mnt/floppy` doesn't exist, create it:

```
# mkdir -p /mnt/floppy
```

4. Mount the floppy so the GRUB installer can access it:

```
# mount -t ext2 /dev/fd0 /mnt/floppy
```

5. Install the GRUB boot loader on the floppy's MBR:

```
# grub-install --root-directory=/mnt/floppy '(fd0)'
```

Probing devices to guess BIOS drives. This may take a long time.

Installation finished. No error reported.

This is the contents of the device map `/mnt/boot/grub/device.map`.

Check if this is correct or not. If any of the lines is incorrect, fix it and re-run the script ``grub-install``

```
(fd0)    /dev/fd0
(hd0)    /dev/hda
(hd1)    /dev/hdb
```

6. Copy the GRUB configuration file onto the mounted floppy disk:

```
# cp /boot/grub/grub.conf /mnt/floppy/boot/grub/grub.conf
```

7. Sync the disks and then unmount the floppy:

```
# sync; sync; sync
# umount /mnt/floppy
```

8. Reboot your system and boot from the GRUB boot floppy you just created to make sure that it works. If it does, you're ready to proceed.

TIP You can use the script `mkgrubdisk` to create a GRUB boot disk.

`mkgrubdisk` is available on the download site for this book.

Installing a Kernel RPM

In most cases, updated kernel RPMs will be installed automatically if you subscribe to the Red Hat Network or use `up2date` (or one of its analogues, such as `yum`). If the kernel is upgraded automatically by the Red Hat Network, the only task left for you is to reboot and select the new kernel (if it isn't the default) at the GRUB boot prompt. As remarked earlier, you might need to rebuild any LKMs provided by third parties against the new kernel.

Of course, you don't have to rely on the Red Hat Update agent. To see if an updated kernel RPM is available for your version of RHEL, go to <https://www.redhat.com/security/updates>, select the version of RHEL you are using (such as Red Hat Enterprise Linux AS, version 4), and see if the errata include any kernel updates. If there is an update, which is usually due to a security advisory, the errata will indicate the kernel RPM you need to download *and* any additional RPM packages that are required.

For example, Red Hat issued a security advisory against the kernel in RHEL AS 4 on February 18, 2005. (You can review the advisory at <https://rhn.redhat.com/errata/RHSA-2005-092.html>.) A list of RPMs that needed to be upgraded to address the issue appeared in the advisory's text. For i386 systems (referred to as IA-32 systems in the advisory), the list of RPMs included:

- `kernel-2.6.9-5.0.3.EL.i686.rpm`
- `kernel-devel-2.6.9-5.0.3.EL.i686.rpm`
- `kernel-doc-2.6.9-5.0.3.EL.noarch.rpm`
- `kernel-hugemem-2.6.9-5.0.3.EL.i686.rpm`
- `kernel-hugemem-devel-2.6.9-5.0.3.EL.i686.rpm`
- `kernel-smp-2.6.9-5.0.3.EL.i686.rpm`
- `kernel-smp-devel-2.6.9-5.0.3.EL.i686.rpm`

Using the `up2date` program (the Red Hat Update agent), you could download and install those RPMs, reboot, and you're done. Alternatively, if you're not a Red Hat Network subscriber or haven't purchased RHEL, you could download the source RPM from Red Hat's FTP server or one of their mirror sites and build it yourself. In this particular case, you would need the file `kernel-2.6.9-5.EL.src.rpm`, which is stored on <ftp.redhat.com> in `/pub/redhat/linux/enterprise/4/en/os/i386/SRPMS`.

The Red Hat Update agent handles all of this for you automatically if you subscribe to the Red Hat Network or have purchased one of the RHEL products, so there's not much more to say about it in this chapter. For more information about using the Red Hat Network and the Red Hat Update Agent, see Chapter 26, which discusses these topics in detail. If you want to use the kernel source RPM, read the section in this chapter titled "Using the Kernel Source RPM."

Getting the Kernel Source

To obtain the kernel source, you have two options, downloading the kernel source in RPM format from Red Hat or the Fedora Project and downloading a kernel source code archive from one of the many Linux kernel mirror sites.

Using the source RPM (SRPM) is simpler because the build and installation process involves only a few steps. Another benefit is that the SRPM contains a number of modifications made by Red Hat engineers and contributors to the Fedora project. On the downside, however, the SRPM makes it difficult to modify the kernel configuration to your liking without doing some additional work that involves modifying the SRPM.

Downloading the source code from one of the kernel mirror sites is more involved but ultimately gives you more control over the kernel configuration and build process. The additional work involves finding a kernel mirror to use, downloading a 35-MB archive file (of course, the SRPM is just as large), and of course, the actual configuration process. The payoff, however, is a much more customized kernel. Which method you use is up to you; the text describes both.

Using the Kernel Source RPM

Installing the kernel SRPM is simplicity itself. For the purposes of this section, we'll assume that you are using the kernel source RPM for Fedora Core 3, `kernel-2.6.10-1.770_FC3.src.rpm`. If you're following along at home, you can download it from `distro.ibiblio.org` FTP site. At the time this was written, it was in the directory `/pub/linux/distributions/fedora/linux/core/updates/3/SRPMS/`.

To install the SRPM after you have downloaded it, execute the following command:

```
# rpm -ivh kernel-2.6.10-1.770_FC3.src.rpm
1:kernel ##### [100%]
```

CROSS-REFERENCE Chapter 30 covers all the gory details of using RPM and working with source RPMs.

To be clear, you *have not* just installed a new kernel. Rather, you have only installed the source code for the kernel that happened to be in RPM (well, *source* RPM) format. To see the fruits of your handiwork, browse through the directory `/usr/src/redhat/SOURCES`. You'll see a bundle of small and not-so-small files named `linux-mumble-patch` and a 35-MB archive file named `linux-2.6.10.bz2`. Without going into details here, all of those `linux-mumble.patch` files are modifications of one sort or another made to the kernel source code contained in the source code archive `linux-2.6.10.bz2`. This kernel source code archive is the kernel source as released by Linus Torvalds. The patch files are modifications made by Red Hat.

To turn this mess into a kernel, execute the following commands and then take a coffee break, because it might take a while to complete. The slower your system, the longer your coffee break.

```
# cd /usr/src/redhat/SPECS
# rpmbuild -bb kernel-2.6.spec --target=i686
Building target platforms: i686
Building for target i686
Executing(%prep): /bin/sh -e /var/tmp/rpm-tmp.65213
+ umask 022
+ cd /usr/src/redhat/BUILD
...
Wrote: /usr/src/redhat/RPMS/i686/kernel-2.6.10-1.770_FC3.root.i686.rpm
Wrote: /usr/src/redhat/RPMS/i686/kernel-smp-
2.6.10-1.770_FC3.root.i686.rpm
Wrote: /usr/src/redhat/RPMS/i686/kernel-debuginfo-
2.6.10-1.770_FC3.root.i686.rpm
Executing(%clean): /bin/sh -e /var/tmp/rpm-tmp.47185
+ umask 022
+ cd /usr/src/redhat/BUILD
+ cd kernel-2.6.10
+ rm -rf /var/tmp/kernel-2.6.10-1.770_FC3.root-root
+ exit 0
```

When the process completes, you *still* haven't installed new kernel. The output of this process is a set of binary RPMs, one or more of which you *can* install. The three lines that begin with `Wrote` show the RPMs you just created (which you can find in `/usr/src/redhat/RPMS/i686`):

- `kernel-2.6.10-1.770_FC3.root.i686.rpm`
- `kernel-smp-2.6.10-1.770_FC3.root.i686.rpm`
- `kernel-debuginfo-2.6.10-1.770_FC3.root.i686.rpm`

Most people will install the first RPM, `kernel-2.6.10-1.770_FC3.root.i686.rpm` using the following command:

```
# cd /usr/src/redhat/RPMS/i686
# rpm -ivh kernel-2.6.10-1.770_FC3.root.i686.rpm
Preparing... ##### [100%]
 1:kernel ##### [100%]
```

If you have an SMP system, install the second RPM using the following command:

```
# cd /usr/src/redhat/RPMS/i686
# rpm -ivh kernel-smp-2.6.10-1.770_FC3.root.i686.rpm
Preparing... ##### [100%]
 1:kernel-smp ##### [100%]
```

If you are a kernel developer or intend to be debugging the Linux kernel, install the third RPM in the list using the following command:

```
# cd /usr/src/redhat/RPMS/i686
# rpm -ivh kernel-debuginfo-2.6.10-1.770_FC3.root.i686.rpm
Preparing... ##### [100%]
 1:kernel-debuginfo ##### [100%]
```

If you are a merely mortal user, *do not* install the debug kernel. The debug kernel runs significantly slower than standard kernels do and contains code that actively attempts to reveal hidden, subtle kernel bugs. As a result, it is potentially unstable and, therefore, unsuitable for use on a production system or by people not prepared to fix the problems it reveals. Some bugs are relatively harmless, but other bugs might crash the system, scrambling your file systems, or making demons fly out of your nose.

Using Pristine Kernel Source

The method most long-time Linux users prefer for upgrading and customizing the kernel is to work with the pristine (unmodified) source code available from the various kernel archive sites scattered around the Internet. Why? Each major Linux vendor, including Red Hat, applies patches to the kernel source code that support the hardware of their strategic partners, to implement features requested by customers but not yet in the “official” kernel, and to differentiate themselves from their competitors. Linux distribution projects, such as Fedora Core, likewise make patched or otherwise modified kernels available. There is nothing wrong with this practice because Linux is open-source software, but it does have the unfortunate side effect of causing the source code distributed by vendors and distribution projects to diverge from the source code Linus maintains. As a result, applying patches produced by Linus and his colleagues becomes a dicey affair because sometimes, patches prepared by Red Hat or the Fedora Project conflict with patches prepared by the kernel developers. See the section titled “Patching the Kernel” to learn how and why to use kernel patches to upgrade the source code.

The primary site for the kernel source code is www.kernel.org (see Figure 27-1).

The main kernel archive site is always busy, especially after a new release, so you are better off using one of its many mirrors throughout the world. Often, you can find one close to you because most countries have at least one mirror and some, like the United States and the United Kingdom, have many. In fact, the Linux Kernel Archives Mirror System currently consists of 116 sites in 47 countries or territories. Another 84 countries or territories are supported by download sites situated in other countries.

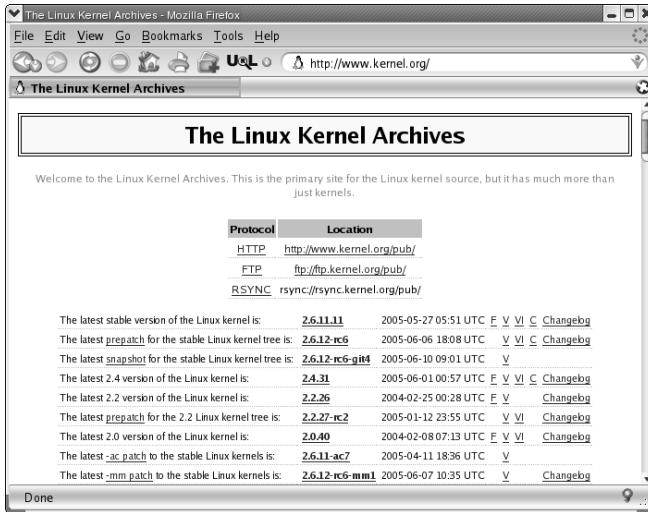


Figure 27-1 The Linux Kernel Archives home page.

To locate a mirror near you, point your Web browser at `www.kernel.org/mirrors`, scroll down the alphabetically ordered list of countries, and click your country name to view a list of mirror sites in your country. Figure 27-2, for example, shows part of the list of HTTP mirror sites in the United States.

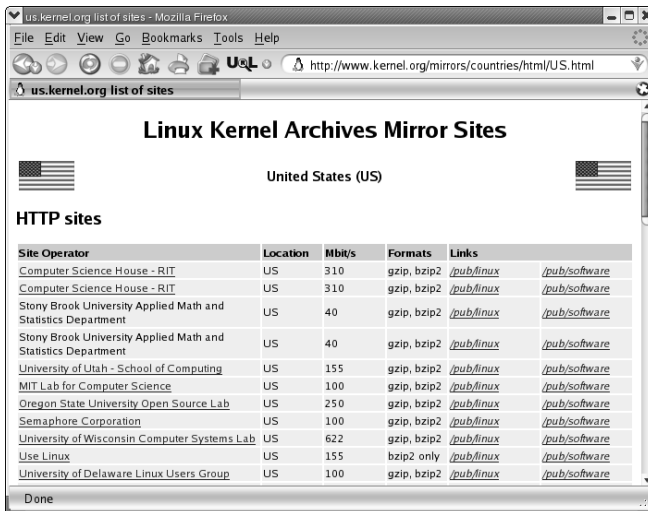


Figure 27-2 Linux kernel archive mirror sites in the United States.

The kernel's archive mirror system is set up so that for each two-letter country code you can simply use the hostname `http://www.country.kernel.org/` or `ftp.country.kernel.org` to reach a mirror supporting that specific country. For example, in the United States, you would use the URL `www.us.kernel.org` in your Web browser. Each mirror has a full archive of `/pub/linux`, the top-level kernel source directory, but it might not carry the source code in both `gzip` and `bzip2` compression formats. The `bzip2` format takes less time to download than `gzip` format, but takes longer than `gzip` format to decompress.

After locating an archive that is near you, in network terms, download the desired file using your Web browser or an FTP client. By way of example, the instructions that follow assume you use the standard FTP client in a terminal window, such as an `xterm`.

1. Change directories to a directory to which you have write permission, for example, your home directory:

```
$ cd ~
```

2. Open an FTP session to the archive site you selected and log in as the anonymous user using your email address as the password:

```
$ ftp ftp.us.kernel.org
Connected to ftp.us.kernel.org.
220 mirror.services.wisc.edu FTP server ready.
User (ftp.us.kernel.org:(none)): ftp
331 Anonymous login ok, send your complete email address as your
password.
Password:
230 Anonymous access granted, restrictions apply.
```

3. Change directories to the FTP server's kernel source code directory. The exact location of this directory varies from mirror to mirror, so you might have to use the `ls` command to locate it.

```
ftp> cd /pub/linux/kernel
```

4. Change directories to the `v2.6` directory:

```
ftp> cd v2.6
```

5. Execute the following command to see a listing of the full source code trees for the 2.6 kernel series (the listing is truncated to preserve space):

```
ftp> ls linux-2.6.*
...
-rw-r--r--  1 mirror  mirror  37099602 Apr  7 19:21 linux-
2.6.11.7.tar.bz2
-rw-r--r--  1 mirror  mirror           248 Apr  7 19:21 linux-
2.6.11.7.tar.bz2.sign
```

```
-rw-r--r-- 1 mirror mirror 46585077 Apr 7 19:21 linux-  
2.6.11.7.tar.gz  
-rw-r--r-- 1 mirror mirror 248 Apr 7 19:21 linux-  
2.6.11.7.tar.gz.sign  
-rw-r--r-- 1 mirror mirror 248 Apr 7 19:21 linux-  
2.6.11.7.tar.sign  
...
```

6. Identify the file to download. For this demonstration, download the 2.6.11.7 archive file in bzip2 format, that is, `linux-2.6.11.7.tar.bz2`.
7. Make sure that you are in binary download format:

```
ftp> binary
```

8. Use the following commands to download the archive file and its PGP signature file (`linux-2.6.11.7.tar.bz2.sign`):

```
ftp> get linux-2.6.11.tar.bz2
```

```
ftp> get linux-2.6.11.tar.bz2.sign
```

9. Close the FTP session:

```
ftp> bye
```

After you have the archive file, verify the file's integrity and unpack it as described in the next section.

Verifying and Unpacking the Archive

Before you unpack the archive, you should check its signature to make sure that it has not been tampered with. Files placed in the Linux Kernel Archives are OpenPGP-signed, and you can use these digital signatures to prove that files you have downloaded from the kernel archive site really originated at the Linux Kernel Archives. The current Linux Kernel Archives OpenPGP key is always available from www.kernel.org/signature.html.

The first step is to import the Linux Kernel Archive key. With an active Internet connection, execute the following command:

```
# gpg --keyserver wwwkeys.pgp.net --recv-keys 0x517D0F0E  
gpg: key 517D0F0E: public key "Linux Kernel Archives Verification Key <ftpadmin@  
kernel.org>" imported  
gpg: Total number processed: 1  
gpg: imported: 1
```

This step adds the Linux Kernel Archive key to root's public key ring. Next, change directories to the directory in which you downloaded the source files and execute the following commands, again as the root user, to verify the file signature:

```
$ gpg --verify linux-2.6.11.7.tar.bz2.sign linux-2.6.11.7.tar.bz2
gpg: Signature made Thu 07 Apr 2005 03:30:08 PM EDT using DSA key ID 517D0F0E
gpg: Good signature from "Linux Kernel Archives Verification Key
<ftpadmin@kernel.org>"
```

Replace the filenames used in the preceding command with filenames that reflect the files you downloaded. As long as you see the two lines of output shown in the example (beginning with `gpg:`), the file is authentic and has not been tampered with or modified. Because you probably have not added a trusted path to the archive verification key, this command probably also generates the following error message, which you can safely disregard:

```
gpg: checking the trustdb
gpg: no ultimately trusted keys found
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the
owner.
Primary key fingerprint: C75D C40A 11D7 AF88 9981 ED5B C86B A06A 517D
0F0E
```

Now you are ready to unpack the archive. If you downloaded a `bzip2` format archive file, execute the following command:

```
$ bunzip2 -c linux-2.6.11.7.tar.bz2 | tar -xf -
```

This command decompresses the archive file using `bunzip2` and pipes the output to the `tar` command, which extracts the archive. The operation might take some time to complete because `bzip2` compression and decompression takes longer than the more familiar and faster `gzip` compression. If you downloaded the `gzip` formatted archive, the proper command to use is:

```
$ gunzip -c linux-2.6.11.7.tar.gz | tar -xf -
```

The result of either command is the same: a new directory named `linux-2.6.11.7` in the directory in which you decompressed and extracted the archive that contains version 2.6.11.7 of the Linux kernel source code.

Patching the Kernel

If you have already downloaded the main source code tree, you can save both bandwidth and time by downloading and applying patches. Patches contain only changes to the underlying files from one kernel version to the next. For example, if you downloaded the 2.6.11.6 kernel source code tree, you do not need to download the 2.6.11.7 source code, only the patch, which is named, in this case, `patch-2.6.11.7.bz2`. If, alternatively, you have the source code for

version 2.6.11.3, you need to download four patches (patch-2.6.11.4.bz2, patch-2.6.11.5.bz2, patch-2.6.11.6.bz2, patch-2.6.11.7.bz2) and apply them in sequential order. The following procedure illustrates the process, assuming you have already downloaded the necessary patch and signature files and have verified their integrity using the method just described.

To apply the patches, change directories to the directory in which you unpacked the kernel source code. So, for example, if you unpacked the kernel source code in your home directory, change to that directory. Next, execute the following command for each patch file, in sequential order:

```
$ bunzip2 -c patch-2.6.11.N.bz2 | patch -p0
```

Replace N with the patch number of each patch you want to apply. For this example, suppose you are patching from the kernel version 2.6.11.6 to 2.6.11.7. You need to apply the patch that takes you to version 2.6.11.7. So, you execute the following command:

```
$ bunzip2 -c patch-2.6.11.7.bz2 | patch -d linux-2.6.11.6 -p1
patching file Makefile
patching file arch/ia64/kernel/fsys.S
patching file arch/ia64/kernel/signal.c
patching file arch/ppc/oprofile/op_model_fsl_booke.c
...
patching file net/netrom/nr_in.c
patching file net/xfrm/xfrm_state.c
patching file sound/core/timer.c
patching file sound/pci/ac97/ac97_codec.c
```

The exact list of filenames varies from patch to patch, and some patches change more files than other patches do. The result, however, is a kernel source tree updated to the latest version.

Finally, execute the following two commands to ensure that you are working with an unblemished source code tree:

```
$ mv linux-2.6.11.6 linux-2.6.11.7
```

```
$ cd linux-2.6.11.7
```

```
$ make mrproper
```

The mv command renames the directory from linux-2.6.11.6 to linux-2.6.11.7 to reflect the correct kernel version. The command make mrproper removes any detritus remaining from previous kernel compiles — if you are starting from scratch, you can skip this step, but it does no harm to include it, either. You are ready, finally, to configure the kernel.

Configuring the Kernel

As remarked at the beginning of this chapter, the kernel configuration process has changed significantly. Long-time Linux users will be pleased to know that `make config`, `make oldconfig`, and `make menuconfig` still work as they always have, albeit with more and different options. Those who have used `make xconfig`, however, are in for a bit of a surprise. The old Tk-based configuration tool has been replaced by kernel configuration interfaces based on Qt and GTK+. When you start the X-based configuration process using `make xconfig`, the kernel configuration tool tries to load a Qt-based tool (named `qconfig`). If the Qt toolkit isn't found, the process stops with an error. In this case, you should try `make gconfig` to invoke the GTK+-based kernel configuration tool. If *that* fails, then you'll need to use the ncurses-based configuration tool by executing the command `make menuconfig`.

In addition to the new interfaces for the graphical configuration tool, the kernel configuration process includes more useful targets and is generally more user-friendly. Table 27-1 lists the available kernel configuration targets available in the new configuration system.

Table 27-1 Kernel Configuration Targets

TARGET	DESCRIPTION
<code>allmodconfig</code>	Creates a new configuration file selecting modules for all options that can be built as modules
<code>allnoconfig</code>	Creates a new (minimal) configuration file by answering No to all possible configuration options
<code>allyesconfig</code>	Creates a new configuration file by answering Yes to all possible configuration options
<code>defconfig</code>	Creates a new configuration file using the default options
<code>gconfig</code>	Updates the kernel configuration file using a GTK+-based graphical interface
<code>menuconfig</code>	Updates the kernel configuration file using an ncurses-based text-mode GUI to present configuration options
<code>oldconfig</code>	Updates the kernel configuration file using the current configuration, prompting only for previously unseen options
<code>xconfig</code>	Updates the kernel configuration file using a Qt-based graphical interface

There are many more kernel build targets than those listed in Table 27-1. A new kernel build target is `help` (as in, `make help`), which shows all of the possible targets you can use.

To use one of these targets, or options, change directories to the top level of the kernel source tree and type **`make target`**, replacing `target` with the option you want to use. For example, to use the `allnoconfig` option, type **`make allnoconfig`** and press Enter.

The `allmodconfig` target is handy if you want a small, statically linked kernel with all of your devices and subsystems built as modules. However, if you do use the `allmodconfig` target, make sure that support for your root file system, `/`, is either included in your `initrd` (see “Creating an Initial RAM Disk” later in this chapter) or is compiled into your kernel, or you won’t be able to boot the kernel because the kernel won’t know how to access the root file system.

The `oldconfig` target is especially handy when you upgrade to newer kernel releases. Suppose, for example, that you have a custom compiled 2.6.11.6 kernel but you want to upgrade to 2.6.11.7. After applying the patch, you want to rebuild the kernel but don’t want to have to go through the entire kernel configuration process again. This is where the `oldconfig` target helps. After you upgrade the kernel, run `make oldconfig`. The configuration process preserves the existing configuration choices and only prompts you to configure new options added in the upgrade from 2.6.11.6 to 2.6.11.7.

It is unnecessary and ill-advised to build your kernel in the traditional kernel location, `/usr/src/linux`. Accordingly, the rest of this chapter assumes that you build the kernel in a subdirectory of your home directory, for example, `/home/kwall/kernel`. Using `/usr/src/linux` is a bad idea because the kernel headers against which `glibc`, the system C library was built, reside in `/usr/src/linux`, so when you install updated kernel source code, particularly header files, you run the risk of having kernel headers that are out of sync with the C library’s object files. The kernel headers *must* match the C library files or bedlam will ensue.

As a practical matter, it is easier to build the kernel in a `nonsystem` directory because you don’t have to mess with running as root. That’s right, you don’t have to be root to build the kernel. You only need root access for the postbuild steps, which include installing the kernel, installing the modules, and updating the GRUB configuration file.

Selecting a Kernel Configuration File

If you are unfamiliar with kernel configuration, you might want to consider using an existing kernel configuration file as a starting point for your custom

configuration. If you are using kernel source code from kernel.org or one of its mirrors, you have a couple of options. For most architectures, you can look for files named `defconfig` or `defconfig.mumble` in the `arch` directory hierarchy. Each `defconfig` file represents a default configuration (hence the name, `defconfig`) with a standard, reasonably well-tested set of features and sane defaults for the specified architecture. As of kernel version 2.6.11.7, there were 210 `defconfig` files, some of which are:

```
arch/mips/defconfig
arch/i386/defconfig
arch/sparc/defconfig
arch/um/defconfig
arch/sparc64/defconfig
arch/m68knommu/defconfig
arch/sh/defconfig
arch/cris/arch-v10/defconfig
arch/cris/defconfig
arch/arm26/defconfig
arch/m68k/defconfig
arch/ia64/defconfig
arch/alpha/defconfig
arch/ppc64/defconfig
arch/parisc/defconfig
arch/h8300/defconfig
arch/m32r/oaks32r/defconfig.nommu
arch/m32r/mappi/defconfig.up
arch/m32r/mappi/defconfig.smp
arch/m32r/mappi/defconfig.nommu
arch/m32r/opspout/defconfig.opspout
arch/m32r/m32700ut/defconfig.m32700ut.smp
arch/m32r/m32700ut/defconfig.m32700ut.up
arch/m32r/defconfig
arch/m32r/mappi2/defconfig.vdec2
arch/sh64/defconfig
arch/x86_64/defconfig
arch/s390/defconfig
```

If your architecture or platform has a `defconfig`, you can use it by executing the following command in the top-level kernel source directory:

```
$ make defconfig
```

This command creates a new configuration file using the defaults in the `defconfig` file for your architecture. This creates a known starting point for your customized configuration file.

Some processor architectures have multiple platforms. For these systems, you're looking for a configuration file that is closest to your system. The place to look is `arch/ARCH/configs`, where (as of kernel 2.6.11.7) `ARCH` is one of

- **arm** — ARM processors
- **ia64** — Intel IA64 (Itanium) processors
- **m68k** — Motorola 68xxx processors
- **mips** — MIPS processors
- **parisc** — HP PA-RISC processors
- **ppc** — PowerPC 32-bit processors
- **ppc64** — PowerPC 64-bit processors
- **sh** — SuperH processors
- **sh64** — SuperH 64-bit processors

For example, the Intel IA-64 architecture has six different configuration files because there are (as version 2.6.11.7), six supported IA-64 platforms:

- **ia64/configs/bigsur_defconfig** — A configuration suitable for systems compliant with the IA-64 Developer Interface Guide (DIG)
- **ia64/configs/sim_defconfig** — A configuration suitable for the IA-64 simulator
- **ia64/configs/sn2_defconfig** — A configuration suitable for SN2-based SGI Altix systems
- **ia64/configs/tiger_defconfig** — Another configuration suitable for DIG-compliant systems
- **ia64/configs/zx1_defconfig** — A configuration suitable for HP's ZX1 and SX1000 IA-64 systems
- **ia64/defconfig** — A "generic" configuration suitable for any supported IA-64 system

To use one of the platform-specific configuration files, copy it into the top-level kernel source directory as `.config` and then execute the command **make oldconfig** to create a starting point for further customization. For example, if you are configuring a kernel for an HP SX1000 IA-64 system (lucky you!), execute the following commands:

```
$ cp arch/ia64/configs/zx1_defconfig .config
$ make oldconfig
```

Configuring the Kernel with xconfig

To start configuring a kernel, change directories to the top level of your kernel source directory, type **make xconfig**, and press Enter. After a few seconds, you should see a screen resembling Figure 27-3.

For each configuration option, a blank box indicates that the corresponding feature is disabled; a check mark indicates that the corresponding feature is enabled; a dot indicates that it will be compiled as a module. A module, specifically, a *loadable kernel module*, or LKM, is a section of kernel code that can be inserted into and removed from the kernel dynamically (while the system is running) without needing to restart the system. LKMs are used to provide drivers and other kernel functionality that isn't always needed and so doesn't need to be compiled into the kernel. Compiling drivers as LKMs rather than compiling them into the kernel makes the resulting kernel smaller, use less memory, and load faster.

To change an option, click the box to cycle through the three states.

In some cases, you might not see an option, such as a device driver or a feature that should be present. If this occurs, select Option ⇄ Show All Options. Currently, xconfig does not include a cross reference to help you identify which options depend on others. However, you can view *some* dependency information by selecting Option ⇄ Show Debug Info. You can use this limited dependency information (and the help text, which is visible even for disabled features) to find the required options or features manually.

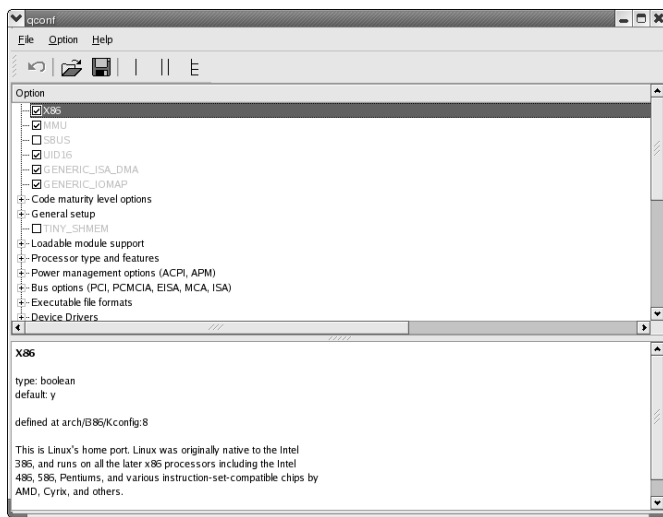


Figure 27-3 Viewing the xconfig kernel configuration tool.

To get the most information about various kernel configuration options, select Option ⇄ Show Name to show the name of the kernel configuration option as it appears in the configuration file, Option ⇄ Show Data to show the value of the kernel configuration options, and Option ⇄ Show Range to show the range of possible values for kernel configuration options.

To save your changes, click the floppy disk icon on the toolbar to save the configuration to the default file, `.config`, in the kernel directory (or select File ⇄ Save from the menu. To save your changes to a different file, select File ⇄ Save As from the menu and type a filename in the dialog box. You can load an existing configuration by clicking the folder icon on the toolbar or by selecting File ⇄ Load from the menu and selecting the configuration file you want to use.

Configuring the Kernel with menuconfig

If you prefer to use a text-based configuration tool, type **make menuconfig** to configure the kernel configuration using an ncurses-based text menu interface. The configuration options are the same as those provided by `xconfig`, but the interface itself is slightly different. The initial screen resembles Figure 27-4.

The following list shows the keystrokes you can use:

- Use the arrow keys to move the blue highlight cursor up and down the menu.
- Highlighted letters indicate hotkeys.
- To select an item, press **Enter**.
- Submenus are indicated by `--->`.
- To enable a configuration option, press **y** or the spacebar. Pressing the spacebar repeatedly will cycle through all of the possible options for a single configuration item.
- To disable a configuration option, press **n**.
- Configuration items preceded by angle brackets (`<>`) can be built as modules.
- Press **m** to build options as modules.
- Press **?** or use the Help button to view the help text for a given option.
- Press the Tab key to move the cursor from the menu to the buttons at the bottom and to move between the buttons.

For example, use the arrow key to move the cursor down to the Processor type and features `--->` selection. Then press Enter to open the submenu, which is shown in Figure 27-5.

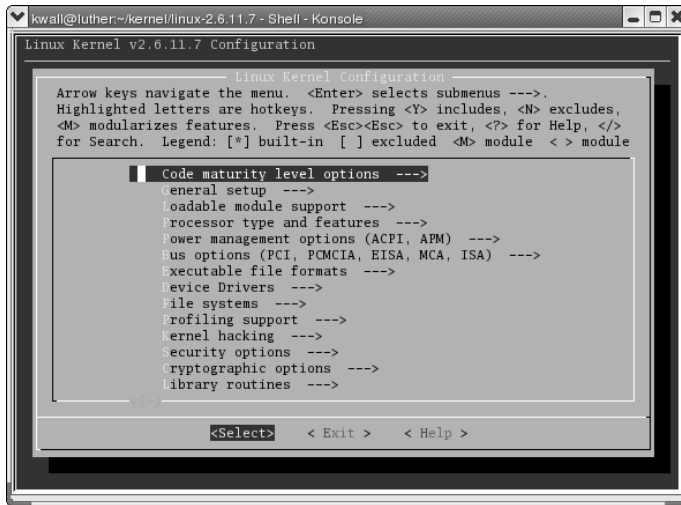


Figure 27-4 Viewing the menuconfig kernel configuration tool.

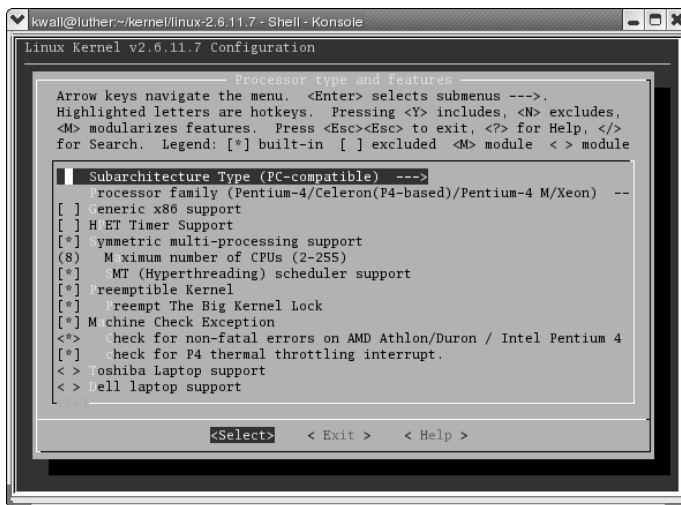


Figure 27-5 The Processor type and features submenu.

The Generic x86 support, HPET Timer Support, and Machine Check Exception options have * next to them, meaning these options have been enabled. Most of other options have not been included or enabled, but Toshiba Laptop support and Dell laptop support have <M>, meaning that support for these features will be built as loadable modules. The Check for nonfatal errors on AMD Athlon/Duron/Intel Pentium option has empty <>, meaning that this support can be built as a modules (by pressing m) or it can be built into the kernel by typing y or pressing the spacebar.

Many of the submenus have submenus of their own. In Figure 27-5, for example, the Processor family (Pentium Pro) ---> option indicates the existence of a subsubmenu, as it were. If you highlight this option and press **Enter**, you'll see Figure 27-6, the Processor Family menu.

When you have completed the configuration process, use the Exit button to exit the configuration menus. Before it exits, `menuconfig` gives you the option to save the changes, as shown in Figure 27-7.

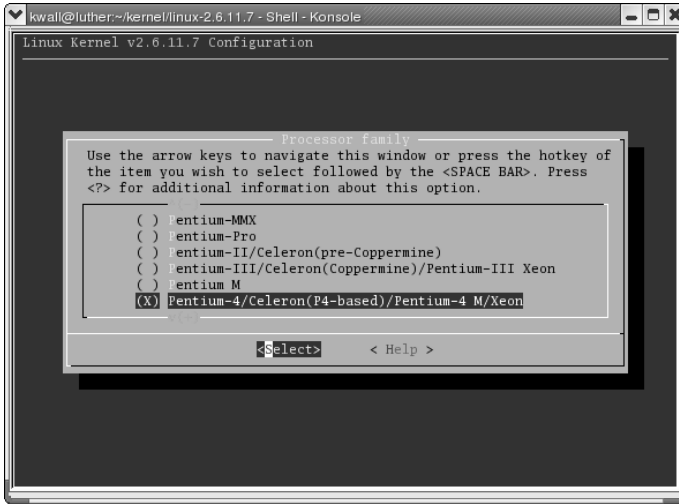


Figure 27-6 The Processor family submenu.

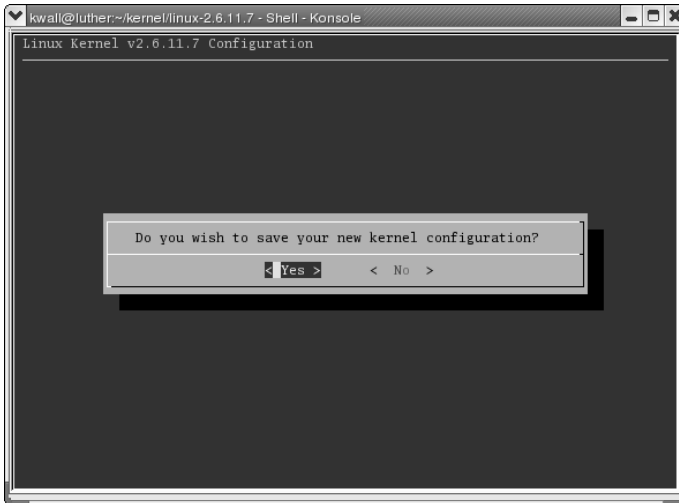


Figure 27-7 Saving the kernel configuration in `menuconfig`.

Highlight the Yes button, and press **Enter** to save the new configuration (or No to abandon it) and exit `menuconfig`. See the next section for information about the kernel's various configuration options.

Reviewing the Configuration Options

The following sections describe many of the configuration options available as of kernel version 2.6.11.7. If you need additional information beyond what is provided by the help text in the tool itself or in this chapter, the kernel ships with a large amount of documentation in the form of text files in the Documentation subdirectory of the kernel directory. As a matter of convenience, for the rest of this chapter, the phrase “the kernel directory” refers to the top level of the directory in which you unpacked the kernel source code.

Code Maturity Level Options

Some kernel features and drivers might not be fully tested. The Code maturity level options section enables you to choose whether you even see options that are known or suspected to have problems. If you enable Prompt for development and/or incomplete code/drivers, you will see the associated configuration options elsewhere in the interface. Clearing this check box ensures that you will not be prompted for them. This option also makes obsolete drivers available, that is, old drivers that have been replaced by something else or that might be removed in a future kernel release. Disable this option on production systems or if you don't want to deal with broken devices drivers. Enable this option if you enjoy living on the edge or if you need support for a device or a kernel feature for which stable drivers are not yet available.

General Setup

The General setup options enable allow you to set global kernel characteristics and enable kernel features that don't fit neatly into the other categories. The Local version - append to kernel release option enables you to add a string to the end of the kernel release that identifies your custom kernel. This string will be shown when you execute the command `uname -r`. It is a good idea to use this when you build custom kernels so that you can quickly distinguish between the stock kernels provided by Red Hat or the Fedora Project and your own. To set this value, double-click the label to create a small, editable text box under the Value column of the display. (See Figure 27-8.)

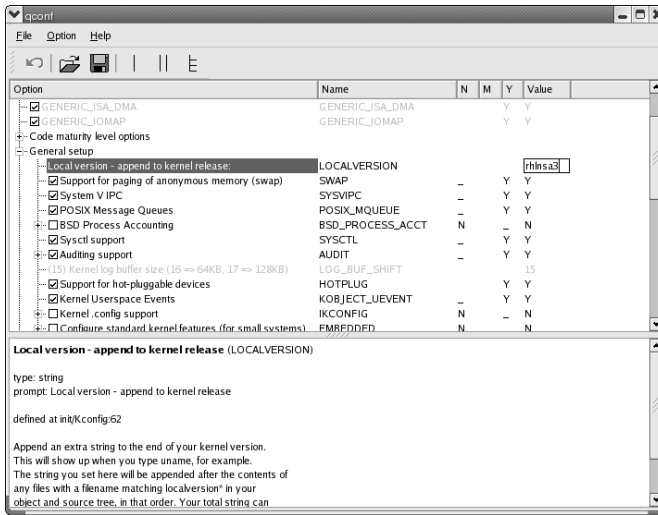


Figure 27-8 Changing a string value in xconfig.

Type your local version string and then press Enter.

If you are configuring a kernel for a desktop or server system, make sure that Support for paging of anonymous memory (swap) is enabled because this option enables the kernel's use of virtual memory, which allows the kernel to use disk-based devices and swap files as RAM. Similarly, most systems on which you will run Fedora Core or RHEL should have System V IPC enabled. IPC, which stands for Interprocess Communication, allows programs running on a Linux system to share data using message queues, semaphores, or shared memory. POSIX Message Queues are a variant of the message queues that provide functionality not available in standard System V message queues. You can usually disable this option because few Linux applications use POSIX message queues.

BSD Process Accounting enables programs to request the kernel to maintain process accounting information. This bookkeeping information includes:

- **Average memory usage** — The average amount of memory the process used
- **Blocks read and written** — The number of disk blocks read or written
- **Chars transferred** — The number of characters read or written
- **Command name** — The command line used to start the process
- **Controlling terminal** — The terminal (TTY or PTY) on which the process ran, if any
- **Elapsed time** — The total time the process ran
- **Exit code** — The process's exit code

- **Flags** — If the process did one or more of the following:
 - Executed a `fork()` system call without also calling one the `exec()` family of system calls
 - Used root privileges (perhaps via `su`)
 - Dumped core because of a segmentation fault
 - Was killed by a signal
- **Major page faults** — The number of times the CPU requested data on behalf of the process that had to be read from disk
- **Minor page faults** — The number of times the CPU requested data on behalf of the process that was in the CPU's level 1 or level 2 cache
- **Number of swaps** — The number of times the process used swap memory
- **Padding bytes** — Unused space to allow the data structure to grow over time
- **Process creation time** — The time at which the process started
- **Real GID** — The GID (group ID) under which the process ran
- **Real UID** — The UID (user ID) under which the process ran
- **System time** — The time the process spent executing kernel mode code
- **User time** — The time the process spent executing user mode code

A kernel configured to use BSD-style process accounting does nothing with these statistics except accumulate them and save them to a file as monitored processes exit. Userspace programs are required to interpret the accounting information. See the `sa(1)`, `accton(1)`, `ac(8)`, and `acct(5)` manual pages. Do not enable the BSD Process Accounting version 3 file format option because the version 3 file format is incompatible with previous versions of the process accounting file format, and the tools for processing version 3 file formats are still under development.

You should definitely enable `Sysctl` support because it allows you to use the `sysctl` command, which gives you a means to change many of the kernel's characteristics at runtime without needing to restart the system. For more information about `sysctl`, see the `sysctl` manual page.

Only enable Auditing support if your server will be deployed in a high-security environment and you will be using SELinux. (See Chapter 33.) Other features you can disable include Kernel `.config` support, which embeds the kernel configuration file into the kernel, and `Configure` standard kernel features (for small systems), which provides configuration options for kernels that will be used in resource-constrained environments such as embedded devices (think cellular phones, set-top devices).

Loadable Module Support

The Loadable module support options enable you to control how (and if) the kernel will support loadable modules. Kernel modules are small pieces of compiled code that can be inserted into a running kernel to add support for infrequently used hardware devices and to enable support for other types of functionality. Likewise, modules can also be *unloaded*, removed from the kernel, when they are no longer needed. If you want to use modules, check the Enable loadable module support check box. You should also enable Module unloading and Forced module unloading. If you forcibly unload a module, you run the risk of destabilizing or outright crashing your system, but there are situations in which you might need to forcibly unload a module, and it is only possible to do so if Forced module unloading is enabled.

To further simplify module handling, enable Automatic kernel module loading. This option makes it possible for the kernel itself to load (some) modules without requiring your intervention using `kmod`, the kernel module autoloader. You might still need to load other modules manually (using the `modprobe` command), but at least some of the modules will load automatically. For more information about `kmod`, read `Documentation/kmod.txt` to learn how to configure `kmod`, the kernel module autoloader.

Module versioning support (EXPERIMENTAL) makes it possible in some situations to use the same kernel module across multiple kernel versions. While this is a handy feature, you're usually better off recompiling your modules if the kernel version changes, so disable module versioning unless you are building a kernel that will be deployed on multiple systems or you want to have a chance to reuse existing binary-only modules, such as those provided by framebuffer vendors. Source checksum for all modules is really a kernel developer tool that you won't need unless you are, you guessed it, a kernel hacker.

If you are unsure what to do, disable Source checksum for all modules and enable the following options:

- Enable loadable module support
- Module unloading
- Forced module unloading
- Module version support (EXPERIMENTAL)
- Automatic kernel module loading

Processor Type and Features

The kernel features in this section enable you to customize the kernel for the specific processor (and, in some cases, processor features) in your system. The information you provide here is used to configure the kernel and to configure

the build process itself to generate code that takes advantage of specific CPU features. If you need to compile a kernel for another system, select the CPU for the target system, not the host (the system on which you are building the kernel). If you need to compile a kernel that can run on any Intel x86 or x86-compatible system, click the 386 radio button. Otherwise, use the following list to select the appropriate CPU for your system. The help text for each option explains better than we can the specific features that the kernel supports for each CPU.

- 386 for the AMD/Cyrix/Intel 386DX/DXL/SL/SLC/SX, Cyrix/TI86DLC/DLC2, UMC 486SX-S and NexGen Nx586 CPUs
- 486 for the AMD/Cyrix/IBM/Intel 486DX/DX2/DX4 or SL/SLC/SLC2/SLC3/SX/SX2 and UMC U5D or U5S CPUs
- 586/K5/5x86/6x86/6x86MX for generic Pentium, AMD K5, and Cyrix 5x86, 6x86, and 6x86MX (Cyrix 6x86 with multimedia extensions) CPUs
- Pentium-Classic for the original Intel Pentium processors
- Pentium-MMX for the Intel Pentium MMX
- Pentium-Pro to enable support for Intel's Pentium Pro extended instructions
- Pentium-II/Celeron (pre-Coppermine) for the Intel Pentium Pro/Celeron/Pentium II
- Pentium-III/Celeron (Coppermine)/Pentium-III Xeon
- Pentium M for systems (usually notebooks) with Pentium M chips (Pentium 4 M CPUs are separately support)
- Pentium-4/Celeron (P4-based)/Pentium-4 M/Xeon for the Intel Pentium 4, Xeon, and Pentium 4 M CPUs
- K6/K6-II/K6-III for the AMD K6, K6-II and K6-III (also known as K6-3D)
- Athlon/Duron/K7 for the AMD K7 family (Athlon/Duron/Thunderbird)
- Opteron/Athlon64/Hammer/K8
- Crusoe for the Transmeta Crusoe series
- Efficeon for the Transmeta Efficeon CPU
- Winchip-C6 for original IDT Winchip CPU
- Winchip-2 for IDT Winchip 2 CPU
- Winchip-2A/Winchip-3 for IDT Winchips with 3dNow! capabilities
- CyrixIII/VIA-C3 for VIA Cyrix III or VIA C3 (pre-model 9) CPUs
- VIA-C3-2 (Nehemiah) for C3 model 9 and later CPUs

New motherboard chipsets have an improved high-frequency timer in addition to the older 8254 Programmable Interval Timers (PITs). In fact, the newest motherboards no longer have 8254-based PITs. If you select HPET Timer Support *and* your system has a High Precision Event Timer (HPET), the kernel will use it for its internal time keeping. However, if you choose to use the HPET timer, you should also select Provide RTC interrupt so that the HPET timer will generate a real-time clock interrupt. HPET timers are designed specifically for OS kernels and other system software that need interrupts for thread schedulers, kernels, and multimedia timer servers. HPETs will eventually replace legacy 8254 PITs and RTC interrupt-driven hardware on all Intel-based systems.

The Preemptible Kernel option is the kernel feature in the 2.6 kernel about which most users care. By way of brief explanation, if the kernel is *preemptible*, certain portions of kernel mode code can be interrupted by user mode processes. The result is a more responsible system overall, even when the system is under a heavy load because user level processes, which can be deprived of CPU time on a busy system running a nonpreemptible kernel, are more likely to get CPU time on a preemptible kernel. This explanation glosses over some of the subtleties and fine points of OS scheduling, but accurately describes the practical effect of preemptibility from the end user's perspective. Just enable this option and be glad you can.

If you are configuring a kernel for certain Toshiba or Dell laptop computers, enabling the support option for those computers (Toshiba Laptop support and Dell laptop support) gives allows user mode programs access to system management functions available on those systems. You should enable this support. Similarly, you should enable either statically or as a module, Model-specific register support because doing so enables the kernel to access specific registers on various CPUs not ordinarily accessible. On the other hand, do *not* enable the BIOS Enhanced Disk Drive option for detecting the boot disk. It is experimental and you're better off specifying the boot disk in normal situations.

Unless you have more than 1 GB of physical RAM installed in your system, you can safely disregard the High Memory Support options. Otherwise, select the appropriate memory value. Most CPUs used in desktop Linux systems have math processors on-die, so don't bother with math emulation. If you use X with a reasonably modern frame buffer chip, enable MTRR (memory type range register) support so that the frame buffer and its X driver can map the frame buffer memory into main memory at the appropriate location.

The final option in this section, Use register arguments (EXPERIMENTAL) can result in significant performance gains because it causes GCC to generate code that passes the first three arguments to a function in CPU registers, rather than on the stack. Register-passed arguments require significantly less overhead for the CPU to access than stack-passed arguments, so you should use this feature if it doesn't cause problems for you.

Power Management Options

If you are configuring a kernel for a desktop system, you can jump ahead to the next section, “Bus Options,” because power management is not an issue for you. Notebook users will want to pay close attention to this section, however. If you enable Power Management Support, you can choose between APM (Advanced Power Management) support and ACPI (Advanced Configuration and Power Interface) support. ACPI is enabled by default if you don’t enable Power Management support, and you should use it unless you know that ACPI on your notebook is flaky or unreliable. ACPI is newer and more general than APM and is better supported in the current kernel.

Another reason you might want to enable power management is to take advantage of Software Suspend, the Linux version of the suspend-to-disk feature available on most notebook computer that are running, ahem, another operating system. It is still experimental, though, so if you can’t replace the data on your laptop, don’t use it.

CPU Frequency scaling enables the kernel to issue instructions that slows the CPU (actually, to reduce the CPU’s clock speed) in a notebook computer when it is not in use. The purpose, obviously, is to reduce power consumption and, thus, prolong battery life. If you enable this option, you will need to select the appropriate CPU so the kernel driver can issue the proper CPU commands. Notice that in many cases, CPU frequency scaling requires ACPI support, which is another reason to use ACPI instead of APM.

Bus Options

The bus options section of the kernel configuration is hardly a page-turning read. In short, if you need support for one of the listed data busses, enable that support. Some items do bear additional explanation, however. For example, more and more computer systems these days ship with only a PCI bus. Older motherboards might still have both ISA and PCI busses, but some new system boards combine PCI with the newer PCI bus, known as PCI-X.

Another subtlety of the bus option is that the ISA bus, while no longer accessed using an ISA slot, is still the underlying technology for the I2C bus, which is the data bus used by some hardware sensors. You might need to enable ISA bus support here if you have hardware sensors you want to use. The section on I2C covers hardware sensors in somewhat more detail.

Finally, a word on the PCI Hotplug Support options. These options provide support for inserting and removing PCI cards from a running system. In addition, the CardBus subsystem (formerly known as PCMCIA) and some docking stations use PCI hotplug. This is not the same as inserting and removing mice or keyboards or other peripherals from a running system. Device hot-plugging

WHAT IS PCI-X?

PCI-X stands for PCI extended, an enhanced PCI bus designed to be backward compatible with existing first generation PCI cards and expansion slots. Among other features, PCI-X increases PCI bus speeds from a maximum of 133 MBps (megabits/second) to as much as 1 GBps (gigabits/second). PCI-X, jointly designed by IBM, HP, and Compaq, was primarily motivated by the need to increase performance of high bandwidth devices, such as Gigabit Ethernet and Fibre Channel network connections and the interconnects between processors that are part of a cluster.

Just as the computing public began to understand and use PCI-X, the PCI Special Interest Group (PCI-SIG), the organization that develops and maintains the PCI standard, released PCI-X 2.0. PCI-X 2.0 is a new, higher-speed version of the conventional PCI standard, which supported signaling speeds up to 533 megatransfers per second (MTS). Revision 2.0 also adds two new bus speeds, PCI-X 266 and PCI-X 533, offering data transfer speeds up to 4.3 GBps, 32 times faster than plain vanilla PCI.

A second major feature of the PCI-X 2.0 specification is enhanced system reliability through the addition of error checking and correction (ECC) support to the PCI-X protocol header and payload. ECC is an important addition because it allows automatic single-bit error recovery and double-bit error detection, which in turn enable PCI-X 2.0 to stay current with respect to the latest networking and technology advances, such as iSCSI and InfiniBand.

is covered by a different kernel subsystem. Unless you have server class hardware or another type of system that has on on-board PCI hotplug controller, need to use CardBus, or have a laptop that requires hotplug support for proper docking station support, you probably don't need to enable PCI Hotplug Support.

Executable File Formats

Selecting executable file formats to support is simple. Stick with the defaults, enabling Kernel support for ELF binaries (Linux's native binary format) and Kernel support for MISC binaries, which allows you to execute arbitrary binary formats provided a module is available that supports the format. Even better, you can disable support for miscellaneous binaries and make your kernel a bit smaller. Chances are good you'll never miss it. On the other hand, you might want to enable a.out support as a module if you will be running old binaries provided by a third-party vendor or need to run any type of legacy application that does not use ELF format and for which you do not have the source code.

Device Drivers

The Device Drivers section is the longest section and has the most options. It is certainly too long to discuss in extensive detail in this chapter. The general advice is simple and obvious: if your system does not have a class of devices, don't include support for that class of devices in your kernel. Including support for devices or features you don't have or don't need only makes the kernel larger. Even if you build support for unavailable devices as modules that you never load into the running kernel, all you succeed in doing is taking up disk space and making kernel compilation last longer than it needs to.

Generic Driver Options

As a rule, you will want to enable all three options in this section. The first option selects only those kernel drivers for which firmware is available in the kernel tree (and, naturally, for which firmware is required). The second option disables building firmware at all; you will only need to use this option if a device vendor releases new firmware that must be built with the associated driver. The third option, Hotplug firmware loading support, allows loading firmware for hotplug devices whose associated drivers are built as kernel modules outside the kernel tree. You might not need it, but if you are not sure, build it as a module. See the section titled "Library Routines" for more information about in-tree and out-of-tree modules.

Memory Technology Devices

If you have some sort of device that uses flash memory chips or similar devices, enable the first option, Memory Technology Device (MTD) support and then enable support for the specific MTD device or technology you need. Notice that USB memory stick readers, CompactFlash cards, and MMC/SD cards do not appear in this list. USB memory stick readers are configured under the USB support option, CompactFlash support requires IDE support, and MMC/SD cards are configured under the MMC/SD Card support. You will come to these configuration sections shortly. If you don't need MTD support, disable all of these options and then move on to the Parallel port support item.

Parallel Port Support

You should enable both the Parallel port support and PC-style hardware options if you want to use a printer, Zip drive, or other device attached to your system's parallel port. If you have a computer that is less than four to five

years old, you might also want to enable the IEEE 1284 transfer modes option, which enables devices (and device drivers) to use the status read-back and device identification features available on modern parallel ports.

Plug and Play Support

Most modern PCs and PC BIOSes support the Plug and Play (PnP) protocol, so enable Plug and Play support. Similarly, if you have any legacy ISA bus devices, enable the ISA Plug and Play support feature. If you do not have ISA devices in your system, disable this option and proceed to the Block Devices section. If you have a PNP BIOS, enable Plug and Play BIOS support (EXPERIMENTAL). Likewise, enable Plug and Play ACPI support (EXPERIMENTAL) because most motherboards produced use ACPI for power management.

Block Devices

The Block devices options enable you to configure support for block devices such as disk drives, drives attached to the parallel port, RAID controllers (such as the Compaq SMART2 and Mylex RAID cards), and ramdisks. ramdisks are important if you need initrd (Initial RAM Disk). One common reason to use and initrd is to support booting from a SCSI disk, but any device setup or module pre-loading can be done from an initrd. A *block device* is so-named because I/O to such a device occurs in groups, or blocks, of bytes. A *character device*, on the other hand, reads and writes data one byte at a time.

At a bare minimum, enable Normal floppy disk support unless your system does not have a standard 3.5-inch floppy drive. If you have one of the drives or devices listed under Parallel port IDE device support, enable that option plus the option for the specific device (or protocol). If you do have a parallel port device you want to support, be sure to read the help text and note any conditions, qualifications, or caveats related to driver support for the selected device.

You might also want to enable the Loopback device support option. This feature allows you to treat a disk file as a block device. One common use for loopback devices is to mount an ISO image to verify its integrity before attempting to burn the ISO to a CD-R or CD-RW. Loopback devices are also often used to create initrds.

Speaking of initrds, unless you know for sure that you do not need ramdisk support, enable the ramdisk support option and the Initial RAM disk (initrd) support option. The easiest way to determine if you need initrd support is to execute the command `ls -l /boot`. If you see a filename that begins with `initrd`, such as `initrd-2.6.11-1.14_FC3.img`, then you need ramdisk and initrd support.

If you create a lot of CDs, you would do well to enable the Packet writing on CD/DVD media option and the change the Free buffers for data gathering from the default value of 8 to something larger. The purpose is to speed of the read/write process when creating CDs and DVDs, but each additional buffer requires an additional 64 KB of kernel memory. Think seriously before enabling the Enable write caching option. It might be prone to error at this time because the underlying code performs no significant error checking before writing cached data to the recording media.

The final group of block device options to consider is the IO Schedulers options. This option group makes it possible for you to select between three different I/O scheduling algorithms. The anticipatory scheduler is the default scheduler and is a good general-purpose scheduler. The major disadvantage is that the code is complicated compared to the other options. The deadline I/O scheduler is simpler and, while suitable for general desktop usage, is probably better suited for servers, especially those running databases. The CFQ scheduler, which is also good for general desktop use, shares I/O amongst all the running processes on a system. You can select an I/O scheduler at boot time by passing the parameter `elevator=scheduler` at the kernel boot prompt. For example, to use the deadline scheduler, pass the kernel command-line parameter

```
elevator=deadline
```

By way of recommendations, compile all three of these schedulers into the kernel and use the boot parameter to select them until you decide which one gives you the best performance.

ATA/ATAPI/MFM/RLL Support

You will almost certainly need to enable at least some of the options in ATA/IDE/MFM/RLL support section unless you have a SCSI-only system (in which case you should jump ahead to the next section, “SCSI Device Support”). In fact, anything that appears to be or that acts like an IDE disk, such as CompactFlash, requires IDE support. Enable Enhanced IDE/MFM/RLL disk/cdrom/tape/floppy support if you have any IDE devices that you want to use — if you want to boot from such a device, like an IDE disk drive, build support into the kernel. Scrolling down, you can disable and enable support for specific devices such as IDE CD-ROM and floppy drives, for bug fixes and workarounds for specific IDE chipsets that have known problems, such as the CMD640 or RZ1000 chipsets, and for a large number of specific chipsets and controllers. If you do not need such support, disable these options.

An important change from 2.4 to 2.6 is that SCSI emulation is no longer needed to enable CD writing applications to work. Rather, the Linux kernel's ATAPI drivers are finally capable of writing CDs. This means that if you have an IDE ATAPI CD-R or CD-RW, you don't have to enable SCSI emulation support (known colloquially by the name of the corresponding kernel module, `ide-scsi`), SCSI CDROM support, and SCSI generic support (the latter two are in the SCSI device support section, discussed next), resulting in a much smaller kernel and more efficient.

NOTE For the record, calling this option *SCSI emulation* has always been a misleading misnomer. It has really been *ATAPI emulation* in the SCSI driver.

SCSI Device Support

The SCSI device support section is broken into a number of subsections that permit you to configure the low-level drivers pertaining to specific SCSI devices and controllers. Take some time to review the information carefully, resorting to the help text if necessary. In particular, older SCSI controllers often need to have their I/O addresses and IRQ numbers specified in the kernel configuration dialog box. Most drivers for most new controllers can usually autodetect the I/O addresses and IRQ numbers.

If you want to use an Iomega Zip drive that attaches to a parallel port, you should enable SCSI device support, SCSI disk support, and then enable support for either IOMEGA parallel port (`ppa` — older drives) or IOMEGA parallel port (`imm` — newer drives) in the SCSI low-level drivers subsection. If your Zip drive came with a cable labeled “AutoDetect,” disable the `ppa` module and enable the `imm` module). The good news is that both the `ppa` and `imm` drivers enable you to use the Zip drive and a printer, just as you can with Windows.

If you need additional information on configuring and using Zip drives under Fedora Core or RHEL, see the README file for the `ppa` module, `ppa.txt`, in the kernel's Documentation directory. You will also find the Iomega Zip drive page at torque.net/~campbell/useful.

Old CD-ROM Drivers

If you need support for proprietary CD-ROM interfaces, such as the old Panasonic or Sound Blaster CD-ROMs, configure them appropriately using the Old CD-ROM drivers section. Do not use this box to configure SCSI or IDE CD-ROM drivers! You need this support only if you have one of the drivers listed in the dialog box, and only older systems should need this support. Most people will not need these drivers.

Multidevice Support

In `xconfig`'s Multi-device support section, you configure the kernel's support for software redundant array of inexpensive disks (RAID) and for logical volume management (LVM). Software RAID enables you to combine multiple hard disk partitions to create a single logical block device. It is referred to as *software RAID* because the kernel, rather than special hardware, implements the RAID functionality. LVM combines multiple devices in a volume group. A *volume group* is roughly analogous to a virtual disk. You can resize volume groups after creating them as your disk space capacity requirements change without having to go through painful disk partition manipulations and copy data around. Most people do not need multidevice support, which is disabled by default.

Fusion MPT Device Support

If you have an LSI Logic Fusion card, which can run both a high-performance SCSI connection and a LAN interface across 2 GHz Fibre Channel connections, you should enable support for it in this section. If the previous sentence is incoherent nonsense to you, you don't need Fusion MPT support and can continue with the FireWire support section.

IEEE 1394/FireWire Support

As you might know, FireWire, more formally known by its specification document, IEEE 1394, is a high-speed serial bus for connecting disks, digital cameras, scanners, and other external peripherals to your system. It is comparable in purpose to USB, but not as widely used on PC hardware as it is on Apple hardware. If you have a FireWire device you want to use, enable IEEE 1394 (FireWire) support and also enable support for the FireWire controller and/or device you have. If you don't need FireWire, disable it and continue to the next section of configuration options.

I2O Device Support

I2O stands for Intelligent Input/Output. It is a specification and protocol for offloading I/O processing to a dedicated I/O processor (IOP), freeing the CPU for more important tasks (like calculating pi to 3,000,000 places). If you have an I2O device, enable support for it in this section. Otherwise, disable I2O support.

Networking Support

The Networking support section enables you to configure a wide array of networking options for your system. If you intend to use any sort of network

device or feature, enable Networking support. Under Networking options, select the features and protocols you intend to use. For example, if you want to use programs like `tcpdump`, a packet sniffer that facilitates troubleshooting otherwise opaque network problems, click enable the Packet socket option. In the TCP/IP networking section, while you should definitely enable TCP/IP networking itself, you can usually disable all the listed options unless you need to use the features listed there. You should also enable UNIX domain sockets because this option provides Berkeley sockets support, which is used by many applications (such as the X Window system).

Other features that you can disable unless you need them include

- The various options between IP: ARP daemon support and IP: Virtual Server Configuration, but see the following discussion about TCP syn-cookie support
- The IPv6 protocol
- IPsec user configuration interface
- SCTP Configuration (SCTP is short for Stream Control Transmission Protocol)
- Asynchronous Transfer Mode (ATM)
- The IPX protocol
- Appletalk protocol support
- Frame Diverter
- WAN router
- QoS and/or fair queueing
- Network testing
- Netpoll support
- Netpoll traffic shaping
- IrDA (infrared) subsystem support
- Bluetooth subsystem support

If you connect this system to the Internet and want to use a packet filter (firewall), enable Network packet filtering, which enables you to use the Netfilter firewall feature (also known as *iptables*), which replaces *ipchains* (the 2.2 kernel packet filter). Netfilter also enables you to use your Fedora Core or RHEL system for IP masquerading. IP masquerading, also known as Network Address Translation (NAT), enables you to route all Internet traffic through a single system or IP address without requiring all systems to have valid Internet

addresses (IPs) and without making the systems on the internal network visible to the Internet.

You might want to enable the IP: TCP syncookie support option, which enables legitimate users to continue to use systems that are experiencing a SYN flood, a type of denial of service attack. If you do enable this option, you also need to enable support for the `sysctl` interface and the `/proc` file system (which you should do anyway) and add the following command to `/etc/rc.d/rc.local` or another initialization script:

```
echo 1 > /proc/sys/net/ipv4/tcp_syncookies
```

Additional discussion about netfilter configuration might prove helpful. If you intend to use IP masquerading, enable the Connection tracking and IP tables support options under Network packet filtering (replaces ipchains) ⇨ IP: Netfilter Configuration. Under the IP tables support submenu, select the iptables options that you need. If you are unsure which options to use, the following list is a reasonable starting point:

- Packet filtering ⇨ REJECT target support
- LOG target support
- Full NAT ⇨ MASQUERADE target support
- Full NAT ⇨ REDIRECT target support
- Packet mangling

If you have one or more network cards, enable them in the Network device support section. The Dummy net driver support option enables you to assign a configurable IP address to otherwise inactive Point-to-Point Protocol (PPP) addresses, which can make programs requiring an Internet connection work properly. If you have a constant connection to the Internet and don't need or use PPP support, however, you can disable dummy net driver support.

The configuration sections for specific classes of network devices, such as ARCnet, Ethernet (10 or 100 Mbit), Ethernet (1000 Mbit), Token Ring devices, PCMCIA network device support, and so on, enable you to configure support for such devices if you need it. Although you can probably skip or ignore the dialog boxes for devices you do not use, it does not hurt anything to look at each one and make sure that support is in fact disabled for unnecessary devices. Most of these sections begin with an option that enables/disables that category of device. Figure 27-9, for example, shows the wireless LAN configuration items. Notice that you can only see the subitems if the Wireless LAN drivers (non-hamradio) & Wireless Extensions option is enabled (as it is in Figure 27-9).

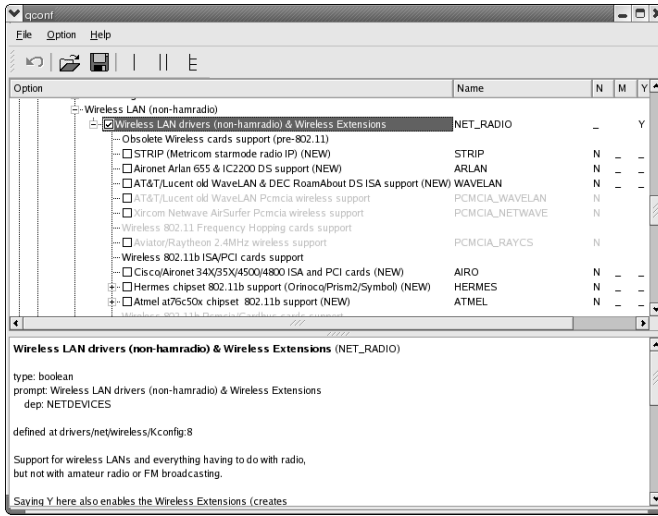


Figure 27-9 Viewing xconfig's Wireless LAN configuration section.

Each class of networking devices can be similarly disabled. Proceed sequentially through each class of devices. Most people will be able to disable support for the following device classes:

- ARCnet devices
- Ethernet (1000 Mbit)
- Ethernet (10000 Mbit)
- Token Ring devices
- Wireless LAN (non-hamradio)
- PCMCIA network device support
- WAN interfaces

If you need wireless support, however, you must enable Wireless LAN (non-hamradio) support and PCMCIA network device support.

Enable FDDI support if you need support for fiber-optic networking. PLIP (parallel port) support builds a module that enables you to create a network connection using the parallel port, but most people do not need this. If you are building a kernel on a system that connects to the Internet using a PPP connection, enable PPP (point-to-point protocol) support and enable the following options (as modules if you are building a modular kernel):

- PPP support for async serial ports
- PPP support for sync tty ports

- PPP Deflate compression
- PPP BSD-Compress compression

Disable SLIP (serial line) support unless you use SLIP instead of PPP (most people use PPP or PPP over Ethernet these days). Likewise, enable Fibre Channel driver support only if you need it.

ISDN and Telephony

If you use ISDN, you should enable ISDN support and select your ISDN options. ISDN is much more widely used in Europe than in the United States and Asia. Likewise, if you will use your Fedora Core or RHEL system with a telephone card or for voice-over-IP (VOIP), enable Linux telephony support. At the moment (that is, at the time this paragraph went to press), the only supported telephone devices are the QuickNet Internet LineJack and Phone-Jack cards.

Input Device Support

Proceed carefully in the Input device support section because if you disable the wrong device, you won't be able to use your keyboard. The default configuration helps you out in this regard because it includes keyboard support (and mouse support if the kernel build system detects a mouse), but you should still be careful. If you have a joystick (no one here plays games, right?), enable the Joystick interface option and in the Joysticks section a little farther down in the interface, select the joystick you have. Similarly, if you have one of the listed sound cards and want to be able to use the game port on it, enable the appropriate game port support. Otherwise, disable the Gameport support option entirely.

The Input Device Drivers section is the section in which you select the specific input devices (mice, joysticks, touch screens, and miscellaneous devices) you want to use. Under the Keyboards subsection, you will not see an option for standard PS/2-style keyboards because support for these kinds of keyboards is defaulted into the kernel, as described in the previous paragraph. On the other hand, if you're just dying to run Linux on your old Sun and want to use that Sun Type 5 keyboard on your Linux system, or you're trying to configure a kernel for Sun hardware, you probably want to enable the Sun Type 4 and Type 5 keyboard support.

In the Mice subsection, enable support for the type of mouse you have. Most people will enable the PS/2 mouse option. Enable support for your joystick, if you have one, as mentioned earlier. If you don't have a joystick, you can disable the Joysticks option entirely. The same goes for the Touchscreens option,

unless you are configuring a kernel for a kiosk system or cash register and need touch screen support. Finally, in the Misc subsection, you can configure support for a basic sound driver for the (awful-sounding) PC speaker. You can safely disable the User level driver support unless you have a user space program that needs to interact with the input subsystem.

Character Devices

Use the Character devices section to configure your kernel's support for your system's character devices, which includes serial ports, printers, tape drives, special-purpose chips and devices such as watchdog cards and clock chips, and a broad but shallow selection of frame buffer cards. You can also configure such devices that are provided by PCMCIA cards. The array of supported devices is mind-numbingly long.

If you have a special serial card, such as a multiport serial device, enable Non-standard serial port support and then select your device from the long list of support hardware and configure it appropriately.

If you need to use a serial port for Internet access (say, via a PPP dial-up connection) or you want to use your modem to send and receive facsimiles, make sure you enable 8250/16550 and compatible serial support. It isn't necessary to enable the Console on 8250/16550 and compatible serial support option unless you will be interacting with this system via serial connection (using minicom, Seyon, Kermit, or a similar terminal communications program). In the Extended 8250/16550 serial driver options subsection, you might want to enable two options: Support for sharing serial interrupts and Autodetect IRQ on standard ports. The latter option is qualified as unsafe, but it is very rare these days for IRQ autodetection to hang a machine. Of course, if you experience this problem, disable this option.

The default value (disabled) for Legacy (BSD) PTY support should be sufficient. If you intend to use a printer attached to this system, enable Parallel printer support. Support for special-purpose devices, such as the enhanced real-time clock or the hardware random number generator, is optional. Read the help text if you want to experiment with these features.

I2C Support

The primary reason to enable I2C (pronounced "eye-square-see") support is to use system and motherboard hardware sensors (lm_sensors) and Video4Linux devices (discussed in the next section, "Multimedia Devices"). Hardware sensors use a subset of the I2C protocol known as SMBus, or System Management Bus. To activate your hardware sensors, you need to enable I2C support and

then select the specific hardware sensors your system has under the Hardware Sensors Chip support submenu. For more information about configuring hardware sensors, see Chapter 32.

Multimedia Devices

The Multimedia devices section contains the kernel configuration items for Video For Linux, colloquially referred to as video4linux or v4l. Video4Linux comprises support for audio and video capture cards, digital video cameras (Web cams), and radio tuner cards. For more information about Video4Linux, see the project Web site at <http://linux.bytesec.org/v4l2>. You will also find configuration options for Digital Video Broadcasting (DVB) devices, which generally refers to any digital television broadcast or reception add-in card or peripheral. Strictly speaking, the DBV abbreviation refers to the DVB standard that defines the encoding characteristics of broadcast digital video streams. For more information about DVB, see the Linux TV Web site at linuxtv.org.

If you intend to use a Web cam, frame grabber, or radio tuner on your Fedora Core or RHEL system, enable the Video For Linux option and then select your adapter or device from the Video Adapters submenu or, for radio tuners, the Radio Adapters submenu. For DVB support, you must enable the top-level DVB For Linux option, DVB Core Support, and then select your adapter or device from one of the DVB submenus.

Graphics Support

`xconfig`'s Graphics support section permits you to configure support for certain motherboard and graphics chipsets and various kernel features related to graphics output, such as frame buffer support, the all-important boot logo, and limited support for LCD displays. Frame buffers enable the kernel to provide a generalized interface to a range of graphics hardware, allowing application software to manipulate the display without needing to know anything about the particular graphics device in use. One of the benefits of Linux frame buffer support is that it provides that *trés* chic Linux boot logo displayed on the screen in console (text) mode. More importantly, Linux's frame buffer support makes it possible to use the X Window system with graphics hardware that lacks X-specific drivers. In fact, on some non-X86 architectures, the frame buffer device is the only way to use the graphics hardware.

If you wish to use the frame buffer, enable the Support for frame buffer devices option and then select the specific graphics card you have. For example, if you have a Cirrus Logic-based graphics adapter, enable the Cirrus Logic support option. If your particular graphics adapter is not supported, enable VGA 16-color graphics support and the VESA VGA graphics support options.

TIP If you have a Matrox-based graphics adapter, be sure to select the **Enable Tile Blitting Support** option because the Matrox frame buffer driver, known as `matroxfb`, relies heavily on tile blitting for fast graphics performance.

For more attractive text mode displays, enable the VGA text console option and also enable Framebuffer Console support. Under this latter item, check **Select compiled-in fonts** and enable VGA 8x16 font to get clearer fonts when you work in text mode. Finally, for that boot logo, select **Logo configuration** ⇨ **Bootup logo** ⇨ **Standard 224-color Linux logo**.

Sound

Use the Sound section to configure support for your sound card, if any. Because the kernel now supports an amazingly wide array of sound cards, it simply is not possible to describe each option in this space. Locate the device or chipset for which you need support, enable it, and be sure to read and understand the help text and any supporting documentation to ensure that you can take full advantage of your sound card. If the list of drivers and cards under Advanced Linux Sound Architecture (ALSA) does not include a driver for your particular card, you might be able to use one of the Open Sound System (OSS) modules to run your sound card. If you need to use an OSS driver, be sure to disable the ALSA drivers or the OSS driver won't work.

USB Support

If you need support for Universal Serial Bus (USB), which almost anyone using a current X86-based PC does need, expand the USB support section and try not to despair when you see the list of features and options. To organize the discussion, each of the major configuration sections is covered separately, so you can go straight to the section that interests you.

Support for Host-Side USB

To use your USB devices, you need to enable the **Support for Host-side USB** option. A *USB host* refers to the host controller, which you can picture as the root of an inverted tree of USB devices. It is the USB host, or rather, the USB host port, that provides power to all attached devices. Continuing the tree analogy, if the USB host (or host port) is the root of the USB tree, attached USB devices, such as scanners or printers, are the end points, nodes, or leaves of the tree. Between the host and individual USB peripherals are special USB devices known as hubs, which might represent branches or branch points on the USB tree. In addition to enabling your host USB port, you must select one of the host controller drivers, described in the next section. You should also enable the

USB device file system option so that you can view the USB device tree in `/proc/bus/usb/xxx/yyy` (where `xxx` is the bus number and `yyy` the device ID).

USB Host Controller Drivers

Host controller drivers (abbreviated HCD) fall into three classes, OHCI or UHCI for USB version 1.1 and EHCI for USB 2.0. *OHCI* stands for Open Host Controller Interface. This class of host controllers is more widely used on non-X86 systems. *UHCI* stands for Universal Host Controller Interface. UHCI is almost universally (no pun intended) used on chipsets made by Intel and VIA; most people will enable UHCI HCD (most Intel and VIA) support. The third option, EHCI, stands for Enhanced Host Controller Interface. Whereas OHCI and UHCI support USB version 1.1, EHCI supports version 2.0. However, for backward compatibility, EHCI usually includes either OHCI or UHCI support. So, if you select EHCI, you might also need to select OHCI or UHCI to support USB devices that do not speak the version 1.1 protocol.

NOTE `xconfig`'s **USB Device Class Drivers** item is simply a heading under which the various classes of USB device drivers are arranged. It is not configurable.

USB Audio Support

If you want to connect USB audio devices to your system *and* you are not using the ALSA sound subsystem, enable USB Audio Support. ALSA provides its own USB audio driver, so you can disable this option if you use ALSA sound. If you have USB-attached MIDI devices, modems, or printers, enable the corresponding configuration options, USB MIDI support, USB Modem (CDC ACM) support, or USB Printer support, respectively.

USB Mass Storage Support

Enable USB Mass Storage support and select the appropriate device driver support if you want to use USB-connected devices such as CompactFlash or smart card readers. For CompactFlash, you will also need IDE support, as remarked on earlier in the chapter. A potential “gotcha” in this section is that USB mass storage devices usually require SCSI disk support, so you might need to go back to the SCSI configuration section and enable SCSI device support ⇔ SCSI disk support. One of the nice features of mass storage support is that it plays nicely with the udev system: plug in a supported device and udev will detect the hotplug event and automatically create the appropriate device node and file system entries so that you can access the storage.

You should to be aware of one subtlety with USB mass storage support. When you enable mass storage support and select the corresponding device, support for that device is compiled into the mass storage module. For instance, if you want to support your Zio! compact flash reader, you would enable USB Mass Storage support and enable Microtech/ZiO! CompactFlash/SmartMedia support. The resulting module, `usb-storage`, would contain general mass storage support code *and* code to support the Zio! reader. This is a departure from normal practice, in which feature support and support for a specific device are provided by separate modules. In terms of the example, normal practice might manifest in general mass storage support residing in the `usb-storage` module and Zio!-specific support being provided by a module named, say, `zio`.

USB Human Interface Devices

The USB protocol has complete support for a broad range of so-called human interface devices (HID), which refers to any device you can use to provide input, including mice, keyboards, graphics tablets, and joysticks. If you want to be able to interact with your computer using any tools in this class of USB devices, enable USB Human Interface Device (full HID) support and then select HID Input layer support plus a driver for your particular device. Notice, though, that USB HID and USB HIDBP (HID boot protocol) drivers don't play nicely together. If you intend to use (or must use) HIDBP support (USB HID Boot Protocol drivers) for your keyboard or mouse, you cannot also use the full HID driver. As a rule, you should opt for full HID support and omit the HIDBP drivers. Naturally, if your mouse or keyboard uses the PS/2 protocol, you don't need USB HID or USB HIPBP support.

USB Imaging Devices

At the moment, support for USB imaging devices is fairly limited in the kernel itself. You can enable only a single digital camera (the Mustek MC800) and a couple of scanners, the Microtek X6USB and the HP53xx series of scanners. However, the Scanner Access Now Easy (SANE) project provides a much deeper level of support for all manner of scanners, including USB-attached scanners, using the USB file system (`usbfs`), described shortly in the "File Systems" section.

USB Multimedia Devices

USB multimedia devices include a variety of Web cameras and the D-Link RM radio. If you need support for one of the listed devices, enable the appropriate driver.

USB Network Adapters

USB network adapters are becoming increasingly common and popular. Accordingly, the kernel's support for these adapters continues to grow. If you have one of the USB-based adapters listed under USB Network Adapters, enable support for it.

USB Serial Converter Support

USB is a high-speed serial port protocol. Thus, it should be no surprise that `xconfig`'s USB Serial Converter support section includes several dozen options. In most cases, to support any given device, you must enable the driver for that particular device and the USB Generic Serial Driver option at the top of the list. For example, if you have a Palm PDA, you would enable the following options:

- USB Serial Converter Support
- USB Generic Serial Driver
- USB Handspring Visor/Palm m50x/Sony Clie Driver

USB Miscellaneous Drivers

The miscellaneous drivers category is aptly named because it lists USB-attached odds and ends that neither fit anywhere else nor (yet) have their own configuration subsections. Enable support for the device or devices you have and intend to use.

USB Gadget Drivers

Although 99.999 percent of you won't need this option, it bears mentioning because it shows how ubiquitous Linux has become. So far, all USB devices discussed have been attached to a system running Linux. The USB Gadget Drivers section is for USB devices that are themselves running Linux. Such devices, known as *USB gadgets*, might or might not be connected to a host system running Linux. When the gadget is connected to a Linux system, it functions as a peripheral and communication is via a USB peripheral controller, which is not the same as USB host controller. The canonical example of a USB gadget device is a PDA that runs Linux (such as the Sharp Zaurus). In some cases, the USB gadget needs a controller to communicate with the host system. If you have such a gadget, enable the appropriate controller in the USB Peripheral Controller section. Of course, you might also need to enable the appropriate gadget.

MMC/SD Card Support

MMC is the bus protocol for so-called multimedia cards. If you have an MMC or SD card (rather, MMC or SD card *reader*), enable the MMC support option and then select the driver for the MMC interface you need to support. Unless you are working on the kernel's MMC support, though, disable the MMC debugging option.

InfiniBand Support

If you need InfiniBand support, enable it in this section. InfiniBand is another entry into the already crowded field of marketing buzzwords for various high-speed bus and data interconnect protocols. Here's the definition from the InfiniBand trade association:

InfiniBand is an interconnect or I/O architecture that connects servers with remote storage and networking devices, and other servers. It can also be used inside servers for inter-processor communication. InfiniBand is a channel-based, switched fabric, point-to-point interconnect, which provides scalability and performance for a wide range of platforms and price performance points. InfiniBand provides a scalable performance range of 500 MB/s to 6 GB/s per link, meeting the needs from entry level to high-end enterprise system (storagesearch.com/glossary.html).

In short, InfiniBand is a supercalifragilisticexpialidocious data interconnect. For more information about InfiniBand, and to decipher the impenetrable network-speak in the preceding quote, visit the InfiniBand Trade Association on the Web at infinibandta.org/home/.

File Systems

The File systems configuration section enables you to configure support for the file systems you expect to use or access from your system. One of Linux's greatest strengths is the breadth of its support for non-native file systems, which makes it possible for Linux to interoperate with almost any other operating system used on more than five computers.

TIP Build support for the file system on which your / file system resides into the kernel. Although this is not strictly necessary because you can use an `initrd` loaded at boot time, building the support into the kernel is simpler, less error prone, and marginally faster. Static support for your root file system is simpler and less error prone because you don't need to remember to create an `initrd` each time you build a new kernel. It is faster because the kernel does not have to uncompress the `initrd` and load the module at boot time.

At a minimum, enable support for `ext2` and the `ext3` journaling file system. If you wish, or need to, enable support for one of the other journaling file systems currently available:

- **ReiserFS support** — Provides support for the ReiserFS file system
- **FS** — Provides support for IBM's Journaling File System
- **XFS** — Provides support for SGI's high-performance Extended File System

Unless you know you need them, you can safely disable support for the Minix and ROM file systems. If you intend to use file system quotas, enable Quota support and refer to Chapter 29, for information about using disk quotas. If you want to use the kernel's automounter support as described in Chapter 12, enable Kernel automounter version 4 support (the automounter enables you want to mount and unmount file systems on the fly). As a general rule, build non-Linux file system support as modules to minimize the size of the kernel.

CD-ROM/DVD File Systems

You'll need to enable support for CD-ROM and DVD file systems if you intend to use a CD-ROM or DVD on your computer. As a starting point, enable the following options:

- **ISO 9660 CD-ROM file system support** — Enables kernel support for standard CD-ROM file system
- **Microsoft Joliet CDROM extensions** — Enables the kernel to support Microsoft Windows' long filename extensions for CD-ROMs in Unicode format

Some newer CD-ROMs and DVDs use the UDF file system. If you expect to use such disks, enable the UDF file system support option.

DOS/FAT/NT File Systems

For better or worse, most Linux users and system administrators need to provide access to various DOS or, more likely, Windows file systems. To do so, enable all three options, as modules, under DOS/FAT/NT File systems. You'll be happy to know that incomplete but safe write support is available for NTFS file systems.

Pseudo-File-Systems

Pseudo-file-systems are so named because they are not true file systems with a distinct file system structure laid down on a physical disk. Rather, pseudo-file-systems present a view into kernel data structures using a file system metaphor and standard system calls for file system operations, such as opening, closing, reading, and writing. An increasing number of such pseudo-file-systems exist, as you can see under the Pseudo file systems section. You can probably get by enabling only `/proc` file system support, which adds support for the `/proc` file system. `/proc` enables applications to utilize the `proc` interface to kernel tuning and device access.

Miscellaneous File Systems

The Miscellaneous file systems support section reveals the extent of Linux's support for non-native file systems. If you need to access disks or data that resides on one of the listed file systems, enable support for that file system in this section. Again, to keep the size of your kernel as small as possible, enable support for foreign file systems as loadable modules.

Network File Systems

As an operating system that was designed and built from the ground up in a networked environment, Linux boasts rich support for networked file systems, as you can see the Network File Systems section. We recommend enabling support for NFS (Network File System) version 3, which has considerable improvements and better kernel support for NFS than version 2 does, but is not as unproven as NFS version 4 is. So, enable Provide NFSv3 client support and disable Provide NFSv4 client support under NFS file system support. Similarly, enable Provide NFSv3 server support and disable Provide NFSv4 server support under the NFS server support subsection.

If you want to use Samba to access Windows file systems, enable SMB file system support (to mount Windows shares, and so forth). Similarly, enable NCP file system support if you need to access Netware file systems, and then select the specific options for Netware file systems as fits your situation.

Partition Types

In addition to support for foreign (non-Linux) file systems, you might find it advantageous to configure the kernel to provide support for foreign disk partition types. To do so, enable Advanced partition selection in the Partition Types section and then select the specific partition types you want to use.

Native Language Support

The final subsection concerned with file systems is Native Language Support. Fedora Core and RHEL are configured to use the UTF8 character set encoding, so you should at least enable that character set under Base native language support. You should try to select support for both the codepage you need and the ISO NLS character set. For example, if you need support for Greek, enable the Codepage 869 (Greek) option and the NLS ISO 8859-7 (Modern Greek). American users should select the following:

- Codepage 437 (United States, Canada)
- ASCII (United States)
- NLS UTF8

You can build support for other character sets as modules and loaded at run time if you sometimes need support for languages and character sets other than the default.

Profiling Support

Profiling support, if enabled, activates the kernel's support for the hardware performance counters built into modern CPUs and chipsets. By itself, this option does nothing unless you also enable Oprofile system profiling (EXPERIMENTAL) *as a module* in order to create data that you later turn into information. You should not build OProfile into the kernel; there is little need outside of academic research for a kernel with a static profiler.

What is OProfile? OProfile is a systemwide profiler that gathers performance metrics on all running code at low overhead. All code is profiled: hardware and software interrupt handlers, kernel modules, the kernel, shared libraries, and applications. OProfile consists of a kernel driver, a daemon that collects data samples, and a collection of tools to turn raw OProfile data into more useful information. It works by using the hardware performance registers built into modern CPUs. Although OProfile is currently considered to be in alpha status, it has been used successfully and reliably in a wide array of hardware and computing environments.

Depending on your needs, you can use OProfile to produce performance data at the function level or down to the individual CPU instruction. You can use OProfile data to annotate source code with profile information. You can also use OProfile to identify the top *N* applications or functions that take the most time across the whole system.

Clearly, this is not the sort of tool you would use on a daily basis. But if you are trying to squeeze the last possible CPU cycle out of an application, OProfile is a terrific tool to have. Build it as a module and then forget about it until you need it. You'll be glad you did. Refer to Chapter 32, for a quick-and-dirty OProfile usage primer.

Kernel Hacking

The collection of features in the Kernel hacking section are not meant for mortal users. Our recommendation is to disable all kernel-hacking features unless you are specifically asked to enable them. These options and features make debugging the kernel easier and provide information that kernel developers can interpret or understand, but they have little value to anyone else. In fact, some of the kernel-hacking options actively attempt to break the kernel code, which might result in a crashed or otherwise unstable system.

Security Options

The Security options section is another section best left alone unless you fully understand the consequences and usage of the security feature or tool in question. Accordingly, the kernel features in this section will be discussed in more detail in Chapter 33 and briefly in Chapter 34. If you enabled SELinux support when you installed Fedora Core or RHEL, enable support for SELinux by enabling the following options:

- Enable different security modules
- Socket and Networking Security Hooks
- NSA SELinux Support. Under this option, you should also enable:
 - NSA SELinux boot parameter
 - NSA SELinux runtime disable
 - NSA SELinux AVC Statistics

Refer to Chapter 33 for more information about SELinux.

Cryptography Options

The Cryptographic options section enables you to select which of a multitude of cryptographic APIs (ciphers) that you want to support in the kernel. The reason to include the cipher algorithms in the kernel is that kernel mode code executes faster and is more secure than code that executes in user mode. Kernel mode code is faster because it is highly optimized and less prone to interruption or preemption by other code. It is more secure because access to kernel data is restricted to kernel code.

As you can see in Figure 27-10, there are quite a few cipher choices from which to choose. Most are compiled as loadable modules, but the SHA1 digest (SHA1 digest algorithm) is compiled statically into the kernel.

By making most of the cryptography APIs available as modules, programs that need a particular cipher can load the support and access without having to bulk up the kernel with code and data that is only rarely needed. How much bulk? In the standard Fedora Core and RHEL kernels, the total code and data taken up by the loadable cryptography modules is 193,005 bytes.

If you have cryptographic hardware, then you will likely want to enable kernel support for it, if possible, in the Hardware crypto devices section. If you do have cryptographic acceleration hardware, you know who you are (and *They* know who you are, too!). Otherwise, proceed to the Library routines section of `xconfig`.

Library Routines

The configuration section for Library routines is a common source of confusion. The three options in this section exist for the purpose of supporting kernel modules built outside of the kernel tree that require cyclical redundancy check (CRC) support routines. The phrase “kernel modules built outside of the kernel tree” refers to device drivers and other loadable kernel modules provided by third parties, such as peripheral manufacturers, that are not part of the official kernel source code tree. A CRC is a method of performing error checking and data validation. Ordinarily, device drivers in the official kernel that need CRC routines implement them themselves or use one provided in the kernel sources. However, some third-party loadable kernel modules do not implement their own CRC support and must rely on the kernel’s CRC functions. Therefore, the configuration options in this section implement the necessary support as modules so that third-party drivers can access them. You will not usually need to enable these options, but if want to load a third-party module that indicates your kernel needs in-kernel CRC support, this is the section you’ll need to use.

Saving the Kernel Configuration

After you have worked your way through all of the configuration options, click the Save button on the toolbar or select File ⇨ Save to save your configuration choices to the file `.config` in the kernel source directory. Then select File ⇨ Exit to close `xconfig` and, at *great* length complete the configuration process. You can also just select File ⇨ Exit without specifically saving your changes because `xconfig` will prompt you to save any unsaved changes. (See Figure 27-11.)

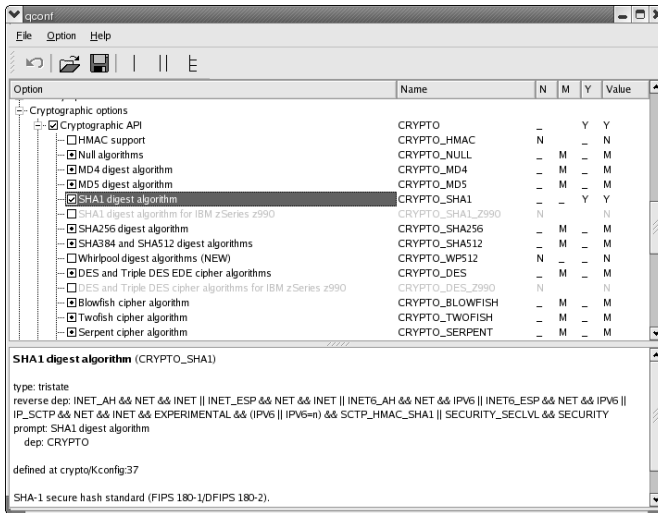


Figure 27-10 Viewing the available kernel cryptography APIs in xconfig.

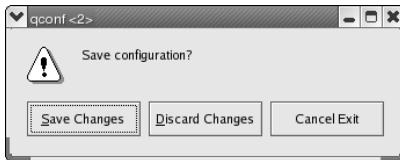


Figure 27-11 xconfig's Save dialog box.

The good news is that future configuration sessions will proceed much more quickly because you only need to perform a complete configuration once. Subsequent sessions are usually a matter of changing only one or two options and then rebuilding the kernel.

Compiling the Kernel

To build the kernel and any loadable module you want, type **make** in the top-level kernel directory and take a coffee break. On an AMD Athlon Thunderbird 1200 with 512 MB of RAM, the compilation process took just over 50 minutes. Your mileage may vary.

```
$ make
...
OBJCOPY arch/i386/boot/vmlinux.bin
HOSTCC arch/i386/boot/tools/build
BUILD arch/i386/boot/bzImage
Root device is (3, 3)
```

```

Boot sector 512 bytes.
Setup is 5487 bytes.
System is 11237 kB
Kernel: arch/i386/boot/bzImage is ready

```

The new kernel build system (known as `kbuild`, oddly enough) combines the `make bzImage` and `make modules` steps familiar to from earlier kernel version into a single step in the 2.6 build system. You will also like notice that the output of the build process is much neater and easier to read in the new system because `kbuild` disables the hard-to-read verbose output with a single line showing each step in the build process (as illustrated in the last few lines of output in the previous code). If everything has proceeded as designed, you should have a compressed kernel image, `arch/i386/boot/bzImage`, and a set of modules (if you built a modular kernel) that are ready to install.

WHAT'S THIS KERNEL SIZE BUSINESS, ANYWAY?

This chapter repeatedly comments that removing unneeded drivers from the kernel and compiling drivers as modules will make it smaller or that, conversely, adding drivers increases the kernel's size. What exactly does this mean and why precisely should you care?

Each block of code, each driver, that goes into the kernel increases the size of the compiled kernel in terms of disk footprint and in terms of the amount of memory it takes when it's running. It is the kernel's runtime memory footprint that matters most. Even though modern computers can have multiple gigabytes of RAM, the amount the kernel consumes should be kept as small as possible to maximize the RAM resources available to other programs and to keep the kernel itself as efficient as possible. Moreover, from a purely mechanical standpoint, the smaller your kernel, that is, the fewer device drivers you configure, whether statically compiled or built as modules, the less time it takes to build and rebuild the kernel.

You don't have to accept this at face value, however. You can judge for yourself by using the `size` command. `size` shows the size in bytes of each section in a binary or object file. The text section contains executable code; the data section contains static data declared in the program; the bss section contains uninitialized global data. The dec field shows the total file size in decimal bytes and the hex field shows the total file size in hexadecimal bytes.

This is more than a merely academic issue. Veteran Linux users remember when a full-featured kernel image could be created on a 3.5" floppy disk using the `make target` `make floppy`. The kernel has become so large that it won't fit on floppy anymore. In addition, the target used to create compressed kernel images, `make bzImage`, used to be `make zImage`. The compression algorithm had to be changed because the old algorithm couldn't compress the image efficiently enough.

(continued)

WHAT'S THIS KERNEL SIZE BUSINESS, ANYWAY? (continued)

Consider the following. The first command uses `size` on a “typical” kernel (2.6.7.10 built from the Red Hat kernel SRPM):

```
$ size vmlinux-typical
      text      data      bss      dec      hex filename
2123796  377916  137916 2639628  28470c vmlinux
```

The next command shows the output of the `size` command used on the uncompressed kernel image (`vmlinux`) built using the `make allnoconfig` configuration described at the beginning of this chapter:

```
$ size vmlinux-allnoconfig
      text      data      bss      dec      hex filename
719553  135731   61028  916312  dfb58 vmlinux
```

This is our idea of a small kernel, just over 700 KB. For embedded systems, such as cell phones or set-top boxes, even 700 KB is much too large!

By way of contrast, here's the output of the `size` command used on the uncompressed kernel image built from the configuration created by the `make allyesconfig` kernel target:

```
$ size vmlinux-allyesconfig
      text      data      bss      dec      hex filename
18101419          6286167 1808916 26196502      18fba16 vmlinux
```

Yes, that's correct, the code space required for the kernel image with all possible options compiled in is approximately 18 MB and the total size is over 25 MB!

Next, `size`'s output for the default configuration created by the `make defconfig` target:

```
$ size vmlinux-defconfig
      text      data      bss      dec      hex filename
3260358  716068   262020 4238446  40ac6e vmlinux
```

Finally, the size information of a kernel built with all options possible compiled as loadable kernel modules (using `make allmodconfig`):

```
$ size vmlinux-allmodconfig
      text      data      bss      dec      hex filename
1945534  909810   242276 3097620  2f4414 vmlinux
```

As you can see, the different default kernel configurations result in kernels that consume varying amounts of disk space. You might find it a revealing exercise to see how small you can make your kernel image while retaining all of the necessary functionality.

Installing the Kernel

Happily, installing the new kernel takes considerably less effort and time than configuring and building it do. You just install the modules, copy the compressed kernel image into place, and create an initrd image. These steps require root access.

1. Install the kernel and modules:

```
# make install
...
if [ -r System.map ]; then /sbin/depmod -ae -F System.map 2.6.11.7; fi
```

make install copies the compressed kernel image into /boot so GRUB can find it and then installs the modules in /lib/modules.

2. Alternatively, if you don't trust make install, you can install the modules using make install_modules and then manually copy the compressed kernel image into /boot so that GRUB can find it:

```
# make modules_install
...
if [ -r System.map ]; then /sbin/depmod -ae -F System.map 2.6.11.7; fi
# cp arch/i386/boot/bzImage /boot/vmlinuz-2.6.11.7-custom
```

Notice that name assigned to the copied kernel image, vmlinuz-2.6.11.7-custom. It is prudent to use a name that distinguishes your custom kernel image from other kernel images already in /boot. You can call it anything you want (fred is fine) as long as it means something to you.

3. Create an initrd image for the kernel to load:

```
# mkinitrd -v /boot/initrd-2.6.11.7-custom.img 2.6.11.7
Creating initramfs
Looking for deps of module ide-disk
Looking for deps of module xfs
Using modules:
/sbin/nash -> /tmp/initrd.W28511/bin/nash
/sbin/insmod.static -> /tmp/initrd.W28511/bin/insmod
/sbin/udev.static -> /tmp/initrd.W28511/sbin/udev
/etc/udev/udev.conf -> /tmp/initrd.W28511/etc/udev/udev.conf
```

This command creates the initrd as /boot/initrd-2.6.11.7-custom from the kernel modules for version 2.6.11.7. The -v option causes slightly verbose output so you can see what mkinitrd is doing.

The `initrd` image contains drivers that the kernel needs to boot and that are neither compiled statically into the kernel nor otherwise accessible at boot time. The classic example of this is drivers for the disk on which the root file system is stored. If the kernel doesn't contain a built-in driver for the disk that contains the root file system, it can't mount the root file system; if the kernel can't mount the root file system, it can't load the driver. Storing the driver in an `initrd` that the kernel loads at boot time avoids this catch-22.

After you've installed the kernel, modules, and created an `initrd`, you can update the boot loader configuration file.

Updating GRUB

To make GRUB aware of the new kernel, open `/boot/grub/grub.conf` in the text editor of your choice and add a stanza at the bottom that resembles the following:

```
title Kurt's Kustom Kernel (2.6.11.7)
    root (hd0,0)
    kernel /vmlinuz-2.6.11.7-custom ro root=/dev/hda3
    initrd /initrd-2.6.11.7.img
```

The `title` line is the name that appears in GRUB's menu when your system first boots. The `root` line tells GRUB the device (`hd0`) and partition (`0`) to mount. The `kernel` line tells GRUB where to find the kernel image and the command line parameters to pass to the kernel as it boots (`ro root=/dev/hda3`). The `initrd` line, finally, tells GRUB where to find the `initrd` image. Because the system used for this example has a separate `/boot` partition, the paths to the kernel and `initrd` image are defined relative to the `/boot` partition. If you are unsure how to modify the example stanza for your system, the easiest thing to do is copy an entry for one of the existing kernels and modify it with values appropriate for your system.

To configure GRUB to boot the new kernel by default, change the value of the `default` directive to correspond to number of the `title` section that contains the new kernel. The count starts with 0. For example, if your new kernel is the fifth `title` section, change the line that reads `default=0` to `default=4` (remember, the count is zero-based). However, *before* you make the new kernel the default kernel, you should test it by rebooting your computer, loading the new kernel, watching the messages to ensure that your hardware has been detected properly, and using it for a few standard tasks, just to make sure that everything is copasetic.

Summary

One of Linux's most persuasive selling points is the ability it gives you to fine-tune the kernel for your specific needs. Often, it isn't necessary or advisable to build your own kernel, especially if you have a support agreement for RHEL. In such a situation, you're better off using the officially sanctioned kernels created by Red Hat Software. Fedora Core users enjoy more leeway because Fedora Core is neither affiliated with Red Hat (formally) nor supported by Red Hat. Consequently, if you're a Fedora user, you can use binary RPMs provided periodically by the Fedora Project or you can build your own custom kernel. The challenge when creating the do-it-yourself kernel is to understand all of the possible configuration options available to you or at least to know which of the options you *don't* need. Whether you use a prebuilt binary kernel or make your own, make sure that you have a boot disk, perhaps even two, so you can still get into your system if the upgrade process fails.

Configuring the System at the Command Line

IN THIS CHAPTER

- Administering Your System from the Command Line
- Managing Processes
- Maintaining the File System
- Timekeeping
- Automating Scripts

This chapter shows you how to use command line programs to perform system and network administration tasks. Using GUI tools for system administration is convenient when such tools are available, but they are not always available and often fail to offer a complete interface to subsystem administration. In fact, many GUI administration tools are little more than graphical wrappers around the command line programs you learn to use in this chapter.

Administering Your System from the Command Line

Why should you learn how to administer your Fedora Core or RHEL system from the command line? First, although the range of available GUI system and network administration tools continues to grow in both number and sophistication, not all administrative tasks have a matching GUI tool, and the available tools are often incomplete, out of date, or as unintuitive and awkward to use as the command line programs they attempt to replace.

Second, GUI tools are not always available. Red Hat systems installed strictly as server machines, rather than desktop or end-user workstations, rarely run the X Window system, much less have it installed. They are not only

headless, that is, lacking an attached monitor, but they also usually lack an attached keyboard or mouse. Running the X Window system on a server uses CPU cycles, RAM, disk space, process table entries, and file descriptors, better used to provide another service. Moreover, underpowered systems may lack the horsepower to run the X Window system, and thus GUI tools.

Another shortcoming of graphical tools is, ironically, the very user-friendliness that makes them so popular. That is, graphical administration tools require user interaction; they do not readily, if at all, lend themselves to automated, unattended execution. They are rarely scriptable or support the capability for making bulk changes, such as adding 25 user accounts in a single operation. In contrast, you rarely find a command line tool that cannot be scripted to facilitate bulk system changes or that cannot be automated using cron or another scheduling utility.

NOTE To be fair, in cases in which GUI tools are available, they can usually be used via X tunneling in `ssh` (`ssh's -X` and `-Y` options), which makes it possible to use GUI tools to administer a remote system.

The benefits of automating and simplifying system administration cannot be overestimated. In a small environment with few users, workstations, and servers, the inability to script GUI configuration and administration tools may not be an issue. However, system maintenance becomes a challenge even at the top end of the small office/home office (SOHO) sector, say 25 users and workstations, two or three servers, and half a dozen printers. At large installations with multiple servers, dozens or hundreds of users and workstations, plus all of the associated administrative overhead, administrators, and there are usually several, must be able to make bulk changes and perform system maintenance and administration unattended and automatically. To do otherwise is inefficient and impractical.

Fedora Core and RHEL are far from the only Linux systems you'll encounter. Graphical tools with which you are familiar on Fedora Core or RHEL usually don't exist on, say, a Debian or Novell system. Speaking from personal experience, the day will come when you need to administer a non-Red Hat-derived system. If you know how to use command line tools and understand the underlying configuration files, you won't be dependent on the GUI tool and, more importantly, your system administration skills will be considerably more portable. In some cases, thankfully less common, GUI tools interfere with system administration tasks because, for example, the modifications they make to a configuration file overwrite manual changes.

Finally, graphical administration tools, by their very nature, encourage uninformed or ill-informed system administration habits. GUI tools deliberately hide complex, often unintuitive configuration files and commands behind an attractive, convenient interface. Hiding the complexity does not

make it disappear, though. In fact, it may make it worse because you never learn how a service really works, how to troubleshoot a misbehaving daemon, or how to customize a program's behavior using command line options the GUI tool does not support.

Should you use graphical system administration tools? Absolutely. They are helpful, convenient, and timesaving additions to every system administrator's toolbox. Not every problem is a nail, so you need more tools than a hammer. GUI tools are only one set of many tools at your disposal. You do yourself, your employer, your colleagues, and your users a valuable service if you take the time to understand the details of system administration. Think of it this way: if the graphical utility for configuring a mission-critical service stops working for some reason, do you really want to tell your boss you do not know how to fix it? You need to know how to use the perfectly serviceable command line utility.

TIP Many of the commands and utilities discussed in this chapter reside in `/sbin` and `/usr/sbin`, which are not ordinarily in a mortal user's path.

Execute the following command to add them to your path:

```
$ export PATH=$PATH:/sbin:/usr/sbin
```

If you use `su -` to become the root user or login as root directly, `/sbin` and `/usr/sbin` will already be part of the path.

The following sections discuss process management, file system administration, monitoring and controlling system performance, configuring the system logs, keeping the system's date and time accurate, and writing and using scripts to perform maintenance tasks. Because many commands can be used by both normal users and the root user, this discussion focuses on usage and options pertinent to administrative needs.

Managing Processes

Administering processes includes identifying, monitoring, controlling, modifying, and obtaining a variety of information about them. The `ps`, `top`, and `kill` commands are the most familiar commands used for working with processes, but there are others that are more focused and, especially in the case of the `ps` command, probably easier to use. This section looks at three categories of commands:

- Commands used to obtain process information
- Commands used to send signals, usually the kill signal (`SIGKILL`), to processes
- Commands used to modify running processes

Obtaining Process Information

Process information is easy to obtain, if you know how to get it. The commands discussed in this section include the following:

- **ps** — Displays process status
- **pgrep** — Lists the PIDs of processes matching a given pattern
- **pidof** — Displays the PID of the process running a specified program
- **pstree** — Displays processes in a hierarchical format
- **top** — Displays process information and status on an ongoing basis
- **tload** — Displays a text mode load average graph

Tables 28-1 through 28-4 borrow the layout of `ps`'s syntax description from its manual page and organize each group of options into tables based on the options' purpose. However, the following tables omit all GNU long options (those preceded with `--`) and options related to defining custom output formats. `ps` supports both Unix98 options, which are preceded by a hyphen (`-`), and BSD options, which lack the initial `-`. Where the functionality is identical or very similar, we omit the BSD options. In some cases, apparently identical Unix98 and BSD are listed because the BSD option shows different output from the similarly invoked Unix98 option.

Table 28-1 Basic Process Selection

OPTION	DESCRIPTION
-N	Negates the selection criteria specified with other options
-a	Selects all processes with a TTY except session leaders
-d	Selects all except session leaders
-e	Selects all processes
T	Selects all processes on the invoking terminal
r	Selects only running processes
x	Selects processes without controlling TTYS

Table 28-2 Process Selection by Category

OPTION	DESCRIPTION
-C command	Selects by the command name matching pattern <code>command</code>
-G rgid name	Selects by RGID (Real Group ID) <code>rgid</code> or group name

Table 28-2 (continued)

OPTION	DESCRIPTION
-U ruid name	Selects by RUID (Real User ID) <i>ruid</i> or user <i>name</i>
-p pid	Selects by PID <i>pid</i>
-u euid name	Selects by EUID (Effective User ID) <i>euid</i> or user <i>name</i>
p pid	Selects by PID <i>pid</i> and displays the command line
U name	Selects processes for user <i>name</i> and displays the command line

Table 28-3 Standard Output Formats

OPTION	DESCRIPTION
-f	Displays full listing
-j	Displays output in jobs format
-l	Displays output in long format
j	Displays output in job control format
l	Display long output format
s	Displays output in signal format
v	Displays output in virtual memory format

Table 28-4 Modifying Output Format

OPTION	DESCRIPTION
-H	Shows process hierarchy (forest)
-w	Displays output in wide format
C	Uses raw CPU time for %CPU instead of decaying average
S	Includes some dead child process data (as a sum with the parent)
c	Shows the true command name
e	Shows environment after the command
f	Displays process hierarchy as ASCII art (forest)
h	Suppresses header lines (repeats header lines in BSD personality)
w	Displays output in wide format

If `ps`'s plethora of options and arguments seems daunting, the `pgrep` command provides relief because it provides a simpler interface, enabling you to select processes using simple pattern matching. It lists the Process IDs (PIDs) of processes matching the specified pattern. You can then use those PIDs with `ps`'s `p` or `-p` options to obtain more information, if you wish. `pgrep`'s syntax is:

```
pgrep [-flnvx] [-P ppid] [-u euid] [-U uid] [-G gid] [pattern]
```

If you specify multiple criteria, the selected processes match all criteria, so keep the selection criteria as simple as possible. `pattern` contains the expression to match against process names or command lines. `-f` matches `pattern` against the process name (the default if `-f` is not specified) or the full command line. `-l` causes `pgrep` to list both the PID and the process name. If multiple processes match `pattern` and other criteria, `-n` limits the output to the most recently started process. `-v` reverses the matching, showing all processes not matching the specified criteria. `-x` forces an exact match of `pattern`. `-P ppid` restricts the output to matches with a parent PID of `ppid`. Similarly, `-u euid`, `-U uid`, and `-G gid`, limit the output to processes whose EUIDs, UIDs, and/or GIDs, respectively, match `euid`, `uid`, and/or `gid`. Multiple `ppids`, `euids`, `uids`, and `gids` may be specified by separating each with commas.

For example, the following `pgrep` command shows all process that match `kio`, using the `-l` option to show the PID and the process name:

```
$ pgrep -l kio
19803 kio_uiserver
15225 kio_file
23092 kio_media
23094 kio_pop3
```

The next command uses the `-v` option to show all processes not owned by `root` or the user `bubba`:

```
$ pgrep -vlu root,bubba
2113 portmap
2131 rpc.statd
2530 mDNSResponder
2641 ntpd
2712 iiimd
2722 cannaserver
2768 xfs
2798 dbus-daemon
18160 pop3-login
18161 pop3-login
18386 imap-login
18746 imap-login
2165 httpd
```

```
2166 httpd
2167 httpd
...
```

The listing is truncated to preserve space. You can combine the options behind a single hyphen, as shown in the example, or specify each individually, that is, `pgrep -v -l -U root, bubba`.

`pidof` enables you to locate the PID of a process by name. Its syntax is:

```
pidof [-s] [-x] [-o pid] program
```

`program` is the base name of the program whose PID(s) you want to find. You can specify multiple programs by separating their names with white space. `-s` returns only the first PID located, additional PIDs are ignored. `-x` causes `pidof` to return the PID(s) of shell scripts running `program`. Specifying `-o pid` tells `pidof` to ignore (not to list) one or more PIDs that match `pid`.

`pstree` displays all running processes as a tree, making clear the parent/child relationships between processes. Its syntax is:

```
pstree [-a] [-c] [-H pid] [-n] [-p] [-G] [basepid | baseuser]
```

Called with no arguments, `pstree` shows all processes, with `init` as the root. Specifying `basepid` or `baseuser` begins the display from the PID specified by the PID `basepid` or the username `baseuser`, respectively. `-a` includes the command line for each process. `-c` expands identically named child processes (such as the `mingetty` instance spawned for each terminal). `-H pid` highlights the PID `pid` and its ancestor (parent) processes. If `pid` does not exist, `pstree` exits with an error. `-n` sorts the output by PID (the default is by ancestry). `-p` causes each process's PID to display and implies `-c`. `-G`, finally, draws the tree using the VT100 drawing characters rather than the default ASCII characters `|`, `+`, `-`, and ```.

`top` displays real-time CPU and memory usage and current system uptime information. Although it is an interactive command, `top` is a vital tool in every system administrator's toolbox, so its command line interface (not its interactive use) is covered here.

```
top [-bcisqS] [-d delay] [-p pid] [-n iter]
```

`-d delay` specifies the number of seconds between screen updates (default is 5), and `-q` specifies constant updates, which run at the highest priority if used by the root user. `-p pid` identifies up to 20 PIDs in white-space-delimited `pids` to monitor. `top` continues to run unless `-n iter` is specified, `iter` defining the number of iterations `top` refreshes its display before exiting (0 is interpreted as 1). `-S` enables cumulative mode, in which each process's CPU

time is shown as a sum and includes the CPU time of any dead child processes. Of course, for a child process's time to count, it must have died. `-s` runs `top` in secure mode and disables potentially dangerous commands in the interactive interface, such as `k`, which can kill processes. `-i` instructs `top` to omit idle or zombie processes from its display. `-b`, finally, runs `top` in batch mode and runs until specifically killed or until the number of iterations specified with `-n` have elapsed (all other command line input ignored). To exit `top`, press `q`.

`tlload` displays a real-time text mode graph of the system's load average, which is the number of processes waiting to run during the last minute, the last 5 minutes, and the last 15 minutes. Its syntax is:

```
tlload [-s scale] [ -d delay ] [tty]
```

`tlload` displays its graph to the current terminal, unless `tty` is specified, when it then becomes `tlload`'s output target. `-d delay` sets the delay between graph updates to `delay` seconds (if `delay` is 0, the graph never updates). `-s scale` sets the vertical height of the graph in scale characters. Thus, the smaller the value of `scale`, the larger the scale. To exit from `tlload`, press `Ctrl+c`. Figure 28-1 illustrates `tlload`'s output.

Signaling Processes

The commands this section discusses (see the following list) can be used to send any signal, not just one of the kill signals, to a running process. Kill signals are the most common, of course, but the complete list of possible signals is significantly longer. This section covers the following commands:

- **kill** — Sends a signal to a process
- **pkill** — Kills or sends another signal to a process matching a given pattern
- **killall** — Kills processes you specify by name

Most Linux users are familiar with the `kill` command. Note, however, that most shells, including `bash`, have a built-in `kill` command. The shell's `kill` is executed before `/bin/kill` in most shells because they execute built-in commands, aliases, and functions, where applicable, before using a command in the `PATH`. The discussion here covers the `kill` command `/bin/kill`, not the shell command. `kill`'s syntax is:

```
kill [-s signal | -p] [-a] [--] pid
```

```
kill -l
```

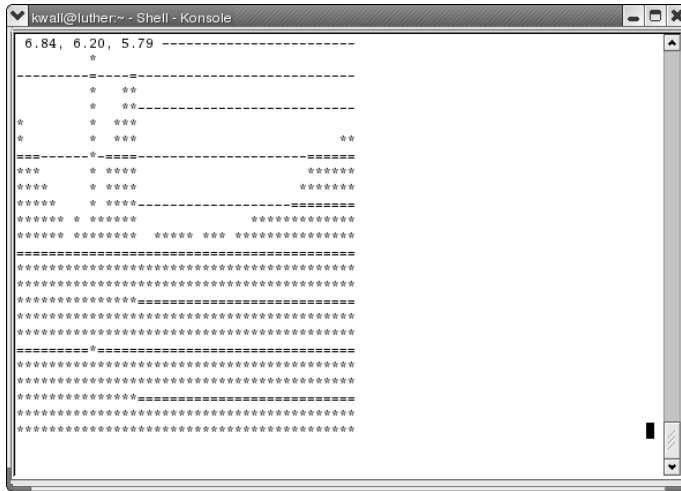


Figure 28-1 `top`'s graphical process monitor.

`pid` is one or more white-space-delimited tokens specifying the processes to kill. Each token can be a PID (where `pid > 0`); a process name; `-1`, which kills all processes with PIDs between 2 and 35,767; or `-pid`, which kills all processes in the process group whose PGID (Process Group ID) is `pid`. If you specify `-a`, all processes matching `pid` are killed (only root may use `-a`). `-p` lists only the PID; no processes are killed. `-s` signal indicates the signal to send; signal can be either a numeric or symbolic name, as shown with `kill -l`.

`pkill`, which has a comparable call interface to `pgrep`, sends a signal to one or more processes matching a given pattern. Its syntax is:

```
pkill [-signal] [-fnvx] [-P ppid] [-u euid] [-U uid] [-G gid] [pattern]
```

If you specify multiple criteria, the selected processes match all criteria, so keep the selection criteria as simple as possible. `pattern` contains the expression to match against process names or command lines. `-signal` specifies the numeric or symbolic signal to send (`SIGTERM` is the default). `-f` matches `pattern` against the process name (the default if `-f` is not specified) or the full command line. If multiple processes match `pattern` and other criteria, `-n` sends the signal to the most recently started process. `-v` reverses the matching, showing all processes not matching the specified criteria. `-x` forces an exact match of `pattern`. `-P ppid` sends the signal to processes whose parent process has a PID of `ppid`. Similarly, `-u euid`, `-U uid`, and `-G gid` send the signal to processes whose EUIDs, UIDs, and/or GIDs, respectively, match `euid`, `uid`, and/or `gid`. Multiple `ppids`, `euids`, `uids`, and `gids` may be specified by separating each with commas.

`killall` kills all processes matching a name. Its syntax is:

```
killall [-l] | [-giqvw] [-signal] name
```

Specifying `-l` lists the numeric value and symbolic names of all recognized signals, and it cannot be used with any other options or arguments. `name` lists the command whose process should be signaled. Multiple names may be specified if separated by white space. `-signal` specifies the signal to send; `SIGTERM` is the default. `-g` kills the process group to which `name` belongs. `-i` runs `killall` in interactive mode and requests confirmation for each name signaled. `-q` suppresses error messages if no processes match `name`. `-v` displays a short message if the signal was sent successfully. `-w` instructs `killall` to wait for each name to die.

Modifying Process Priorities

In some cases it is not necessary to terminate a process, but simply to modify the priority at which it runs. The following two commands accomplish this goal:

- **nice** — Starts a program with a given scheduling priority
- **renice** — Alters the scheduling priority of one or more running processes

The `nice` command enables you to start a program with a higher or lower nice number, which controls how much CPU time the kernel gives a program relative to other programs with the same priority. The `nice` command's syntax is:

```
nice [-n value | -value] [prog [arg]]
```

`prog` is the program to run. `arg` lists the arguments to `prog`, and it is specified using the format `prog` understands. `value` expresses the modified nice number at which to run `prog`. Valid values for `value` are -20 to 19, with smaller values representing a higher priority relative to other programs.

Use the `renice` command to modify the CPU scheduling priority of a running process. Its syntax is:

```
renice priority [[-p] pid] [[-g] pgrp] [[-u] user]
```

`priority` expresses the new nice number at which to run the specified process(es). `-p pid` identifies the processes to modify by PID. `-g pgrp` modifies all processes in the process group `pgrp`. `-u user` causes all processes owned by `user` to run at the new priority.

Maintaining the File System

A significant portion of routine administrative time and effort involves file system maintenance. These tasks include modifying existing file systems, creating new ones, fixing broken ones, monitoring all of them, and ensuring that users do not monopolize any of them. The file system maintenance commands this section covers have been divided into three categories: commands for creating and maintaining file systems, commands for working with files and directories, and commands for managing disk space usage.

Working with File Systems

Unless you have an extremely active system, creating and formatting a file system is an infrequent necessity. Actively maintaining a file system, however, is an ongoing process. The commands for creating, formatting, and checking the integrity of Fedora Core and RHEL file systems discussed in this section are:

- Working with partitions
 - **fdisk** — Creates, deletes, and modifies hard disk partitions
 - **parted** — Edits hard disk partition tables
- Working with file systems
 - **mkfs** — Creates a file system on a device
 - **mkswap** — Creates a swap partition or file
 - **e2fsck** — Checks, and optionally repairs, a Linux file system
 - **resize2fs** — Resizes ext2 and ext3 file systems
 - **symlinks** — Validates, and optionally repairs, symbolic links
- Mounting and unmounting file systems
 - **mount** — Mounts a file system
 - **umount** — Unmounts a file system
 - **swapoff** — Disables a swap partition or file
 - **swapon** — Enables a swap partition or file

Creating and Manipulating Partitions

The **fdisk** program prepares hard disks to hold Linux file systems. **fdisk** is an interactive program that accepts few command line options. You can invoke it using one of the following forms:

```
fdisk -s partition
```

```
fdisk [-lu] device
```

The first form uses the `-s` option to display the size in (blocks) of the disk partition specified by `partition` and then exits. The second form operates on the disk specified by `device`. The `-l` option lists the disk geometry of `device`, followed by a columnar list of each partition on `device` showing each partition's boot status, starting and ending cylinders, total size (in 512-byte blocks), and the partition type. If `device` is omitted, `fdisk` lists the same information based on the contents of the file `/proc/partitions`. The `-u` option instructs `fdisk` to show disk and partition sizes in terms sectors instead of cylinders. Omitting `-l` (second form) starts an interactive `fdisk` session on `device`.

The `parted` program manipulates existing partitions. You can use it check, create, delete, resize, move, and copy the following partition types:

- ext2
- ext3
- FAT
- FAT32
- Linux swap

`parted`'s general syntax is:

```
parted [opts] [dev [cmd [cmd_opts]]]
```

To force `parted` to run in interactive or batch (script) mode, `opts` may be `-i` or `-s`, respectively. Interactive mode is the default. `dev` can specify any block device on which partitions can be created, such `/dev/hdb` or `/dev/sdc` or even a floppy disk (`/dev/fd0`). The `parted` rubber meets the road with the `cmd` argument and any associated command options specified in `cmd_opts`. Table 28-5 lists possible `parted` commands.

Table 28-5 GNU `parted` Commands

COMMAND	DESCRIPTION
<code>check</code>	Checks a partition.
<code>cp</code>	Copies the contents of one partition to another.
<code>help</code>	Prints <code>parted</code> usage instructions.
<code>mkfs</code>	Creates a file system on a partition.

Table 28-5 (continued)

COMMAND	DESCRIPTION
<code>mkpart</code>	Creates a primary, logical, or extended partition by specifying the starting and ending size in MB.
<code>mkpartfs</code>	Creates a primary, logical, or extended partition by specifying the starting and ending size in MB and then creates a file system of a specified type on the newly created partition.
<code>move</code>	Moves a partition by changing the starting and ending blocks, specified in MB.
<code>print</code>	Displays the current partition table.
<code>quit</code>	Exits parted.
<code>resize</code>	Resizes a partition by changing the starting and ending blocks, specified in MB.
<code>rm</code>	Deletes a partition.
<code>select</code>	Chooses the device to edit.
<code>set</code>	Changes or sets flags on a disk partition. Valid flags are <code>boot</code> , <code>root</code> , <code>swap</code> , <code>hidden</code> , <code>raid</code> , <code>lvm</code> , <code>lba</code> , and <code>palo</code> .

CAUTION Exercise extreme care when using parted or any other partition editor to resize or manipulate partition tables. The tools themselves usually work fine and don't exhibit any unexpected behavior. Nonetheless, it is simple for operator error to render a disk unbootable with a stray keystroke.

Most of the commands listed in Table 28-5 accept one or more `cmd_opts`, which are options that specify the device or partition on which to operate, a starting and ending value, and a file system type. For complete details, refer to the parted Info page (`info parted`); less complete but still useful information can be found in the parted man page (`man parted`).

Creating and Manipulating File Systems

`mke2fs` creates a Linux ext2 or ext3 file system on a disk. Its syntax is:

```
mke2fs [-c | -l list] [-b size] [-i bytes-per-inode] [-j] [-n]
        [-m reserve] [-F] [-q] [-v] [-L label] [-S device]
```

`device` indicates the disk partition or other device on which to create the file system. Specifying `-n` results in a test run; `mke2fs` goes through the entire

creation process but does not actually create the file system. Use `-q` to suppress output, for example, when `mke2fs` is used in a script. Conversely, use `-v` to generate verbose output.

To check the disk for bad blocks while creating the file system, specify `-c`, or use `-l list` to read a list of known bad blocks from the file named `list`. By default, `mke2fs` calculates file system block sizes based on the size of the underlying partition, but you can specify `-b size` to force a block size of 1024, 2048, or 4096 bytes. Similarly, to override the default inode size, use `-i bytes-per-inode` (bytes-per-inode should be no smaller than the block size defined with `-b size`). `-m reserve` instructs `mke2fs` to set aside reserve percent of the file system for the root user. If `-m reserve` is omitted, the default reserve space is 5 percent. `-L label` sets the file system's volume label, or name, to `label`.

Normally, `mke2fs` refuses to run if `device` is not a block device (a disk of some sort) or if it is mounted; `-F` overrides this default. `-F` is most commonly used to create a file that can be mounted as a loopback file system. `-S`, finally, causes `mke2fs` to write only the superblocks and the group descriptors and to ignore the block and inode information. In essence, it attempts to rebuild the high-level file system structure without affecting the file system contents. It should be used only as a final attempt to salvage a badly corrupted file system, and may not work. The manual page recommends running `e2fsck` immediately after using `-S`.

To create and manipulate swap space, use the `mkswap`, `swapon`, and `swapoff` commands. `mkswap` initializes a swap area on a device (the usual method) or a file. `swapon` enables the swap area for use, and `swapoff` disables the swap space. `mkswap`'s syntax is:

```
mkswap [-c] device [size]
```

`device` identifies the partition or file on which to create the swap area and `size` specifies the size, in blocks, of the swap area to create. `size` is necessary only if you want a swap area smaller than the available space. If `device` is a file, it must already exist and be sized appropriately. `-c` performs a check for bad blocks and displays a list of any bad blocks found.

TIP To create a swap file before using `mkswap`, use the following command:

```
dd if=/dev/zero of=/some/swap/file bs=1M count=128
```

Replace `/some/swap/file` with the file you want to create as a swap file.

To enable the kernel to use swap devices and files, use the `swapon` command. Its syntax takes three forms:

```

swapon -s
swapon -a [-ev]
swapon [-p priority] [-v] device

```

The first form displays a summary of swap space usage for each active swap device. The second form, normally used in system startup scripts, uses `-a` to activate all swap devices listed in `/etc/fstab`. If `-e` is also specified, `swapon` ignores devices listed in `/etc/fstab` that do not exist. The third form activates the swap area on `device`, and, if `-p priority` is also specified, gives `device` a higher priority in the swap system than other swap areas. `priority` can be any value between 0 and 32,767 (specified as 32767), where higher values represent higher priorities. `-v` prints short status messages.

`e2fsck` checks a file system for possible corruption and repairs any damage found. `e2fsck` is an ext2- and ext3-file-system-specific version of the more general `fsck` command. Ordinarily, you will use `fsck`, which is a wrapper program that invokes a file system-specific version of `fsck` depending on the type of file system. For example, if you call `fsck` on an ext2 or ext3 file system, it will invoke `e2fsck`; if you call `fsck` on a ReiserFS file system, `fsck` invokes `fsck.reiserfs`.

`e2fsck`'s syntax is:

```

e2fsck [-pcnyfvt] [-b sblock] [-B size] [-l list] device

```

`device` is the partition (`/dev/hda1`, for example) to test. `-b sblock` tells `e2fsck` to use the backup super block located on block number `sblock`. `-B size` specifies block sizes of `size` bytes. `-l list` instructs `e2fsck` to add the block numbers listed in the file name `list` to the list of known bad blocks. Using `-c` causes `e2fsck` to identify bad blocks on the disk. Ordinarily, `e2fsck` asks for confirmation before repairing file system errors; specifying `-p` disables any confirmation prompts, `-n` automatically answers "No" to all questions and sets the file system to read-only, and `-y` automatically answers "Yes" to all questions. `e2fsck`'s default behavior is not to check a file system that is marked clean, but using `-f` forces it to do so. `-v` enables verbose output. `-t` generates a timing report at the end of `e2fsck`'s operation.

If `e2fsck` discovers problems with one of your file systems that it cannot repair automatically, you might be able to use the `debugfs` program to repair the file system manually.

`resize2fs` makes it possible to resize ext2 and ext3 file systems without destroying existing data and, in certain cases, without having to use `fdisk` or parted to resize the partition. As with parted, use `resize2fs` with great care and make sure you have good backups of the data on the file system you intend to resize.

The `symlinks` command scans directories for symbolic links, displays them on the screen, and repairs broken or otherwise malformed symbolic links. Its syntax is:

```
symlinks [-cdrstv] dirlist
```

dirlist is a list of one or more directories to scan for symbolic links. `-r` causes `symlinks` to recurse through subdirectories. `-d` deletes dangling links, symbolic links whose target no longer exists. `-c` converts absolute links, links defined as an absolute path from `/`, to relative links, links defined relative to the directory in which the link is located. `-c` also removes superfluous `/` and `.` elements in link definitions. `-s` identifies links with extra `./` in their definition and, if `-c` is also specified, repairs them. To see what `symlinks` would do without actually changing the file system, specify `-t`. By default, `symlinks` does not show relative links; `-v` overrides this default.

To make an existing file system available, it has to be mounted using the `mount` command. `mount`'s syntax is:

```
mount -a [-fFnrsvw] [-t fstype]  
mount [-fnrsvw] [-o fsoptions] device | dir  
mount [-fnrsvw] [-t fstype] [-o fsoptions] device dir
```

The first two forms use the information in `/etc/fstab` when mounting file systems. When invoked with no options, `mount` lists all mounted file systems, and when you specify only `-t`, `fstype` lists all mounted file systems of type `fstype`. `fstype` will be one of `devpts`, `ext2`, `iso9660`, or `vfat`, but many other file system types are supported — the complete list of valid types is available in `mount`'s manual page.

The `-a` option mounts all the file systems listed in `/etc/fstab` (subject to the restriction of using the `-t` option as explained in the previous paragraph) that are configured using the `auto` mount option. (See Table 28-6.) The second form is most commonly used to override the mount options, using `-o fsoptions`, listed in `/etc/fstab`. Note that you only have to specify `device`, the device containing the file system, or `dir`, where in the directory hierarchy the file system should be attached.

Use the third form to mount file systems not listed in `/etc/fstab` or to override information it contains. The third form is also the most widely used. In general, it attaches the file system on `device` to the system's directory hierarchy at the mount point `dir`, using a file system type of `fstype` and the file system options `fsoptions`. Table 28-6 lists `mount`'s global options. `fsoptions` is a comma-delimited list of one or more of the options listed in Table 28-7.

NOTE Because Linux supports so many file systems, this chapter discusses only a few of the many file systems and file system options. `mount`'s manual page contains a complete list of the file systems and their corresponding mount options that Linux currently supports.

Table 28-6 Global Options for the mount Command

OPTION	DESCRIPTION
<code>-a</code>	Mounts all file systems, subject to restrictions specified using <code>-t</code>
<code>-F</code>	Mounts all file systems (used only with <code>-a</code>) in parallel by creating new processes for each file system to mount
<code>-f</code>	Fakes the mount operation, doing everything but actually mounting the file system
<code>-h</code>	Displays a short usage message
<code>-n</code>	Mounts the file system without creating an entry in the mount table (<code>/etc/mtab</code>)
<code>-o fsoptions</code>	Mounts the file system using the file system-specific options <code>fsoptions</code>
<code>-r</code>	Mounts the file system in read-only mode
<code>-s</code>	Ignores options specified with <code>-o</code> that are invalid for the given file system type (the default is to abort the mount operation)
<code>-t fstype</code>	Restricts mount's operation to file system types of type <code>fstype</code> (first and second forms) or specifies the file system type of the file system being mounted (third form)
<code>-v</code>	Prints informational messages while executing (verbose mode)
<code>-w</code>	Mounts the file system in read/write mode

Table 28-7 Common File System Options for the mount Command

OPTION	TYPE*	DESCRIPTION
<code>async</code>	1	Enables asynchronous system I/O on the file system
<code>auto</code>	1	Enables mounting using the <code>-a</code> option
<code>defaults</code>	1	Enables the default options (<code>rw, suid, dev, exec, auto, nouser, async</code>) for the file system
<code>dev</code>	1	Enables I/O for device files on the file system
<code>exec</code>	1	Enables execution of binaries on the file system

(continued)

Table 28-7 (continued)

OPTION	TYPE*	DESCRIPTION
gid=gid	2,3	Assigns the GID <code>gid</code> to all files on the file system
mode=mode	3	Sets the permissions of all files to <code>mode</code>
noauto	1	Disables mounting using the <code>-a</code> option
nodev	1	Disables I/O for device files on the file system
noexec	1	Disables execution of binaries on the file system
nosuid	1	Disables set-UID and set-GID bits on the file system
nouser	1	Permits only root user to mount the file system
ro	1	Mounts the file system in read-only mode
remount	1	Attempts to remount a mounted file system
rw	1	Mounts the file system in read/write mode
suid	1	Enables set-UID and set-GID bits on the file system
sync	1	Enables synchronous file system I/O on the file system
user	1	Permits nonroot users to mount the file system
uid=uid	2,3	Assigns the UID <code>uid</code> to all files on the file system

1 = All file systems, 2 = devpts, 3 = iso9660

To unmount a file system, use the command `umount`. Its syntax is much simpler, thankfully, than `mount`'s:

```
umount -a [-nrv] [-t fstype]
umount [-nrv] device | dir
```

All of `umount`'s options and arguments have the same meaning as they do for `mount`, except for `-r`. Of course, the options must be understood in the context of unmounting a file system. If `-r` is specified and unmounting a file system fails for some reason, `umount` attempts to mount it in read-only mode.

To access swap space, use the `swapon` and `swapoff` commands. To enable the kernel to use swap devices and files, use the `swapon` command. Its syntax takes three forms:

```
swapon -s
swapon -a [-ev]
swapon [-p priority] [-v] device
```

The first form displays a summary of swap space usage for each active swap device. The second form, normally used in system startup scripts, uses `-a` to activate all swap devices listed in `/etc/fstab`. If `-e` is also specified, `swapon` ignores devices listed in `/etc/fstab` that do not exist. The third form activates the swap area on device, and, if `-p priority` is also specified, gives device a higher priority in the swap system than other swap areas. `priority` can be any value between 0 and 32,767 (specified as 32767), where higher values represent higher priorities. `-v` prints short status messages.

To deactivate a swap area, use the `swapoff` command. Its syntax is simple:

```
swapoff -a | device
```

Use `-a` to deactivate all active swap areas, or use `device` to deactivate a specific swap area. Multiple swap areas may be specified using white space between device identifiers.

Working with Files and Directories

This section reviews the basic call syntax of the following commands:

- **chmod** — Modifies file and directory permission settings
- **chown** — Modifies file and directory user ownership
- **chgrp** — Modifies file and directory group ownership
- **lsattr** — Lists special file attributes on ext2 files
- **chattr** — Modifies special file attributes on ext2 files
- **stat** — Shows detailed file information
- **fuser** — Displays a list of process IDs using a file
- **lsof** — Identifies files opened by a process

Here are the syntax summaries for `chmod`, `chown`, and `chgrp`:

```
chmod [-cfRv] symbolic_mode file
chmod [-cfRv] octal_mode file
chown [-cfhRv] owner[:[group]] file
chown [-cfhRv] :group file
chgrp [-cfhRv] group file
```

`chmod`, `chown`, and `chgrp` accept the common options `-c`, `-v`, `-f`, `-R`, and `file`. `file` is the file or directory to modify, and multiple file arguments can be specified. `-R` invokes recursive operation on the subdirectories of the current working directory or of a directory specified by `file`. `-v` generates a diagnostic for each file or directory examined. `-c` generates a diagnostic message only when it changes a file. `-f` cancels all but fatal error messages.

`chmod` has two forms because it understands both symbolic and octal notation for file permissions. For both forms, `file` is one or more files on which permissions are being changed. `symbolic_mode` uses the symbolic permissions notation, while `octal_mode` expresses the permissions being set using the standard octal notation.

CROSS-REFERENCE For a quick refresher on using symbolic and octal permissions notation, refer to the `chmod` manual page.

With the `chown` and `chgrp` commands, `group` is the new group being assigned to `file`. For the `chown` command, `owner` identifies the new user being assigned as `file`'s owner. The colon (:) enables `chmod` to change `file`'s group ownership. The format `owner:group` changes `file`'s user and group owners to `owner` and `group`, respectively. The format `owner:` changes only `file`'s owner and is equivalent to `chown owner file`. The format `:group` leaves the owner untouched but changes `file`'s group owner to `group` (equivalent to `chgrp group file`).

The `lsattr` and `chattr` commands are Linux-specific, providing an interface to special file attributes available only on the `ext2` and `ext3` file systems. `lsattr` lists these attributes, and `chattr` sets or changes them. `lsattr`'s syntax is:

```
lsattr [-adRVv] file
```

`file` is the file or directory whose attributes you want to display; multiple white space separated file arguments may be specified. `-a` causes the attributes of all files, such as hidden files, to be listed. `-d` lists the attributes on directories, rather than listing the contents of the directories, and `-R` causes `lsattr` to recurse through subdirectories if `file` names a subdirectory.

`chattr`'s syntax is:

```
chattr [-RV] [-v version] [+|-|=mode file
```

`file` is the file or directory whose attributes you want to display; multiple white space separated file arguments may be specified. `-R` causes `lsattr` to recurse through subdirectories if `file` names a subdirectory. `-v version` sets a version or generation number for `file`. `+mode` adds `mode` to `file`'s attributes; `-mode` removes `mode` from `file`'s attributes; `=mode` sets `file`'s attributes to `mode`, removing all other special attributes. `mode` can be one or more of the following:

- **A** — Do not change `file`'s time (last access time)
- **S** — Update `file` synchronously

- **a** — File is append-only
- **c** — Kernel automatically compresses/decompresses file
- **d** — File cannot be dumped with the dump command
- **i** — File is immutable (cannot be changed)
- **s** — File will be deleted securely using a special secure deletion algorithm
- **u** — File cannot be deleted

The `stat` command displays detailed file or file system status information. Its syntax is:

```
stat [-l] [-f] [-t] file
```

`file` specifies the file or directory about which you want information. Use multiple white-space-delimited file arguments to specify multiple files. If `-l` is used and `file` is a link, `stat` operates on the link's target (the file that is linked) rather than the link itself. Using `-f` causes `stat` to display information about `file`'s file system, not `file`. Specifying `-t` results in a shorter (terse) output format suitable for use in scripts.

Often, an administrator needs to identify the user or process that is using a file or socket. `fuser` provides this functionality. Its syntax is:

```
fuser [-a | -s] [-n namespace] [-signal] [-kimuv] name
```

`name` specifies the file, file system, or socket to query. By default, `fuser` assumes that `name` is a filename. To query TCP or UDP sockets, use `-n namespace`, where `namespace` is `udp` for UDP sockets and `tcp` for TCP sockets (`file` is the default namespace). `-a` results in a report for all names specified on the command line, even if they are not being accessed by any process. `-s`, on the other hand, causes `fuser` to run silently. You cannot use `-s` with `-a`, `-u`, or `-v`. `-k` kills processes using `name` with the signal `SIGKILL`; use `-signal` to specify an alternate signal to send. Use `-i` (interactive) to be prompted for confirmation before killing a process. Only use `-i` with `-k`. `-m` indicates that `name` specifies a file system or block device, so `fuser` lists all processes using files on that file system or block device. `-u` adds the username of a process's owner to its output when listing processes. `-v`, finally, generates a verbose, `ps`-like listing of processes using the specified name.

For example, to see what process and user is using the Berkeley socket file `/tmp/.X11-unix/X0`, the following command would do:

```
# fuser -u /tmp/X11-unix/X0
/tmp/.X11-unix/X0:    3078(root)
```

This command used the `-u` option to display the username (root) running the displayed process (3078). For a more verbose listing, add the `-v` option:

```
# fuser -uv /tmp/.X11-unix/X0

          USER          PID ACCESS COMMAND
/tmp/.X11-unix/X0    root      3078 f....  X
```

`lsuf` performs the reverse function from `fuser`, showing the files open by a given process or group of processes. A simplified version of its syntax is:

```
lsuf [-LlNRst] [-c c] [+f | -f] [+r | -r [t]] [-S [t]] [file]
```

`file` specifies the file or file systems (multiple file arguments are permitted) to scan. Specifying `-c c` selects processes executing a command that begins with the letter `c`. `-f` causes `file` to be interpreted as a file or pathname, `+f` as a file system name. `-L` suppresses displaying the count of files linked to `file`. `-l` displays UIDs rather than converting them to login names. Specifying `-N` includes a list of NFS files in `lsuf`'s output. `+r` causes `lsuf` to repeat the display every 15 seconds (or `t` seconds if `t` is specified) until none of the selected files remains open; `-r` repeats the display indefinitely. `-R` lists the parent process ID of displayed processes. `-S` enables `lsuf` to time out after 15 seconds, or after `t` seconds if `t` is specified.

One of the most common uses of `lsuf` is to find out what file (or files) are preventing you from unmounting a file system. As you might have experienced, you cannot unmount a file system when a file that resides on it is still open. If you attempt to do this, `umount` complains that the file system is busy. For example, suppose that you want to unmount `/dev/fd0`, which is mounted on the file system `/mnt/floppy`:

```
# umount /mnt/floppy
umount: /mnt/floppy: device is busy
umount: /mnt/floppy: device is busy
```

Nuts. Undeterred, you use the `lsuf` command to determine what files are open on the `/mnt/floppy`:

```
# lsuf /mnt/floppy

COMMAND  PID  USER  FD  TYPE DEVICE SIZE NODE NAME
bash      4436 bubba  cwd  DIR   2,0 1024    2 /mnt/floppy
cat      11442 bubba  cwd  DIR   2,0 1024    2 /mnt/floppy
cat      11442 bubba   1w  REG   2,0    0   12 /mnt/floppy/junk
```

Now, you can use the `kill` command to kill the processes that are keeping you from unmounting `/mnt/floppy`:

```
# kill 11442 4436
# lsof /mnt/floppy
COMMAND  PID  USER  FD   TYPE DEVICE SIZE NODE NAME
bash      4436 bubba  cwd   DIR   2,0 1024    2 /mnt/floppy
```

The bash shell bubba is running isn't dead, so use more force:

```
# kill -KILL 4436
# lsof /mnt/floppy
# umount /mnt/floppy
```

Managing Disk Space Usage

Monitoring and controlling disk space usage is another important part of a system administrator's tasks. The commands covered in this section for managing disk space usage include the following:

- **df** — Shows available (free) disk space on mounted file systems
- **du** — Shows disk space usage for files, directories, and file systems
- **edquota** — Modifies user disk space quota limits
- **quota** — Displays current disk usage and disk usage limits
- **quotaoff** — Disables disk quotas on file systems
- **quotaon** — Enables disk quotas on file systems
- **quotactl** — Manages the quota system
- **quotastats** — Prints statistics about the quota system
- **repquota** — Displays a report summarizing disk quota usage
- **setquota** — Sets disk quotas
- **quotacheck** — Compares disk usage to limits set using the quota system

CROSS-REFERENCE Implementing and using the quota subsystem is discussed in detail in Chapter 26. Please refer to that chapter for examples and illustrations of the quota commands introduced in this section.

The **df** and **du** commands perform complementary functions, listing detail and summary information about the amount of disk space free and used, respectively. **df**'s syntax is:

```
df [-ahklTmx] [-t type] [--sync|--nosync] [name]
```


name, which can contain multiple white space delimited values, is the name of a file whose file system should be checked, or the file system itself (the default is all mounted file systems). `-a` includes empty file systems in the display, which would ordinarily be omitted. `-h` uses more familiar display units, such as GB, MB, or KB, rather than default, blocks. `-k` causes `df` to use block sizes of 1024 bytes, and `-m` block sizes of 1,048,576 bytes. `-l` limits `df`'s report to local file systems, ignoring, for example, NFS mounted file systems. `-x` limits `df`'s report to the current file system or the file system to which name refers. `-t type` limits the report to file systems of `type`, and `--nosync` prevents `df` from syncing file systems before generating its report (the default is to sync the disks to obtain the most accurate report).

`du` displays information about the disk space used. Its syntax is:

```
du [-abcDhklmSsx] [-X file] [--exclude=path] [--max-depth=n] [name]
```

name, which can contain multiple white space delimited values, is the name of a file whose file system should be checked, or the file system itself (the default is all mounted file systems). `-a` displays counts for all files, not just directories. `-b` prints all sizes in bytes. `-c` displays a grand total for names. `-h` uses more familiar display units, such as GB, MB, or KB, rather than default, blocks. `-k` causes `df` to use block sizes of 1024 bytes, and `-m` block sizes of 1,048,576 bytes. `-l` limits `df`'s report to local file systems, ignoring, for example, NFS mounted file systems. If a file or directory in name includes a symbolic link, `-L` causes it to be dereferenced to obtain the target's disk usage, rather than the link's usage. `-S` ignores subdirectories, which are recursed by default. `-s` results in a summary total for each name rather than a detailed report for each file in name. `-x` limits `du`'s report to the current file system or the file system to which name refers. `-X file` causes `du` to ignore any file matching a pattern contained in `file`. Similarly, use `--exclude=pattern` to specify a single pattern to ignore. `--max-depth=n`, finally, limits the displayed report to directories (or files, if `--all` specified) within `n` levels of a path specified by name (all directories and files are evaluated, but the granularity of the report is limited to `n` levels).

Timekeeping

In most situations, maintaining the system date and time is a secondary concern. In larger networks, however, particularly those with multiple servers, synchronizing the time across the network is considerably more important. This is especially true for file and database servers, which use finely grained time values to govern disk writes, reads, and to maintain logs of their activity should one or more operations need to be rolled back. This section discusses

key commands for showing, setting, and maintaining the date and time on a Red Hat system, specifically:

- **hwclock** — Displays and sets the hardware clock
- **date** — Displays and sets the system time and date
- **rdate** — Displays and sets the system clock from a network time server
- **ntpd** — Keeps system time synced to one or more time servers

Single-Use Commands

The `hwclock`, `date`, and `rdate` commands are single-use commands for setting the system date and time. That is, `hwclock`, `date`, and `rdate` have no inherent ability to keep a system's clock synced. Rather, you run one of them, the time is set, and you are done. Unless executed from cron or another periodic command scheduling service, none of these commands work to keep system time accurate on an ongoing basis.

The `hwclock` command displays and sets the hardware clock. Its syntax is:

```
hwclock [-a | -r | -s | -u | -w | --set --date=newdate]
```

`hwclock` invoked by itself or with the `-r` option displays the current time, converted to local time, as maintained by the computer's hardware clock (often called the RTC, or real-time clock). Specifying `-w` updates the hardware clock with the system time, while `-s` updates the system time based on the hardware time. The following examples first show the current system and hardware time, then update the hardware clock to the system time using the `-w` option:

```
# date
Thu Apr 20 09:22:48 EDT 2006
# hwclock
Thu 20 Apr 2006 09:22:49 AM EDT -0.687590 seconds
# hwclock -w
# hwclock
Thu 20 Apr 2006 09:22:56 AM EDT -0.498212 seconds
```

Note that after syncing the hardware clock to the system clock, the hardware clock gained approximately 13 seconds (of course, some time elapsed while the commands were typed). Using `hwclock -w` or `hwclock -s` in a system initialization script (or, as you will see shortly, using `rdate` to sync the system and hardware time to an external time source), enables you to maintain accurate and consistent time on your Fedora Core or RHEL system.

CAUTION Updating the system time after the system has booted could cause unpredictable behavior. Do not use the `-s` option except early in the system initialization process.

Use the `-u` option to tell `hwclock` that the time stored in the hardware clock is maintained in UTC (Coordinated Universal Time) format, rather than in the local time (the default). Yes, you read that correctly. The acronym almost always appears as UTC, even though it refers to Coordinated Universal Time or Universal Coordinate Time — just another one of those little Linux/UNIX idiosyncrasies. The `-a` option enables you to adjust the hardware clock's time to account for systematic drift in its time. `--set`, finally, sets the hardware clock to the date and time specified by the `newdate` argument to the `--date` option. `newdate` can be any date in a format accepted by the `date` command. The next example shows how to use the `--set` argument to update the system time:

```
# hwclock --set --date="July 8, 2006 7:24 PM"
# hwclock
Sat 08 Jul 2006 07:24:05 PM EDT 0.429153 seconds
```

As you will see in the discussion of the `date` command, you can use practically any common date and time specification to set the date using `hwclock` and `date`.

The `date` command displays the current time in a specified format or sets the system time to the specified value. Its syntax comes in two forms:

```
date [-d datestr] [-f datefile] [-r reffile] [+format]
date [-u] [MMDDhhmm[[CC]YY][.ss]] | -s datespec
```

The first form displays the time in the format specified by `format` subject to modification by one of the `-d`, `-f`, `-r`, or `-u` options. By default, `date` prints the current date and time, but specifying `-d datestr` prints the date and time in `datestr`; `-f datefile` prints the date and time of the date strings contained in the file `datefile` (one per line); and `-r reffile` prints the date and time that `reffile` was last modified. The next three examples show `date`'s output using these first three options.

```
$ date -d "July 6, 2006 11:48 AM"
Thu Jul 6 11:48:00 EDT 2006
$ cat datefile
January 1, 2010 00:01:01
December 31, 2010 11:59:59 PM
[root@luther /root]# date -f datefile
Fri Jan 1 00:01:01 MST 2010
Fri Dec 31 23:59:59 MST 2010
$ date -r /boot/vmlinuz
Sat Jul 8 18:57:28 EDT 2006
```

Note that regardless of how the source or input date is formatted, `date`'s output always has the same format. To modify the output format, use the `+format` argument. Specify format using one or more of the tokens listed in Table 28-8.

Table 28-8 Output Format Tokens for the `date` Command

TOKEN	DESCRIPTION
%a	Prints the locale's three-letter abbreviated weekday name (Sun–Sat)
%A	Prints the locale's full weekday name (Sunday–Saturday)
%w	Prints the day of the week (0–6, 0 represents Sunday)
%d	Prints the day of the month (01–31)
%e	Prints the blank padded day of the month (1–31)
%j	Prints the day of the year (001–366)
%U	Prints the week number of the year, with Sunday as the first day of the week (00–53)
%V	Prints the week number of the year, with Monday as the first day of the week (01–52)
%W	Prints the week number of the year, with Monday as the first day of the week (00–53)
%b	Prints the locale's three-letter abbreviated month name (Jan–Dec)
%B	Prints the locale's full month name (January–December)
%m	Prints the two-digit month (01–12)
%y	Prints the last two digits of the year (00–99)
%Y	Prints the four-digit year (1970)
%D	Prints the date in US format (<i>mm/dd/yy</i>)
%x	Prints the locale's date representation
%S	Prints the two-digit second (00–60)
%M	Prints the two-digit minute (00–59)
%H	Prints the two-digit hour in 24-hour format (00–23)
%I	Prints the two-digit hour in 12-hour format (01–12)
%p	Prints the locale's AM or PM
%Z	Prints the time zone (for example, EDT) or nothing if no time zone is determinable

(continued)

Table 28-8 Output Format Tokens for the date Command

TOKEN	DESCRIPTION
%T	Prints the 24-hour time (hh:mm:ss)
%r	Prints the 12-hour time (hh:mm:ss AM PM)
%X	Prints the locale's time representation (same as %H:%M:%S)
%c	Prints the locale's date and time (Sat Jul 08 12:02:33 EDT 2006)
%s	Prints the seconds elapsed since 00:00:00, Jan 1, 1970
%%	Prints a literal %
%n	Prints a newline
%t	Prints a horizontal tab

Here are some examples using `+format`. The first example prints the four-digit year, the Julian day, the hour in 24-hour format, the minute, and the second, separating each element with a hyphen (-). Note that characters, such as the hyphen, that are not part of a formatting token (prefixed with %) are interpreted literally as part of the output.

```
$ date +%Y-%j-%H-%M-%S
```

```
2006-190-20-44-05
```

The next example mimics `date`'s standard output for U.S. locales. Note that because the format string contains spaces, you have to use strong (') or weak quotes (") to prevent the shell from interpreting the spaces:

```
$ date +' %a %b %e %H:%M:%S %Z %Y'
```

```
Sun Jul  9 20:49:24 EDT 2006
```

The final example shows the current date and time using full names for the month and day using the standard 12-hour time format:

```
$ date +" %A, %B %d, %Y%n%-I:%M %p"
```

```
Sunday, July 09, 2006
8:59 PM
```

The example also used the `%n` to insert a newline and the `-` modifier between % and I to remove the default padding GNU `date` inserts into numeric fields. Again, because the format string used spaces, the string had to be surrounded with quotes.

Use the second form of the `date` command to set the system date and time. Use `-u` to indicate that the specified date and time are relative to Coordinated Universal Time. The string `MMDDhhmmCCYY.ss` defines the time to set. The pairs of characters, in order, mean:

- **MM** — The month
- **DD** — The day
- **hh** — The hour
- **mm** — The minute
- **CC** — The century (optional)
- **YY** — The year (optional)
- **ss** — The second (optional)

For example, to set the current system date and time to 11:59 p.m. on December 31, 2006, you would execute the command (as root):

```
# date 123123592006
Sun Dec 31 23:59:00 MST 2006
```

Fortunately, you can also use more familiar date and time specifications. In fact, GNU `date` can interpret most commonly used date and time formats. To use this type of syntax, use the `-s` option and place the date in quotes if it contains embedded white space. For example, the following command sets the current date and time to 5:55:55 a.m. on May 5, 1955:

```
# date -s "May 5, 1955 5:55:55 am"
Thu May 5 05:55:55 MST 1955
```

The next command just sets the time, leaving the date untouched:

```
# date -s "9:33 PM"
Thu May 5 21:33:00 MST 1955
```

The last example, finally, corrects the date, but, unfortunately, has the side effect of resetting the time:

```
# date -s "07/08/2006"
Sat Jul 8 00:00:00 EDT 2006
```

The `rdate` command is a simple, effective way to maintain accurate system and hardware clock time on your system. Its syntax is:

```
rdate [-sp] host
```

`host` indicates the name of the network time server to contact. If `-p` is specified, `rdate` prints the time host returns and, if `-s` is also specified, the system time is set to the time host returns. `rdate`, like `hwclock`, is best used during system initialization. `rdate` needs network connectivity, so it must be executed after network has started, perhaps during one of the scripts executed when starting run level 3.

Using the Network Time Protocol

The Network Time Protocol, or NTP, is a standardized way to keep system time synchronized across a network. NTP consists of a daemon, `ntpd`, a configuration file, `/etc/ntp.conf`, and a set of supporting utilities (`ntpdate`, `ntpd`, `ntpdc`, `ntpdate`, and so forth) that, working together, keep your system's clock set. NTP is also quite simple to use on Fedora Core and RHEL systems because Red Hat configured it to sync against their NTP servers, `clock.redhat.com`. All you have to do is start it (if it is not already started):

```
# service ntpd start
```

To make sure NTP starts at each boot, use `chkconfig`:

```
# chkconfig --levels 345 ntpd on
# chkconfig --levels 0126 ntpd off
```

Automating Scripts

Admittedly, scripts enable a greater degree of customization, flexibility, and convenience in performing system administration tasks, but repeatedly typing script names soon becomes just as tedious and impractical as complete command line invocations. You can use `at` and `cron` to run commands automatically and unattended, enabling you to realize the full power and benefit of the scripting administrative tasks. Use the `at` command to schedule scripts you want to run later and that you want to run only once. Use the `cron` facility for programs and scripts that need to run on some sort of regular schedule.

Running One-Shot Jobs with `at`

The `at` command provides an interface to the `atd` daemon's scheduler. It is the `atd` daemon that executes the scripts. Scheduling a job with `at` is surprisingly simple to do: just type **`at`** followed by the time at which you want the

script to run, and then press Enter. For example, the following sequence of commands schedules the `/home/kwall/bin/incrback.sh` script to run at 1:05 a.m. tomorrow.

```
$ at 1:05
warning: commands will be executed using (in order) a) $SHELL b) login
shell c) /bin/sh
at> /home/kwall/bin/incrback.sh
at> <EOT>
job 1 at 2006-04-20 01:05
```

The `at>` prompt is a small interactive shell for scheduling commands. The initial command, `at 1:05`, used the simplest possible format for specifying the time. Table 28-9 shows additional options for indicating time to `at`. Once you have the `at>` prompt, enter scripts and other commands just as you would at the shell prompt. To specify multiple commands, press Enter after each command, type the next one, press Enter, and so on. Press `Ctrl+D` after you have entered all the commands you want. `at` responds with the `<EOT>` sign and then displays a job number (1, in the example) for your commands and the date and time (April 20, 2006 at 1:05 a.m.) the job will execute.

Table 28-9 Specifying Time with the `at` Command

COMMAND	WHEN THE JOB RUNS
<code>at now</code>	Executes the job immediately.
<code>at now + 15 minutes</code>	Executes the job 15 minutes from now.
<code>at now + 2 hours</code>	Executes the job 2 hours from now.
<code>at now + 10 days</code>	Executes the job 10 days from now.
<code>at noon</code>	Executes the job at noon today. If it is past noon, the job executes tomorrow at noon.
<code>at now next hour</code>	Executes the job 60 minutes from now.
<code>at 15:00 tomorrow</code>	Executes the job at 3:00 p.m. tomorrow.
<code>at 1:05am</code>	Executes the job at 1:05 a.m. today (if it is past 1:05 a.m., the job is executed tomorrow at 1:05 a.m.).
<code>at 3:00 Aug 16, 03</code>	At 3:00 a.m. on August 16, 2003.

To view the current list of jobs in `atd`'s queue, use the `atq` command. To remove a job from the queue, use the `atrm` command. The following commands show the current list of jobs using `atq`, and then remove them using `atrm`.


```
$ atq
1      2006-04-20 01:05 a kwall
2      2006-04-21 01:05 a kwall
$ atrm 1 2
```

The first field of `atq`'s output shows its job number, the same one displayed when you submitted the job. The rest of the fields are, in order, date and time the job will execute, the queue (a, in this case), and the user who submitted the job. Only the root user can see all of the jobs submitted; normal users can see only their own. Removing a job from the queue is a simple matter of typing **atrm** followed by the job number of the job you want to remove, as shown in the example.

Running Regularly Scheduled Jobs with cron

To run a script automatically at regular intervals, use the cron service. You schedule repeating jobs using the `crontab` command, either by placing the script name and scheduling information in a specially formatted file that `crontab` reads, or using `crontab` interactively. The cron daemon, `crond`, takes care of running the job and, as with `at`, emails its output to you.

To use cron, you need to understand the format of its job file, also called a `crontab` (which should not be confused with the `crontab` command, although the two are related). Each job in a `crontab` file is specified on a single line and each line contains at least six fields. The first five define when the job executes, and the sixth and subsequent fields name the script or program to run, along with any arguments the script or program takes. For example, this line executes the `incrback.sh` shell script at 1:05 a.m. each day:

```
05 01 * * * incrback.sh
```

Table 28-10 lists the meaning of the first five fields.

Table 28-10 crontab Field Values

FIELD	DESCRIPTION	VALID VALUES
1	Minute	0–59
2	Hour	0–23
3	Day of month	0–31
4	Month	1–12 (1 is January) Three letter month abbreviations (Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec)
5	Day of week	0–7 (0 and 7 both mean Sunday, 1 is Monday) Three letter day abbreviations (Sun, Mon, Tue, Wed, Thu, Fri, Sat)

Entries may be single values, a comma-delimited set of values to specify multiple days, a range of values separated by a hyphen (-), or any combination of these three options. An asterisk (*) represents all possible values for that field. For example, an asterisk in the hour field means a job would execute every hour of the day.

For example, to run `incrback.sh` at 4:55 p.m. on the 1st and 15th of January, March, June, and September, the crontab entry would look like one of the following:

```
55 16 1,15 1,3,6,9 * incrback.sh
55 16 1,15 Jan,Mar,Jun,Sep * incrback.sh
```

In this case, the * in the day of the week field is ignored because it is overridden by the other date and time specifications.

The easiest way to schedule a job with cron is to place the crontab entry in a file, and then invoke the `crontab` command with the filename as an argument. So, if one of the crontab entries shown previously was stored in a file named `kwall.crontab`, you would submit the job using the following command:

```
$ crontab kwall.crontab
```

To verify that the job is indeed scheduled, type `crontab -l` to list your crontab entries:

```
$ crontab -l
# DO NOT EDIT THIS FILE - edit the master and reinstall.
# (/tmp/crontab.2433 installed on Sun Jul  9 17:48:30 2006)
# (Cron version -- $Id: crontab.c,v 2.13 1994/01/17 03:20:37 vixie Exp
$)
55 16 1,15 1,4,6,9 * incrback.sh
```

To remove all of your cron jobs, type `crontab -r`. Once you become comfortable with the crontab format, you might find it most convenient to use crontab's interactive mode. To do so, type `crontab -e`. The interactive mode uses the vi editor, so use vi keystrokes to enter, edit, save your changes, and, of course, to exit the interactive mode.

Summary

In this chapter, you learned the basic commands that every system administrator must know how to use in a command line environment. You explored a number of commands for working with users and groups, including adding and deleting users and groups and how to modify multiple accounts simultaneously. You also learned how to obtain useful information about who is

logged in and what they are doing. Managing file systems is also important, and this chapter discussed commands for creating and checking file systems and managing disk space usage using quotas. The process administration commands discussed include commands for identifying which processes are active, killing processes, and modifying running processes. You also learned how to use `at` and `cron` to run scripts automatically, and got some final tips and hints on writing, testing, and debugging shell scripts.

Administering Users and Groups

IN THIS CHAPTER

- Administering User Accounts
- Understanding the Root Account
- Implementing Sudo
- Using File System Quotas

This chapter discusses the finer points of user and group maintenance on Fedora Core and RHEL systems. One feature of Red Hat–derived systems that most confuses newcomers is the user private group scheme. You will also learn how to add, modify, and delete user accounts and how to use Sudo to give normal users root capabilities on a limited and monitored basis. The final section shows you how to implement user and group file system quotas to control and monitor disk space usage.

Administering User Accounts

Administering users and groups, or, more precisely, administering user and group *accounts*, is a fundamental Linux system administration activity. Ordinarily, most people understand user accounts as accounts tied to a particular physical user. However, as you will see later in this chapter, Fedora Core or RHEL systems also have *logical user accounts*, user accounts that exist for particular applications, such as MySQL, or system functions, such as the mail and bin user accounts.

Other than this distinction between real and logical user accounts, there are few substantive differences between actual and logical users. In particular, both actual and logical have user identification numbers (UIDs), numeric values that

the kernel and many applications use instead of the account name. Ordinarily, each user account has a unique UID (on a given system), but this is not strictly required.

Because properly managing user and group accounts and assigning and revoking user and group permissions is so important on any Fedora Core or RHEL system, this chapter spends a good deal of time examining the command line and graphical tools for doing so.

The commands covered in this section concern user and group management. Commands used for working with users and groups fall into three broadly defined categories: creating, modifying, and deleting user accounts; creating, modifying, and deleting group accounts; and displaying current and historical login and usage information.

Working with User Accounts

One of the most common administrative tasks is working with user and group accounts. Although some administrators find the traditional command line tools for managing users and groups tedious or inconvenient to use, this chapter examines them in detail. For those readers who prefer GUI tools, the section titled “Using the User Manager” covers the User Manager tool, a GUI application for creating, modifying, and deleting both users and groups. Table 29-1 lists the commands for adding, modifying, and deleting user accounts. They are discussed in detail in the following subsections.

You use the following commands most often:

- **useradd** — Create user login accounts
- **userdel** — Delete user login accounts
- **usermod** — Modify user login accounts
- **passwd** — Set or change account passwords
- **chsh** — Set or change a user’s default shell
- **chage** — Modify password expiration information

The User Database Files

To understand the following discussion, you need to know the format of the user database files, `/etc/passwd` and `/etc/shadow`. Each line in both files consists of colon-separated fields, one line per user. The format of the password file, `/etc/passwd`, is:

```
username:password:uid:gid:gecos:directory:shell
```

Table 29-1 describes the fields in `/etc/passwd`.

Table 29-1 Fields in the Password File

FIELD	DESCRIPTION
Username	The user's account name on the system
password	username's encrypted password or an x
uid	username's numeric UID (user ID)
gid	username's numeric primary group ID (group ID)
gecos	An optional field used for informational purposes that usually contains username's full name
home	username's home directory
shell	username's login shell

On Fedora Core and RHEL systems (and almost any Linux systems these days), the actual password is stored in `/etc/shadow`, indicated by an `x` in the password field in `/etc/passwd`. Because `/etc/passwd` is readable by all users, storing even encrypted passwords in it makes password guessing easier. `/etc/shadow` is more secure because it is readable only by programs that run with root privileges, such as `login` and `passwd`. The sidebar “The Shadow Password System” discusses shadow passwords in more detail.

Strictly speaking, the `shell` field in `/etc/passwd` identifies the program or command to run when a user logs in. If it is empty, `/bin/sh` is used. If it is set to a nonexistent executable or `/bin/false`, the user is unable to log in. In fact, you can use any command, program, or shell script in the shell field. For example, you might want to create a shell script that sends email to the security administrator if someone attempts to log in using a certain account. As long as the script ends with a call to `/bin/false` (or `/sbin/nologin`), the user won't be able to log in.

An entry in `/etc/passwd` should resemble the following:

```
marysue:x:502:502:Mary Sue:/home/marysue:/bin/bash
```

In this entry, username is `marysue`; password is `x`, meaning it is stored in `/etc/shadow`; uid is 502; gid is 502; `gecos` is `Mary Sue`; home is `/home/marysue`; and shell is `/bin/bash`.

NOTE The `gecos` field is primarily of historical interest. **GECOS** is an acronym meaning **General Electric Comprehensive Operating System** and was renamed to **GCOS** when Honeywell purchased General Electric's large systems division. Dennis Ritchie, one of the creators of UNIX, writes of it: “Sometimes we sent printer output or batch jobs to the GCOS machine. The `gc` field in the password file was a place to stash the information for the \$IDENT card. Not elegant.”

THE SHADOW PASSWORD SYSTEM

Red Hat Linux, like most, if not all, Linux and UNIX systems, uses shadow passwords because they offer enhanced protection for your system's authentication files. During the installation of Red Hat, shadow password protection for your system is enabled by default, as are MD5 passwords, which are an alternative and far more secure method of encrypting passwords because they are longer and use a encryption method stronger than the standard DES encryption used by the standard authentication utilities.

Shadow passwords offer a number of distinct advantages over the traditional password system, including:

- ◆ Improved system security by moving the encrypted passwords (normally found in `/etc/passwd`) to `/etc/shadow`, which is readable only by root
- ◆ Information concerning *password aging*, how long it has been since a password was last changed
- ◆ Control over how long a password can remain unchanged before the user is required to change it
- ◆ Settings in `/etc/login.defs`, particularly concerning passwords to give you the ability to set and enforce a security policy

The shadow password suite contains a number of utilities that simplify working with shadow passwords and, if you wish, also simplify reverting to traditional password management. These utilities include:

- ◆ `pwconv` and `pwunconv` for switching from normal to shadow passwords and back, respectively
- ◆ `pwck` and `grpck` for verifying the contents and consistency of the password and group files, respectively, against their shadowed counterparts
- ◆ `useradd`, `usermod`, and `userdel`, for adding, deleting, and modifying user accounts
- ◆ `groupadd`, `groupmod`, and `groupdel`, for adding, deleting, and modifying group accounts
- ◆ `gpasswd`, for administering the groups file `/etc/group`

In addition to storing the encrypted password, `/etc/shadow` stores password expiration information. As in `/etc/passwd`, fields are separated by colons. It contains the following fields, listed in the order in which they appear in `/etc/shadow`:

- The account name
- The account's encrypted password
- The number of days since 1 January 1970 that the password was last changed

- The number of days permitted before the password can be changed
- The number of days after which the password must be changed
- The number of days before the password expires that the user is warned the account will expire
- The number of days after the password expires before the account is disabled
- The number of days since 1 January 1970 after which the account is disabled
- Reserved for future use

The entry from `/etc/shadow` that corresponds to the entry from `/etc/passwd` shown earlier is:

```
marysue:$1$TJp8osKa$GDUrFMa9Twac/1sjV3mA90:12877:0:99999:7:::
```

The `useradd` command creates new user accounts and, when invoked with the `-D` option, modifies the default values applied to new accounts. As a result, it can be invoked in two ways. The syntax of the first form is:

```
useradd [-c comment] [-d dir] [-e date]
        [-f time] [-g initial] [-G group[,...]]
        [-m [-k dir] | -M]
        [-p passwd] [-s shell] [-u uid [-o]]
        [-n] [-r] username
```

The first form creates a new user account named `username`. Optional values not specified using options are assigned default values drawn from `/etc/login.defs` and `/etc/default/useradd`. Table 29-2 lists the options `useradd` accepts.

Table 29-2 `useradd` Options and Arguments

OPTION	DESCRIPTION
<code>-c <i>comment</i></code>	Uses <i>comment</i> for the name field
<code>-d <i>dir</i></code>	Names the new user's home directory <i>dir</i>
<code>-e <i>date</i></code>	Sets the account's expiration date to <i>date</i>
<code>-f <i>time</i></code>	Disables the account <i>time</i> days after the password expires
<code>-g <i>group</i></code>	Sets the user's primary group membership, or login group, to <i>group</i>
<code>-G [<i>group</i>[,...]]</code>	Makes the user a member of each supplemental group <i>group</i>

(continued)

Table 29-2 (continued)

OPTION	DESCRIPTION
<code>-m</code>	Creates the home directory if it does not exist and copies the files and directory structure in <code>/etc/skel</code> to the new directory
<code>-k dir</code>	Copies the files and directory structure in <code>dir</code> , not <code>/etc/skel</code> , to the new home directory; must be specified with <code>-m</code>
<code>-M</code>	Disables creation of the home directory; cannot specify <code>-m</code> and <code>-M</code>
<code>-p passwd</code>	Sets the account password to the encrypted password <code>passwd</code>
<code>-s shell</code>	Sets the user's default shell to <code>shell</code>
<code>-u uid</code>	Sets the user's UID (User ID) to <code>uid</code> , which must be a unique number
<code>-o</code>	Allows the UID specified with <code>-u uid</code> not to be unique; must be specified with <code>-u</code>
<code>-n</code>	Disables use of Red Hat's user private groups
<code>-r</code>	Creates a system account (an account with a UID less than 100) but does not create a home directory
<code>username</code>	Sets the login name to <code>username</code>

TIP The `adduser` program is a symbolic link to `useradd`.

The second way to invoke `useradd` uses the `-D` option. Invoked with only `-D`, `useradd` displays its current default settings. Using `-D` with any of the options listed in Table 29-3 modifies the default value for the corresponding field. Here is the syntax for the second form:

```
useradd -D [-g group] [-b home_dir]
          [-f inactive_time] [-e expire_date]
          [-s shell]
```

`useradd`'s default values are stored in `/etc/default/useradd`.

Table 29-3 Options for Changing `useradd` Defaults

OPTION	DESCRIPTION
<code>-g group</code>	Sets the default group to <code>group</code>
<code>-b dir</code>	Sets the default home directory to <code>dir</code>

Table 29-3 (continued)

OPTION	DESCRIPTION
<code>-f time</code>	Sets the default account disable time to <code>time</code> days
<code>-e date</code>	Sets the default account expiration date to <code>date</code>
<code>-s shell</code>	Sets the default login shell to <code>shell</code>

The `userdel` command deletes a user account and, optionally, related files. Its syntax is:

```
userdel [-r] username
```

`username` identifies the user account to delete. Using `-r` deletes the corresponding home directory and mail spool. Without `-r`, `userdel` removes only the account references in the user and group database files. You cannot delete the account of a logged in user, so `userdel` fails if `username` is logged in.

The `usermod` command modifies an existing user account. Its syntax is:

```
usermod [-c comment] [-d dir [-m]] [-e date]
        [-f inactive] [-g group] [-G group[,...]]
        [-l new_username] [-p passwd]
        [-s shell] [-u uid [-o]] [-L|-U] username
```

`usermod` accepts the options and arguments listed in Table 28-1 for `useradd` and adds three new ones, `-l new_username`, `-L` and `-U`. `-l new_username` changes the account name from `username` to `new_username`. `-L` disables (locks) `username`'s account by placing a `!` in front of the user's encrypted password in `/etc/shadow`. `-U` enables (unlocks) the account by removing the `!`. At least one option must be specified, but `-p`, `-U`, and `-L` may not be used together in any combination. If `username` is logged in, `usermod` fails because you cannot change the login name of a logged-in user.

The `passwd` command, generally regarded as "the password changing utility," actually has more capabilities than merely changing passwords. In general, it updates all of a user's authentication tokens, of which the login password is only one. Its syntax is:

```
passwd [-dkluf] [-S] username
```

`-d` removes the password for `username`, disabling the account. `-k` causes `passwd` to update only expired authentication tokens (passwords, in this case). `-l` or `-u` lock or unlock, respectively, `username`'s password by placing and removing a `!` in front of `username`'s password in `/etc/shadow`. Finally, the `-S` option displays a short status message about `username`, indicating whether the account is locked or unlocked, the kind of encryption used, and so forth.

Another very handy method to set or change a password, especially from a script, is to use the `passwd` command's `--stdin` option, which allows you to pipe a new, plain-text password in. For example, the following command changes the user `bubba`'s password to `sekritword`, using the `--stdin` option to `passwd`:

```
# echo 'sekritword' | passwd --stdin bubba
Changing password for user bubba
passwd: all authentication tokens updated successfully.
```

This command pipes the password through the `passwd` command. When used with the `--stdin` option, `passwd` reads its input from `stdin` rather than interactively at the keyboard. Notice that the `echo` command embeds the password in single quotes because the password contains shell metacharacters that must be protected from expansion.

The `chsh` command changes a user's login shell. Its syntax is:

```
chsh [-s shell ] [-l] [username]
```

`-s shell` sets `username`'s login shell to `shell`. Unless configured otherwise, `shell` can be the full pathname of any executable file on the system. One common way to take advantage of this feature is to disable an account by setting `shell` to `/bin/false` or another command that does not display a login prompt to the user. Using the `-l` option displays the shells listed in `/etc/shells`.

The `chage` command changes the expiration policy for a user's password. Its syntax is:

```
chage [-l] [-m mindays] [-M maxdays] [-d lastday] [-I inactive]
      [-E expiredate] [-W warndays] username
```

Table 29-4 lists the valid options that `chage` accepts.

Table 29-4 Options for the `chage` Command

OPTION	DESCRIPTION
<code>username</code>	Specifies <code>username</code> as the account name to query or modify.
<code>-l</code>	Displays expiration information for <code>username</code> .
<code>-m mindays</code>	Sets <code>mindays</code> days as the minimum amount of time permitted between password changes.
<code>-M maxdays</code>	Sets <code>maxdays</code> days as the maximum number of days a password is valid.

Table 29-4 (continued)

OPTION	DESCRIPTION
-d lastday	Sets <code>lastday</code> as the date on which the password was last changed, expressed as the number of days elapsed since 1 January 1970. <code>lastday</code> can be set using a more convenient date format, such as June 21, 2003, or 2003-0621.
-I inactive	Sets <code>inactive</code> days as the number of days <code>username</code> 's account may be inactive after the password has expired before the account is locked.
-E expiredate	Sets <code>expiredate</code> as the date on which <code>username</code> 's account expires.
-W warndays	Sets <code>warndays</code> as the number of days before the password expires that a warning message is issued.

If no options are used, `chage` executes in interactive mode, prompting the user for each item of information. The `chage` command's `-l` option to obtain a friendlier display:

```
# chage -l marysue
Minimum:          0
Maximum:          99999
Warning:          7
Inactive:         -1
Last Change:      Apr 04, 2005
Password Expires: Never
Password Inactive: Never
Account Expires:  Never
```

`chage` does not display the fields in the order in which they appear in `/etc/shadow`.

Modifying Multiple Accounts Simultaneously

In busy or large IT environments, system administrators often find themselves faced with the necessity of creating multiple user accounts. Using `useradd` to add one or two accounts is relatively simple, but it quickly becomes tedious if 10 or 20 accounts need to be created. Fortunately, the shadow password suite includes the `newusers` utility, which can be used to create and update multiple user accounts. As remarked on at the beginning of the chapter, one of the advantages of command line tools is that they can be used to perform bulk or mass changes. Two commands, `chpasswd` and `newusers`, make multiple changes to the user password database in a single operation. The syntax is:

```
newusers userfile
```

`userfile` is the name of a text file consisting of lines in the same format as the standard password file, subject to the following exceptions:

- The password field appears as clear text — `newusers` encrypts it before adding the account.
- The `pw_age` field is ignored for shadow passwords if the user already exists.
- The GID can be the name of an existing group or a nonexistent GID. If the GID is the name of an existing group, the named user is added to that group, but if it is a nonexistent numeric value, a new group with the specified GID is created.
- If the specified home directory refers to a nonexistent directory, `newusers` creates it. If the directory already exists, ownership of the directory is set to that of the named user.

The following code shows the contents of `newusers.txt`, which is passed to `newusers` to create three new user accounts, `bubba`, `joebob`, and `marysue`:

```
bubba:mypass:901:901:Bubba User:/home/bubba:/bin/bash
joebob:yourpass:902:902:Joe Bob:/home/joebob:/bin/bash
marysue:somepass:903:903:Mary Sue:/home/marysue:/bin/bash
```

After executing the command `newusers newusers.txt`, you will see the entries in `/etc/passwd`, `/etc/group`, and `/etc/shadow`, as shown in Listing 29-1.

```
# tail -3 /etc/passwd
bubba:x:901:901:Bubba User:/home/bubba:/bin/bash
joebob:x:902:902:Joe Bob:/home/joebob:/bin/bash
marysue:x:903:903:Mary Sue:/home/marysue:/bin/bash
# tail -3 /etc/group
901:x:901:bubba
902:x:902:joebob
903:x:903:marysue
# tail -3 /etc/shadow
bubba:jYNrf8iU4DM:12895:0:99999:7:::
joebob:b.hw8uEMl6eNM:12895:0:99999:7:::
marysue:R1ER36oNXeUaA:12895:0:99999:7:::
```

Listing 29-1 Entries in user database files after using `newusers`.

The `chpasswd` command updates existing user passwords en masse. It reads a file consisting of colon-separated `username:password` pairs. `password` must be plain text, which will be encrypted at runtime, unless `chpasswd` is

invoked with the `-e` option, in which case `password` must already be encrypted using a `crypt(3)`-compatible encryption algorithm.

TIP Type `man 3 crypt` to learn more about how the password is encrypted.

Viewing Login and Process Information

To view current and past login information and to determine what processes users are running, you can use one of the following commands:

- **last** — Displays historical login information
- **who** — Displays information about currently logged in users
- **w** — Displays a user's currently running process

For all logins, `last` prints the user name, TTY, date, time, elapsed time, and the host name or IP address of the remote host, if applicable, from which the login originated of all user logins, starting with the most recent login. Its syntax is:

```
last [-R | [-ai]] [-num | -n num] [username] [tty]
```

By default, `last` lists all the entries in `/var/log/wtmp`, so you can use `-num` and `-n num` to specify the number of output lines to display. Ordinarily, `last` displays the hostname in the third column, but using `-a` places the hostname in the rightmost column, `-i` shows the hostname's IP address, and `-R` completely suppresses display of the hostname. To view the login activity of a specific user, use the `username` argument. `tty` enables you to view logins per TTY. Multiple usernames and ttys can be listed.

The `who` command displays information about currently logged-in users. Its default output includes the user name, login TTY, and the date and time each user logged in. `who`'s syntax is:

```
who [-Hil] | [-q]
```

Using the `-H` option adds column headings to `who`'s output. Specifying `-i` adds each user's idle time to the display. Use `-l` to force `who` to show fully qualified domain names (FQDNs). To obtain the total number of logged-in users, use the `-q` option by itself.

The `w` command is very similar to `who`, except that it also displays the command line of each user's currently running process and a summary of each user's CPU usage. `w`'s syntax is:

```
w [-husf] [username]
```

By default, `w` prints header information when it starts; `-h` disables the header. `-s` generates a short output format that omits the login time and the CPU usage. `-f` disables displaying the host from which users are logged in. Specifying `username` lists only `username`'s login session and process information.

Working with Group Accounts

Unlike user accounts, group accounts always represent some sort of logical organization of users. Like user accounts, groups have group identification numbers, or GIDs, and it is common for users to be members of several groups. Groups are used to tie one or more users together to simplify administrative tasks. For example, an administrator can assign a group permission to execute a certain application, and then add and delete users from that group, rather than granting permission to individual users. Handling access control at the group level is a simpler, less labor-intensive approach. Similarly, file access can be controlled at the group level because files are assigned user and group owners when files are created and because files carry separate read, write, and execute permissions for the owner, the group assigned to the file, and any other users.

In large part, the group account administration commands parallel the interface of user administration commands with similar names, except that the group commands have fewer command line options. As the section “Understanding User Private Groups” later in the chapter suggests, Red Hat Linux makes greater use of group accounts than other Linux distributions do. So, knowing how to add, modify, and delete group accounts is more important on Red Hat systems than it is with other Linux distributions.

Table 29-5 lists the commands used to add, modify, and delete group accounts. They are discussed in greater detail in the following subsections.

As with the discussion of the password file in the previous section, you will find the following discussion of working with group accounts less confusing if you understand the format of the group file, `/etc/group`. It has one entry per line, and each line has the format:

groupname:password:gid:userlist

- `groupname` is the name of the group
- `password` is an optional field containing the encrypted group password
- `gid` is the numeric group ID number
- `userlist` is a comma-separated list of the user account names that compose the group

Table 29-5 Group Account Administrative Commands

COMMAND	DESCRIPTION
<code>gpasswd</code>	Sets group passwords and modifies group accounts
<code>groupadd</code>	Creates a new group account
<code>groupdel</code>	Deletes an existing group account
<code>groupmod</code>	Modifies existing group accounts

If `x` appears in the password field, nonmembers of the group cannot join it using the `newgrp` command. A typical entry in the group file might resemble the following:

```
admins:x:507:joebob,marysue,bubba
```

`groupname` is `admins`; password is empty, meaning no group password has been set; `gid` is 503; and `userlist` is `joebob,marysue,bubba`.

Creating Groups

To create a new group, use the `groupadd` command. Its syntax is:

```
groupadd [[-g gid [-o]] [-r] [-f] groupname
```

`groupname` is the only required argument and must be the name of a nonexistent group. When invoked with only the name of the new group, `groupadd` creates the group and assigns it the first unused GID that is both greater than 500 and not already in use. Specify `-f` to force `groupadd` to accept an existing `groupname`. Use the `-g gid` option if you want to specify the new group's GID, replacing `gid` with a unique GID (use the `-o` option to force `groupadd` to accept a nonunique GID). To create system group, one that has special privileges, use the `-r` option.

The following command creates a new group named `admins`:

```
# groupadd admins
```

Here is the resulting entry created in `/etc/group`:

```
admins:x:507:
```

As this point, `admins` has no members and the password field has an `x` in it, meaning that no one (which is everyone at this point) except root can join the group using `newgrp`.

Modifying and Deleting Groups

After creating a new group, you will likely want to add user accounts to it. Two commands modify group accounts, each serving different purposes. `groupmod` enables you to change a group's GID or name, and `gpasswd` enables you to set and modify a group's authentication and membership information. You should rarely need to change a group's name or GID; you're on your own to read the `groupmod`'s short manual page. We're more interested in `gpasswd`, which enables the root user to administer all aspects of a group account and to delegate some administrative responsibilities to a group administrator. For simplicity's sake, the following discussion explains the uses of `gpasswd` *only* available to root. Then it covers the `gpasswd` calls a group administrator can perform.

From root's perspective, `gpasswd`'s syntax is:

```
gpasswd [-A username] [-M username] groupname
```

Root can use `-A username` to assign `username` as `groupname`'s group administrator. `-M username` adds `username` to `groupname`'s membership roster. Assigning a group administrator using `-A` does not make the administrator a member of the group; you have to use `-M` to add the administrator as a member of the group. Multiple `username`'s can be specified with `-A` and `-M`. The following command shows how to add `marysue` and `joebob` to the `admins` group:

```
# gpasswd -M marysue,joebob admins
```

NOTE To use the `-A` option, the shadow group file, `/etc/gshadow` must exist. Read the subsection titled “Using a Shadowed Group File” to understand the implications of using shadowed group files.

After this change, the `admins` entries in `/etc/group` should resemble the following:

```
admins:!:507:marysue,joebob
```

Notice that adding users to the `admins` group account replaced `x` with `!` in the password field, meaning that password-based access to the group (using `newgrp`) is disabled.

For group administrators, `gpasswd`'s syntax is:

```
gpasswd [-R] [-r] [-a username] [-d username] groupname
```

`gpasswd` called with only `groupname` changes the group password. Once a group password is set, group members can still use `newgrp` to join the group without a password, but nonmembers of the group must supply the password. For example, the following commands show what happens when the user `bubba` uses `newgrp` to join the `admins` group after `root` sets a group password, which, for the record, is `secret`:

NOTE `newgrp groupname` changes the group identification of the calling user to `groupname`. After calling `newgrp` successfully, file access permissions are calculated based on the new GID. If `groupname` is omitted, the GID is changed to the calling user's primary (login) GID.

```
$ newgrp admins
Password:
$ groups
admins bubba
```

By contrast, here is what happens when `joebob`, who *is* a member of `admins`, uses `newgrp` to join the `admins` group. Notice that `joebob` is not prompted for a password as `bubba` was:

```
$ newgrp admins
$ groups
admins joebob
```

Conversely, if no group password is set, *only* group members can use `newgrp` to join the group. To remove a group password, use the `-r` option. The next snippet shows what happens when `bubba` tries to join `admins` after the group password is removed. Keep in mind that the password field in the group file will be empty after the password is removed using `-r`:

```
$ newgrp admins
newgrp: Permission denied.
```

This time, `bubba` was not even prompted for a password. `joebob`, however, has no problem:

```
$ newgrp admins
$ groups
admins joebob
```

Calling `gpasswd` with the `-R` option disables access to a group using the `newgrp` command. Oddly, if you use this option, `gpasswd` places a `!` in the password field in the group file, so nonmembers of the group get a password prompt but no password works.

To add a user to the group, a group administrator must use the `-a username` option. The `-d username` option removes a user from a group. The next example shows how to add and remove bubba using `gpasswd`'s `-a` and `-d` options:

```
# gpasswd -a bubba admins
Adding user bubba to group admins
# grep admins /etc/group
admins:!:507:marysue,joebob,bubba
# gpasswd -d bubba admins
Removing user bubba from group admins
# grep admins /etc/group
admins:!:507:marysue,joebob
```

Using a Shadowed Group File

Much of the behavior described in the previous subsection does not apply if the shadow group file, `/etc/gshadow`, is present. In particular, if the shadow group file is in use:

- Adding a group creates an entry for that group in the shadow group file that resembles the following:

```
admins:x:507:
admins:!::
```

- Adding a user to a group adds that user to both the standard group file and the shadow group file:

```
# gpasswd -M marysue admins
# grep admins /etc/group /etc/gshadow
group:admins:x:507:marysue
gshadow:admins:!:marysue
```

- The third field in the shadow group file holds the name of the group administrator, not the GID, if an administrator is added using `gpasswd`'s `-A username` option:

```
# gpasswd -A marysue admins
# grep admins /etc/gshadow
admins:!:marysue:marysue
```

- A group administrator cannot join the group unless the administrator's account is also a member of the group. Similarly, a group administrator can add and delete her user account from the group without affecting her administrative function.

- Only group members can use `newgrp` to join the group. To put it another way, nonmembers of a group cannot use `newgrp` to join groups of which they are not members, even if they know the group password. In fact, passwords are irrelevant because they do not work for non-members and members do not need to use them.

Deleting a group is quite simple. Use the `groupdel` command, which takes no options except the name of the group to delete. For example, the following command deletes the `admins` group:

```
# groupdel admins
```

NOTE Those of you who find typing commands tedious, the next section, “Administering Users and Groups with User Manager,” shows you how to use User Manager, Red Hat’s new GUI tool for administering user and group accounts.

Using User Private Groups

You need to understand the user private group (UPG) scheme and how the UPG scheme uses the semantics of set-GID directories. The UPG scheme as adopted in early Red Hat Linux distributions and carried forward into Fedora Core and RHEL. UPGs are intended to make Linux groups easier to use. Although the UPG scheme does not add or change the normal Linux way of handling groups, it *does* introduce a new convention that is different from traditional Linux user and group idioms: when you create a new user, Fedora Core and RHEL create a unique group for that user. Although it is unusual and a departure from traditional norms, after you become accustomed to the UPG scheme, you will find that it is very natural to use and makes good sense.

The UPG scheme has the following salient characteristics:

- Each user has a primary group with the same name as the user account. For example, the user named `bubba` has a primary or initial group of `bubba`.
- Each user is the only member of her primary group. Thus, the user `bubba` is the only member of the group `bubba`.
- Each user’s `umask` defaults to `002`; because every user has her own private group in the UPG scheme, the group protection afforded by the normal Linux `umask` of `022` is unnecessary.

- Group-specific directories, such as project directories, have the set-GID (set group ID) bit enabled. If you set the set-GID bit on a directory, all files created in that directory have their group set to the directory's group. The behavior of set-GID directories is not specific to UPGs, but the UPG scheme does take advantage of set-GID features.

TIP The default umask is set in `/etc/profile`.

For example, suppose that the finance department maintains a large number of files in the `/opt/finance` directory and that many people work with these files on a daily basis. If you want to use set-GID directories and the UPG scheme, you first create a group named, say, `finance`, use the `chgrp` command to change the group ownership on `/opt/finance` to the `finance` group, use the `chmod` command to set the set-GID bit on `/opt/finance`, and then add the appropriate users to the `finance` group. As a result, all users in the `finance` group can edit existing files in the `/opt/finance` directory. Similarly, when new files are created in the `/opt/finance` directory, the files' group ownerships are automatically assigned to the `finance` group, and all users who are members of the `finance` group can edit them without taking any special steps.

Another benefit of set-GID directories is that any users who work on multiple projects do not have to change their umask or group as they move from project to project or directory to directory. Each project directory's set-GID bit automatically sets the proper group for all files created in that directory and its subdirectories.

At the user level, each user's home directory is owned by the user and her private group. Although it is safe to set the set-GID bit on the home directory, it is unnecessary. Why? By default, files are created with the primary group of the user and that user is the only member of the primary group. Thus, the set-GID bit is redundant with respect to a user's home directory and its subdirectories.

The following steps illustrate the scenario and process just described. The point of this exercise is to provide a concrete illustration of Red Hat's UPG scheme, so a later section discusses the commands and options used.

1. Create the `finance` group:

```
# /usr/sbin/groupadd finance
```

2. Change the group ownership of `/opt/finance` to the `finance` group to associate the directory contents with the `finance` group:

```
# /bin/chgrp -R finance /opt/finance
```

3. Add the proper users to the group (add the user `bubba` in this case):

```
# /usr/bin/gpasswd -a bubba finance
```

```
Adding user bubba to group finance
```

4. To enable the `finance` group's members to create, make the directory writable by the group:

```
# /bin/chmod g+w /opt/finance
```

5. Set the set-GID bit on `/opt/finance` to cause newly created files in the `/opt/finance` tree to have `finance` group ownership:

```
# /bin/chmod g+s /opt/finance
```

After this command, the letter `s` appears where the group execute bit (denoted by the letter `x`) would ordinarily appear when you generate a long listing of `/opt/finance`'s attributes:

```
$ ls -ld /opt/finance
```

```
drwxrwsr-x 2 root finance 6 Apr 20 19:26 /opt/finance
```

With the default umask set to `002`, files that `bubba` creates in `/opt/finance` are owned by the user `bubba` and the group `finance` and are read/write for both the user and group owner, enabling other `finance` users to modify the file:

```
$ touch /opt/finance/20050420
```

```
$ ls -l /opt/finance/20050420
```

```
-rw-rw-r-- 1 bubba finance 0 Apr 20 19:29 /opt/finance/20050420
```

To summarize, the set-GID bit on directories, combined with the Red Hat UPG scheme, makes it trivial to create project groups that permit members of those groups to write files in the groups' common directory without unduly burdening users.

Administering Users and Groups with User Manager

User Manager is a graphical tool for administering user and group accounts. To use it, you must be logged in as root or otherwise have root access. To start User Manager, click Main Menu ⇨ System Settings ⇨ Users and Groups. You can start from a command line using the command `system-config-users` in a terminal window. The initial screen resembles Figure 29-1.

From this screen you can view, modify, and delete existing user and group accounts or create new ones. To reduce the list of displayed accounts or to search for a specific account, type the first few letters of an account name in the Filter by text box and click the Apply filter button. You can update most windows by clicking the Refresh button on the toolbar. To get context-sensitive help, click the toolbar's Help button or, to view the entire User Manager manual, select Help ⇨ Manual from the toolbar.

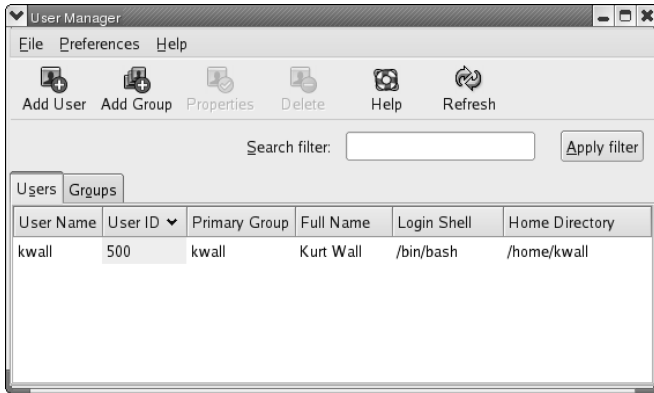


Figure 29-1 The main Red Hat User Manager dialog box.

Creating User Accounts

To add a new user:

1. Click the Add User button. The Create New User dialog box, shown in Figure 29-2, appears.
2. Type the new account name in the User Name text box.
3. Type the user's full name in the Full Name text box.

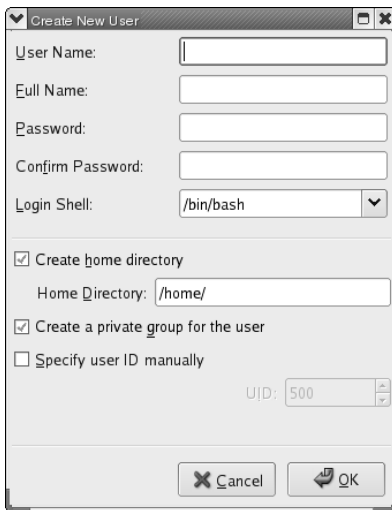


Figure 29-2 Adding a new user.

4. Type the user's password in the Password and Confirm Password fields. The password must be at least six characters.
5. Select a login shell. If you choose not to accept the default shell, select an alternative shell from the Login Shell drop-down box.
6. As noted earlier in this chapter, the default home directory is `/home/username`. You can change the home directory by editing the Home Directory text box or not create a home directory at all by clearing the Create home directory check box.
7. To prevent creation of a user private group, remove the check from the Create new group for the user check box. A completed Create New User dialog box might resemble Figure 29-3.
8. Click OK to create the user.

Modifying and Deleting User Accounts

After you have created a user account, you can configure additional properties by clicking User Manager's User tab, selecting the user, and clicking the Properties button to open the User Properties dialog box. To add the user to additional groups, click the Groups tab (see Figure 29-4). Click the check box next to the groups of which the user should be a member, then click the Apply button.

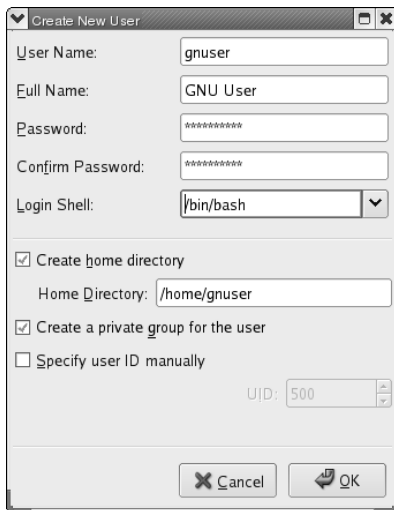


Figure 29-3 A newly created user account in User Manager.

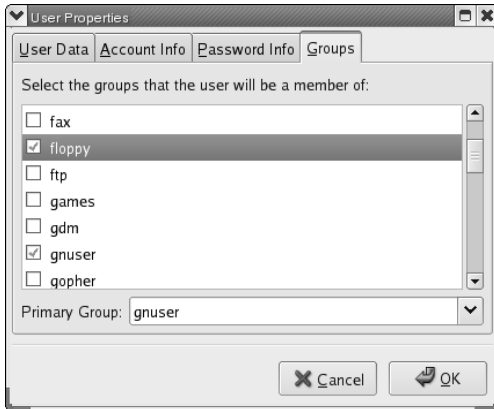


Figure 29-4 Adding a user to additional groups.

Other account data you can modify from the User Properties window includes the basic user information you supplied when you created the user (the User Data tab), account information (the Account Info tab), and password expiration information (the Password Info tab). On the Password Info tab, click the Enable account expiration check box to set the user account's expiration date if you want the account to expire on a certain date. To prevent this user account from logging in, place a check mark in the User account is locked check box.

Click the Password Info tab to view and change the account password expiration information. (See Figure 29-5.) The date that the user last changed her password appears across the top of the tab. Click Enable password expiration to force a password change after a certain number of days, and then enter the number of days between required password changes in the Days before change required text box. You can also set the number of days before the user can change her password, the number of days before the user is warned to change her password, and the number of days before the account becomes inactive. When you have finished modifying the user account properties, click OK to apply the changes and close the User Properties dialog box.

Finally, to delete a user account, click the account to delete on User Manager's Users tab, and then click the Delete button.

Creating Group Accounts

To add a new user group, click the Add Group button. In the Create New Group dialog box, shown in Figure 29-6, type the name of the new group, and then click OK to create the group.

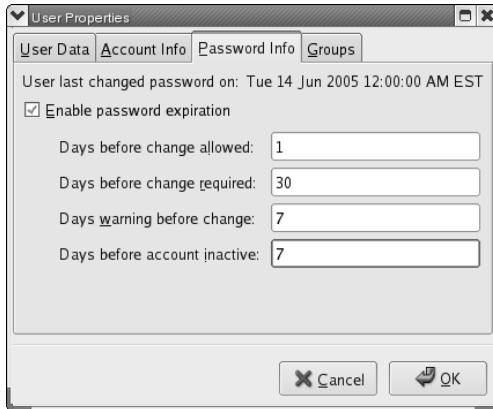


Figure 29-5 Modifying user account password expiration information.



Figure 29-6 Adding a new group.

Modifying and Deleting Group Accounts

To view or modify the properties of an existing group, select the group to modify from the group list on the Groups tab and click the Properties button. The Group Properties dialog box, shown in Figure 29-7, appears.



Figure 29-7 Modifying group properties.

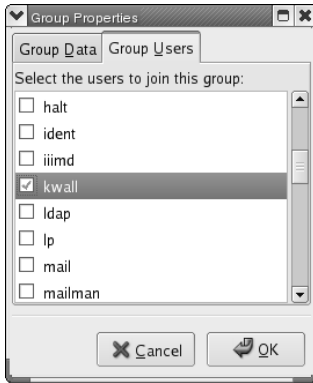


Figure 29-8 Modifying group properties.

The Group Users tab, shown in Figure 29-8, displays the users that are members of the group. To add other users to the group, place a check mark next to the user account names in the list, and deselect account names to remove them from the group. Click OK to apply the changes and close the Group Properties box.

After you have finished adding or modifying user and group accounts, click File ⇨ Quit or press Ctrl+Q to save your changes and close User Manager.

Understanding the Root Account

With very few and limited exceptions, the root account has unlimited power on any Linux or UNIX system, and, in this respect, Red Hat Linux is no exception. The root account, or, to use the expression you see throughout this chapter, *root*, can access any file and modify any process. Indeed, it is for this reason that root is often called the *superuser* — root is effectively omnipotent.

The exceptions to root's capabilities are few. As explained in Chapter 12, root on an NFS client (that is, a system mounting an NFS exported file system from an NFS server) typically *cannot* exercise root privileges on the exported file system because the NFS server exports the file system using the `root_squash` option. As you should recall, the `root_squash` option maps root's UID and GID to the UID and GID of the anonymous user on the client system. Keep in mind that this is the default behavior of the NFS server daemon (`exports`) and can be overridden using `no_root_squash` on the NFS server. This is one limit on root's power. On the local system and for local resources, root always has unlimited power and control. It is only in the case of remote or networked resources where root's power is subject to special considerations and restrictions on root's power emerge.

The ext2 and ext3 file systems also restrict root's power, although only slightly. The ext2 and ext3 file systems support a number of special file attributes, including immutability. Using the `chattr` utility, root can set a file's

immutable attribute, which prevents *all* users, including root, from modifying the file; it cannot be deleted, renamed, or written to, and hard links cannot be created to it until the immutable attribute is cleared. You guessed it — only root can set or clear a file's immutable attribute.

Linux capabilities and Linux access control lists (ACLs) are being developed that enable root's power with respect to process management and file access to be subdivided into more finely grained capabilities. Based on the IEEE's POSIX capabilities and first appearing in the 2.2 kernel, *capabilities* work by splitting root's traditional privileges into smaller sets of more specific privileges that you can enable and disable. Eventually, POSIX capabilities will also be applied to files in the file system.

The most immediately useful application is what is referred to as a *capability bounding set*, which defines a list of capabilities any process running on a Linux system can hold. If a capability does not appear in the bounding set, no process, regardless of how privileged it is, can exercise it. For example, you can disable kernel module loading by removing this capability from the bounding set. Although manipulating kernel capabilities is an advanced topic beyond this book's scope, you might find it interesting or useful, depending on your environment, to examine the Linux Kernel Capability Bounding Set Editor (LCAP), a tool that takes advantage of and manipulates POSIX capabilities. Additional information about POSIX capabilities is available via FTP from the kernel source code repository at <ftp://ftp.kernel.org/pub/linux/libs/security/linux-privs/>.

The LCAP editor's old Web page, <http://pweb.netcom.com/~spoon/lcap/>, no longer works. Nevertheless, you can still download the editor from the kernel.org FTP site or from the Security Focus Web site, securityfocus.com/tools/lcap-0.0.2.tar.

NOTE Security Enhanced Linux, or SELinux, offers another alternative for reigning in root's absolute power. Chapter 33 introduces SELinux and its features for enhancing system security by more precisely defining user capabilities.

Implementing Sudo

Considering root's privileges, you can easily understand why root access on a Linux system is carefully protected and the root password tightly guarded. Nevertheless, it is often desirable to grant privileges to a nonroot user (humorously referred to as *merely mortal user*) that have traditionally been solely root's domain, such as printer management, user account administration, system backups, or maintaining a particular Internet service. In other operating systems, such users are often called *wheel users* or *administrative users*. Indeed, in

many environments, subdividing system administration responsibilities is a necessity because the responsibilities of maintaining multiple servers in a large IT shop or ISP can quickly overwhelm a single individual. The problem in such a situation is clear: How do you grant administrative privileges to merely mortal users *without* providing unfettered root access? In many situations, Sudo, a mnemonic for *superuser do*, is one solution. Sudo enables you to give specific users or groups of users the ability to run some (or all) commands requiring root privileges. Sudo also logs all commands executed, which allows you to maintain an audit trail of the commands executed, by whom they were executed, when they were executed, and so on. As the README in the source distribution states, Sudo's "basic philosophy is to give as few privileges as possible but still allow people to get their work done." Sudo's features include:

- Enabling the ability to restrict the commands a given user may run on a per-host basis.
- Maintaining a clear audit trail of who did what. The audit trail can use the system logger or Sudo's own log file. In fact, you can use Sudo in lieu of a root shell to take advantage of this logging.
- Limiting root-equivalent activity to a short period of time using time-stamp based "tickets," thus avoiding the potential of leaving an active root shell open in environments where others can physically get to your keyboard.
- Allowing a single configuration file, `/etc/sudoers`, to be used on multiple machines, permitting both centralized Sudo administration and the flexibility to define a user's privileges on a per host basis.

After the configuration file has been created, a typical Sudo session proceeds as follows:

1. An authorized user prefixes the root command she wants to execute with `sudo` followed by a space, for example:

```
$ sudo shutdown -h +5 "System shutting down for disk replacement"
```
2. Sudo prompts the user for her personal password (*not* the root password) and then checks the configuration file (`/etc/sudoers`) to make sure she has permission to run the given command on a given machine.
The password prompt can be overridden by specifying the `NOPASSWD` flag in `/etc/sudoers`, but this poses as security risk, so we don't cover the `NOPASSWD` flag in this section.
3. If the user is permitted to use that command, Sudo runs the command as root (or another user if specified), logs the details, and timestamps the Sudo session ticket.

4. If the user is *not* permitted to use that command, Sudo logs the attempt and exits. Sudo also logs problems and other invalid `sudo` uses.
5. After executing the first command, the user can use multiple `sudo` commands without being prompted for her password again. The session ticket expires five minutes (the default expiration period) after the last `sudo` command is issued, after which the user is again prompted for a password.

By default, `sudo` logs to `/var/log/messages` using the system logger (`syslogd`), but you can configure the system logger to log `sudo`-related messages to a different file. Sudo can even bypass the system logger completely and maintain its own log file. If Sudo is invoked by an invalid user, is invoked with an invalid command, or if other abnormal situations arise, Sudo notifies the root user (by default) via email.

Sudo's configuration file is `/etc/sudoers`, which must be edited using `visudo`, part of the Sudo distribution. Using `visudo` is vital because it locks the `/etc/sudoers` to prevent simultaneous edits and validates the grammar and syntax in the configuration file, cowardly refusing to save changes if it believes it has detected an error.

Deciphering Sudo's Configuration File

Sudo's configuration file, `/etc/sudoers`, is the key file. It contains three types of entries: alias definitions, privilege specifications, and global configuration defaults. *Alias definitions* are variables or placeholders that you can reuse throughout the configuration file. They come in four flavors: user aliases, command aliases, so-called runas aliases, and host aliases. The rationale for aliases is to simplify maintaining the configuration file — rather than editing multiple user or command lists when you update `/etc/sudoers`, you simply modify the appropriate alias and let `sudo` substitute the alias definition in each place where it is used. *Privilege specifications* define which users may execute what commands. *Global configuration defaults* are general settings that control `sudo`'s overall behavior.

Instead of trying to understand `sudoer`'s configuration syntax in the abstract, consider the following example that illustrates typical Sudo usage. Suppose that you want to enable the users `marysue` and `bubba` to reset passwords for all users except `root`, which means that `marysue` and `bubba` need to be able to use the `passwd` command to set passwords for users other than themselves. Somehow, though, they must be prevented from changing the root password. As you know, the command for changing passwords is:

```
passwd username
```

The general procedure is to use `visudo` to edit `/etc/sudoers` and create the following:

- A *user alias* defining the users to whom you are granting access to one or more commands
- A *command alias* that represents the command or commands to execute
- A *host alias* to identify the host or hosts on which the named users are permitted to execute the named command (if necessary)
- A *runas alias* that identifies the user a command should run as (again, if necessary)
- A *user privilege specification* to connect the necessary aliases together to form a Sudo rule

The following procedure shows the specific steps to follow:

1. As the root user, start the edit session by executing `visudo`. Initially, the file should resemble the following:

```
# sudoers file.
#
# This file MUST be edited with the 'visudo' command as root.
#
# See the sudoers man page for the details on how to write a sudoers
file.
#

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL) ALL
```

The hash sign, #, prefixes a comment unless it is used in the context of a username and is followed by one or more digits, for example, #502, in which case `sudo` interprets it as a UID.

2. Add the following line in the user alias section:

```
User_Alias PWADMIN=marysue,bubba
```

This statement defines a user alias named `PWADMIN` consisting of the accounts `marysue` and `bubba`.

3. Add the following line in the command alias section:

```
Cmdnd_Alias PW=/usr/bin/passwd [A-z]*,!/usr/bin/passwd root
```

This statement defines a command alias named `PW` that has two components separated by a comma. Command aliases must include a full path specification and any arguments that you wish to permit to be invoked using `sudo`. The first component, `/usr/bin/passwd [A-z]*`, indicates that `/usr/bin/passwd` may be used with any argument that begins with the letters `A-Z` or `a-z` followed by zero or more characters. The second element uses the `!` character to prevent `/usr/bin/passwd` from being used with an argument of `root`.

The character used as wildcards, permitted in both pathname specifications and command arguments, are:

- `*`: Matches any set of zero or more characters, but not a `/` in a path specification. That is, `/usr/bin/*` matches `/usr/bin/bc`, but not `/usr/bin/filter/filter_innd.pl`
- `?`: Matches any single character.
- `[...]`: Matches any character in the specified range. For example, `[A-Z]` matches all the uppercase characters.
- `[!...]`: Matches any character *not* in the specified range. For example, `[0-9]` matches any non-numeric character.
- `\x`: Escapes the character `x`, including `*`, `?`, `[`, `]`, `(`, `)`, `@`, `!`, `=`, `:`, `,`, and `\`

4. Add the following line in the user specification section:

```
PWADMIN ALL = PW
```

This statement says that the `PWADMIN` users can use the `PW` command on all hosts (the `ALL` keyword). More generally, a user privilege specification takes the form:

```
user_alias host_alias=[(runas_alias)] cmdnd_alias[,...]
```

Note that the `runas` alias is not required and that you can specify multiple command aliases in the same entry if each is separated by a comma. In fact, it was not strictly necessary to create the user and command aliases, as the user privilege specification could have been written as follows:

```
marysue,bubba ALL=/usr/bin/passwd [A-z]*,!/usr/bin/passwd root
```

After these edits, `/etc/sudoers` should resemble the following:

```
# sudoers file.
#
# This file MUST be edited with the 'visudo' command as root.
#
```



```
# See the sudoers man page for the details on how to write a sudoers
file.
#

# Host alias specification

# User alias specification
User_Alias PWADMIN=marysue,bubba

# Cmnd alias specification
Cmnd_Alias PW=/usr/bin/passwd [A-z]*,!/usr/bin/passwd root

# User privilege specification
root    ALL=(ALL) ALL
PWADMIN ALL=PW
```

5. Save the changes and exit visudo.
6. As marysue or bubba, test the configuration to make sure that everything works as intended. The first test confirms that bubba can use passwd:

```
$ sudo passwd gnuuser
```

We trust you have received the usual lecture from the local System Administrator. It usually boils down to these two things:

- #1) Respect the privacy of others.
- #2) Think before you type.

```
Password:
Changing password for user gnuuser
New password:
Retype new password:
passwd: all authentication tokens updated successfully
```

The next test demonstrates that members of the PWADMIN alias *cannot* change the root password:

```
$ sudo passwd root
```

```
Sorry, user marysue is not allowed to execute '/usr/bin/passwd root'
as root on luther.
```

Sudo rules can be fine-tuned using flags and keywords in the configuration file. The source code directory for this chapter contains the `sample.sudoers` file shipped in Sudo's source code archive. The sample file contains many helpfully commented examples that you can readily adapt to suit your own

needs. In addition, the manual pages for `sudo`, `visudo`, and `sudoers` (`man sudo`, `man visudo`, and `man sudoers`) have complete information on the syntax and useful examples of configuring `sudoers` and using `visudo`.

Sudo Configuration and Usage Tips

You should plan for security, because `sudo` can be used to gain further power by obtaining root access. For example, if you enable users to run `less` as root, they could use its shell access command, `!`, to run other commands as root or to view sensitive files, such as `/etc/shadow`. As the `sudoers` manual page cautions: “There is no easy way to prevent a user from gaining a root shell if that user has access to commands allowing shell escapes.” In general, when configuring root command access using `Sudo`, adhere to the least-access principle, granting the minimum necessary privileges to accomplish a given task. You should also exercise caution when using `Sudo` in such a way that lets users edit files because a malicious user can set the `$EDITOR` environment variable to almost any command or program, which creates a serious security risk.

When using `visudo`, consider using its `-s` option, which enables a more rigorous level of syntax and grammar checking. In addition, do *not* give users permission to use `visudo` for two reasons: first, they can use `vi`’s `:!` command to obtain a root shell, and, more importantly, it enables them to edit `/etc/sudoers` and alter its contents to give themselves unfettered root access and to disable logging.

Using File System Quotas

The aphorism “Disk space is cheap” has never been more true than it is today, when 120-GB disk drives are standard equipment on new PCs and a 300-GB disk drive can be purchased for \$150. Unfortunately, just as the much heralded paperless office has resulted in skyrocketing paper consumption, the proliferation of massive disk drives has resulted ever-increasing pressures on disk space. For system administrators, one of the perennial challenges is managing disk space and making sure that no single user takes more than his or her fair share. The final section of this chapter shows you how to use the quota utilities to set, monitor, and enforce file system usage quotas for individual users and for groups of users. After you have performed the initial setup, managing file system usage with the quota suite is a largely automated task that leaves you free to concentrate on more pressing concerns.

The programs you use to set and enforce disk usage quotas include the following:

- **edquota** — Sets, edits, and removes user and group file system quotas
- **quota** — Displays defined quotas and current file system usage
- **quotacheck** — Creates, checks, and repairs file system quota files
- **quotaoff** — Disables file system quotas
- **quotaon** — Enables file system quotas
- **repquota** — Summarizes and reports on quota utilization
- **warnquota** — Checks file system usage and sends email to users who exceed their assigned quotas

Quotas are set on a per-file-system basis, rather than per disk. You also must use disk blocks or disk inodes to set quotas, not the more familiar and more easily understood units of megabytes. Despite these inconveniences, however, the procedure for initializing quotas on file systems is straightforward. Briefly, the steps to follow are:

1. Edit `/etc/fstab` to enable quotas on the desired file systems.
2. Create the quota accounting files on the root directory of each file system for which quotas are enforced.
3. Turn on quotas.
4. Set the desired file system quotas.
5. Review quota utilization regularly and frequently.

In the following sections, you look at each of these steps and the commands to accomplish them.

Enabling Quotas

To enable file system quotas, the first step is to drop the system to single user mode. To do so, press `Ctrl+Alt+F1` to flip over to the first virtual console and then log in as root. After you have logged in, execute the following command to bring the system down to single-user mode:

```
# /sbin/telinit 1
```

The reason you should put the system into single-user mode is to prevent users from logging in and altering files. If users alter files while you are setting up quotas, they might lose data. Next, edit `/etc/fstab` and add the mount options `usrquota` or `grpquota` to the file systems on which enable quotas for users or groups, respectively. You can use both options if you want to enable quotas for both users and groups. For example, the following line from `/etc/fstab` enables user quotas on the `/data` file system:

```
/dev/hdb1      /data      ext3      defaults,usrquota      1 2
```

mount itself ignores the `usrquota` and `grpquota` keywords, but programs in the quota suite expect to see it in `/etc/fstab`.

To activate the changes you made to `/etc/fstab`, execute the `mount` command using the `remount` option to update the kernel's mount table (`/etc/mtab`) with the quota option. For example, to activate quotas on the `/data` file system shown in the example, execute the following command:

```
# mount /home -o remount,rw
```

Not all quota implementations are created equal. Quota usage as described in this section assumes that you are using an `ext3` file system. If you are using another file system, such as `XFS` or `ReiserFS`, you should use the quota semantics described in their manual pages to set up and manage quotas.

Creating the Quota Files

Now that the system is prepared, the next phase of the procedure for setting up quotas is to create the accounting files quota uses to monitor file system usage. There are two such files for each file system on which quotas are used, `aquota.user` if user quotas are enforced and `aquota.group` if group quotas are enforced. The quota accounting files are stored in the root directory of each file system. To create these accounting files, execute the `quotacheck` command, as shown in the following example:

```
# quotacheck -uv /data
quotacheck: Scanning /dev/hdb1 [/data] quotacheck: Cannot stat old user
quota file: No such file or directory
quotacheck: Old group file not found. Usage will not be subtracted.
done
quotacheck: Checked 10 directories and 23 files
quotacheck: Old file not found.
```

`quotacheck` scans the specified file system to determine its current usage and then writes this information into the quota accounting files. Because this is the initial scan, `quotacheck` complains that it cannot find the user and group quota files. Only user quotas are enabled in this example, so only the user quota file, `/data/aquota.user`, is created. `Quotacheck's -u` option causes `quotacheck` to create (or check) only user quotas, and the `-v` option specifies verbose operation. `/data`, as you might guess, tells `quotacheck` which file system to scan. `quotacheck's` most commonly used syntax is:

```
quotacheck [-bcgRuv] -a | filesystem
```

`filesystem` specifies the file system to check. If specified, `-a` instructs `quotacheck` to scan all mount file systems listed in `/etc/mtab` that are not NFS mounts. If `-a` is *not* specified, then `filesystem` must be specified. Table 29-6 explains other `quotacheck` command line options.

Turning on Quotas

After creating the quota accounting files, use the `quotaon` command to turn on quotas. `quotaon`'s invocation is simple

```
quotaon [-guv] -a | filesystem
```

`quotaon`'s options have the same meaning as the corresponding options for `quotacheck` listed in Table 29-6. So, to turn on quotas for the `/home` file system used in this section, execute the following command:

```
# quotaon -v /data
/dev/hdb1 [/data]: user quotas turned on
```

Pretty easy, eh?

Setting and Modifying Quotas

At this point, you are finally ready to configure file system quotas because all of the preliminary setup is now complete. To set quotas, use the quota editor, `edquota`, which has the following syntax:

```
edquota [-ug] -t
edquota [-ug] account
```

Table 29-6 `quotacheck` Options

OPTION	DESCRIPTION
<code>-b</code>	Make backup copies of quota files before overwriting old ones.
<code>-c</code>	Ignore existing quota files.
<code>-g</code>	Only check file system group quotas (see <code>-u</code>).
<code>-R</code>	Used with <code>-a</code> , tells <code>quotacheck</code> not to check the root file system.
<code>-u</code>	Only check file system user quotas, the default behavior if neither <code>-g</code> nor <code>-u</code> is specified.
<code>-v</code>	Operate in verbose mode.

The `-u` and `-g` options have the meanings shown in Table 29-6. The `-t` option in the first form of the command enables you to edit the time limits during which file system usage is permitted to be over quota, that is, to exceed the defined limits. Red Hat configures the default time limit, called a grace period, to seven days. To change this default value, execute the following command:

```
# edquota -u -t
```

By default, `edquota` uses the `vi` editor, so the resulting screen should resemble the following:

```
Grace period before enforcing soft limits for users:
Time units may be: days, hours, minutes, or seconds
Filesystem      Block grace period   Inode grace period
/dev/hdb1        7days                7days
```

To change the time limit, change the text that reads `7days` to another value. You can use time units of seconds, minutes, hours, and days. So, for example, to set a time limit of two weeks, change the line that reads:

```
/dev/hdb1          7days                7days
```

so that it reads:

```
/dev/hdb1          14days               14days
```

After making the changes, save them, and exit `edquota` using the standard `vi` keystrokes (`:wq`).

The second form of the `edquota` command enables you to set the actual file system usage limits. `account` must be the name of a user or group for which you are setting quotas. The following example shows the `edquota` session for editing the user `bubba`'s quotas. To edit user quotas, use `edquota`'s `-u` `username` option, as illustrated in the following example:

```
# edquota -u bubba
Disk quotas for user bubba (uid 500):
Filesystem      blocks    soft    hard    inodes    soft    hard
/dev/hdb1        1188     2000    2025      28         0         0
```

The first column shows the file systems (actually, the partitions) for which `bubba` has quotas; the second column shows the number of blocks `bubba` has used, followed by the soft and hard limits for block usage. The fifth column shows the number of inodes, or file system entries, `bubba` is currently using, followed by the soft and hard limits for inode usage. A *hard limit* is the absolute value beyond which file system usage is forbidden to go; once `bubba` reaches

the hard limit, he will not be permitted to create any more files until he deletes enough files on the specified file system to go below his quota. A *soft limit*, on the other hand, is less restrictive than a hard limit; users (or groups) are permitted to exceed their soft limits for the grace period configured using `edquota's -t` option. After the grace period expires, users must reduce their block or inode usage below the soft limit before they can create any more files on the monitored file system.

How big is a block? It varies from system to system depending on the size of the underlying disk. On this system, a block is 1024K. How can you find out the block size of a file system? One cheesy way that we use is `sfdisk -l` device, replacing device with the disk device in which you're interested. For example:

```
# sfdisk -l /dev/hda

Disk /dev/hda: 38792 cylinders, 16 heads, 63 sectors/track
Units = cylinders of 8225280 bytes, blocks of 1024 bytes, counting from 0
```

Device	Boot	Start	End	#cyls	#blocks	Id	System
/dev/hda1	*	0+	12	13-	104391	83	Linux
/dev/hda2		13	143	131	1052257+	82	Linux swap
/dev/hda3		144	2433	2290	18394425	83	Linux
/dev/hda4		0	-	0	0	0	Empty

Notice that the third line of the output shows the block size with the text blocks of 1024 bytes.

To set bubba's quota, set the soft and hard limits for block usage to a nonzero value. For example, the following entry shows bubba's soft limit set to 2000 blocks and his hard limit set to 2205 blocks:

```
/dev/hdb1      1188      2000      2025      28      0      0
```

After setting the limit, save your changes and exit `edquota` using the standard vi keystrokes (`:wq`).

TIP Do not use inode limits. Each file a user creates requires *at least one* inode, so it is conceivable that someone could create lots of small files (where *small* means less than the block size) and reach their quota long before they've exhausted their disk space quota in terms of actual disk space usage.

Viewing Quota Utilization

Reviewing and monitoring quota utilization is an ongoing process, but also easy to accomplish if you automate the process using a couple of cron jobs.

One cron job should run the `warnquota` utility on a daily basis. The other cron job should run the `repquota` program, again on a daily basis. `warnquota` is a script that sends users a short email message resembling that shown in Listing 29-2 if they are over quota.

```
From: root@localhost
Reply-To: root@localhost
Subject: NOTE: You are exceeding your allocated disk space limits
To: bubba@kurtwerks.com
Cc: root@localhost
Message-Id: <20050423203941.F1B8A712AD55@localhost>
Date: Sat, 23 Apr 2005 16:39:41 -0400 (EDT)

Your disk usage has exceeded the agreed limits on this server
Please delete any unnecessary files on following filesystems:

/data (/dev/hdb1)

      Block limits
Filesystem      used    soft    hard  grace    File limits
/dev/loop0      +-    2168    2000    2025 13days    used  soft  hard  grace
                29      0      0

root@localhost
```

Listing 29-2 Quota exceeded warning message from `warnquota`.

The contact phone numbers, the subject line, and the CC list in `warnquota`'s message can be customized by editing `/etc/warnquota.conf`, an example of which is shown in Listing 29-3.

```
MAIL_CMD        = "/usr/sbin/sendmail -t"
FROM            =
SUBJECT         = NOTE: You are exceeding your allocated disk space limits
CC_TO          = "root@localhost"
SUPPORT        = "root@myhost.com"
PHONE          = "(800) 555-1212 or (800) 867-5309"
MESSAGE         = "Your disk usage has exceeded the agreed limits\
on this server|Please delete any unnecessary files on following filesystems:|
SIGNATURE      = DiskMeister <root@localhost>
```

Listing 29-3 A customized `/etc/warnquota.conf` file.

`warnquota` generates its report by calling the `quota` program to check user quotas. You can use `quota` to check file system quota usage manually. `quota`'s syntax is:

```
quota [-gus] user
```


The `-g` and `-u` options have the meaning shown in Table 29-6. Specifying `-s` tells quota to use more understandable units for displaying disk usage and limits. For example, the following command shows bubba's quota statistics:

```
# quota -s bubba
Disk quotas for user bubba (uid 502):
      Filesystem  blocks    quota   limit   grace   files   quota   limit   grace
      /dev/hdb1    2168*  2000    2025           29      0      0
```

To see a complete list of quota statistics for all users and groups for a file system, use the `repquota` program, which accepts the same options as `quota` but requires a file system argument rather than a user argument. You can use the `-a` option to see a report for all file systems on which quotas are being used. The following command shows a `repquota` report for `/dev/hdb1`, using the `-s` option to display the output in units of megabytes:

```
# repquota -s /dev/hdb1
*** Report for user quotas on device /dev/hdb1
Block grace time: 14days; Inode grace time: 14days

                                Block limits                File limits
User              used    soft   hard  grace    used  soft  hard  grace
-----
root              --    34088      0      0           3     0    0
bubba             +-    2168   2000   2025 13days    29     0    0
```

Summary

This chapter briefly recapped the power of the root account on a Red Hat Enterprise Linux system and showed you how to delegate some of that power to nonroot users using Sudo. You also learned to create and manage user and group accounts using a variety of command line utilities and User Manager graphical administration tool. Finally, you read about how to prevent users and groups from monopolizing disk space and how to monitor system disk space utilization using the quota suite of programs.

Installing and Upgrading Software Packages

IN THIS CHAPTER

- Using the Red Hat Package Manager
- Checking Software Versions
- Obtaining Newer Software
- Installing Software from Source

One of Linux's best qualities is the amount of software available for it. Even novice Linux users can download and install new or updated software with little or no difficulty using RPM, the Red Hat Package Manager. Blindly installing software, even on an RPM-based system, may cause problems, though, and in general the more you understand about software installation and maintenance, the more comfortable you will be with the process. This chapter describes how to use RPM and to keep Fedora Core and RHEL-based systems current vis-à-vis updated packages and security patches. It also describes how to use the current version of RPM to add, remove, upgrade, and query software packages. Finally, it briefly describes the process of configuring and installing software from source code.

Using the Red Hat Package Manager

RPM is a powerful software configuration manager and the preferred tool for installing, removing, verifying, and updating software packages on Fedora Core and RHEL systems. This section describes how to use most RPM features. Later sections cover some topics in detail, though, so they are mentioned only briefly in this one. Most examples illustrate command line usage, but some show how to use Package Manager, a graphical RPM front-end application.

RPM consists of two components: a set of databases that store information about installed software and the programs that interface with the databases. RPM can work with binary and source packages. Binary packages, referred to simply as RPMs, contain compiled software ready for installation. They use the file extension `.rpm`. Source packages, more often called *source RPMs* or *SRPMs*, are uncompiled packages containing source code, patches, and build instructions, all of which are used to create binary RPMs. SRPMs have a `.src.rpm` file extension. Because RPM offers a rich feature set that makes it seem complex and difficult to learn to use, the following sections each explore one of RPM's modes, in order to simplify the discussion:

- General options
- Querying
- Package maintenance
- Administrative options
- Miscellaneous options
- Package verification

The general options control the overall behavior of the `rpm` command line tool, such as displaying basic usage information. The query functions can be used to obtain a considerable amount of information about installed software. Package maintenance enables package installation, removal, and upgrading. Package verification gives system administrators the ability to compare the present state of files installed by an RPM against information taken from the original package. The package-building mode is used to build or rebuild RPMs from source code. The administration and miscellaneous modes, finally, affect RPM itself, rather than software packages. They are used to fix possible database corruption and to determine RPM's general configuration.

General Options

At the command line, the primary RPM command is `rpm`. In addition to the mode-specific command line options discussed in the following sections, `rpm` accepts the general command line options listed in Table 30-1.

Table 30-1 General RPM Command Line Options

OPTION	DESCRIPTION
<code>-v</code>	Displays basic information about the RPM operation's status
<code>-vv</code>	Displays debugging information. Most useful to software packagers and RPM developers

Table 30-1 (continued)

OPTION	DESCRIPTION
<code>--quiet</code>	Displays only error information
<code>--help</code>	Shows a usage summary
<code>--test</code>	Performs a “dry run” of the requested operation without actually modifying the file system or the RPM database
<code>--version</code>	Shows the RPM version number
<code>--justdb</code>	Updates only the database, not the file system

The `-vv` option may prove useful when troubleshooting package installation or removal that fails, but be prepared to sort through voluminous and cryptic-looking output to find the information you need. `--justdb` is used to help repair the database. The results of any operation, such as installing or removing an RPM, affect only RPM’s databases; no files are added to or removed from the file system. `--test` makes it possible to see if a command will succeed without actually changing anything. For example, the following command uses `--test` to delete the `whois` package from RPM’s database without actually deleting the files:

```
# rpm --test -e jwhois
# rpmquery jwhois
jwhois-3.2.2-14
# ls -l /usr/bin/jwhois
-rwxr-xr-x  1 root      root          57040 Nov  9 21:28 /usr/bin/jwhois
```

The first command deletes the `jwhois` package but, because the `--test` was specified, the package wasn’t actually deleted. The second command uses the `rpmquery` command to see if the `jwhois` RPM is installed. As you can see, the package is still installed.

NOTE The `rpm` command supports more options than those listed in Table 30-1. To simplify the discussion, this chapter examines only the most common and helpful options.

One common use of `--justdb` is to remove an RPM’s entry from the database after the package has been upgraded using a non-RPM source, such as a tarball. In such cases, the RPM entry is invalid and needs to be removed without deleting the installed files. The most common use of `--justdb` is to repair the RPM database if it becomes corrupted.

Query Mode

RPM's query mode is one of its most powerful and useful features. The general form of an RPM query is:

```
rpmquery [query_opts]
```

`rpmquery` (or, if you prefer the old style, `rpm -q` or `rpm --query`) specifies a query operation and `query_opts` specifies what to query, the type of query, how the query should run, or the format of its output. You can use the command `rpmquery` in place of `rpm -q` or `rpm --query`. Most commonly, queries use the following general syntax:

```
rpmquery [query_opts] package [...]
```

`package` names the RPM to query. Query multiple RPMs using a space-separated list of package names. Query mode's power comes at the cost of a long list of options for the `query_opts` argument. The options fall into two broad categories. One group, referred to as *package selection options*, controls which package or packages to query, and the other, known as *output selection options*, defines what information to display. Table 30-2 lists many but not all of the options available in query mode. The Type column uses *S* to mark a package selection option and *I* to mark an information selection option. Unless mentioned otherwise, all options require at least one package name as an argument.

Table 30-2 RPM Query Mode Options

OPTION	TYPE	DESCRIPTION
<code>-a</code>	S	Queries all installed RPMs. Does not require a package specification.
<code>-c</code>	I	Lists only the configuration files stored in the queried RPM(s).
<code>--changelog</code>	I	Displays change information about the queried RPM(s).
<code>-d</code>	I	Lists only the documentation files stored in the RPM.
<code>--dump</code>	I	For each file stored in the queried RPM(s), displays its path, size, modification time, MD5 checksum, permissions, owner, group, and whether it is a configuration file, documentation file, a device, or a symlink (must be used with <code>-l</code> , <code>-c</code> , or <code>-d</code>).

Table 30-2 (continued)

OPTION	TYPE	DESCRIPTION
-f file	S	Queries the RPM that owns file. Does not require a package specification.
-g group	S	Lists the packages in the RPM group named group. Does not require a package specification.
-i	I	Displays complete information about the queried RPM(s).
-l	I	Lists all of the files stored in the RPM.
--last	I	Displays the installation date and time of each RPM queried, starting with the most recently installed RPM.
-p package [...]	S	Queries the uninstalled RPM named package.
--provides	I	Lists all of the capabilities the queried RPM(s) provides.
--qf 'format_str'	I	Creates a customized output format for displayed information, using format_str as the model.
--querytags	I	Prints all known tags for use with the --qf option. Does not require a package specification.
--requires	I	Lists all RPMs on which the package depends.
-s	I	For each file in the original RPM, displays its state, which is one of normal, not installed, or replaced.
--whatprovides capability	S	Queries all RPMs that provide capability.
--whatrequires capability	S	Queries all RPMs that need capability in order to function properly.

As you can see in Table 30-2, RPM's query mode is extensive and flexible, allowing you to obtain any type of information stored in the RPM databases. Using the --qf option, in fact, you can create customized query output for use in reports and scripts. The next few sections demonstrate how to use many of these options. First, here are a couple of usage tips:

- When using the -f, option, the file argument must be to a full path. That is, the command `rpm -qf /usr/bin/xmms` will show the name of the RPM that contains xmms; `rpm -qf xmms` will not.
- When specifying a package name with the -p option, you must use the complete RPM filename.

Querying Package Dependencies

The `--provides`, `--requires`, `--whatrequires`, and `--whatprovides` options allow you to identify dependencies between packages. The capability argument represents the dependency itself, which is often the name of another RPM or the name of a particular file. RPM uses dependencies to maintain system integrity, so, for example, if one RPM requires something a second RPM provides, you cannot, in normal usage, delete the second RPM. To illustrate, to determine on what capabilities the RPM package depends, use the `--requires` option as shown in the following command:

```
$ rpmquery --requires rpm
/bin/bash
beecrypt >= 4.1.2
config(rpm) = 4.4.1-18.1
fileutils
libbeecrypt.so.6
libbz2.so.1
libc.so.6
[...]
```

NOTE Throughout this chapter, RPM commands that display package version numbers might result in different version numbers on your system.

The example output is truncated to preserve space. As the output shows, the RPM package requires, in part, the `/bin/bash` capability and a `config(rpm)` capability greater than or equal to version 4.4.1-18.1. In this case, the `/bin/bash` capability refers to the existence of bash shell, `/bin/bash`, and the `config(rpm)` capability identifies the minimum acceptable version of rpm.

To identify what capabilities a package provides, use the `--provides` option:

```
$ rpmquery --provides rpm
config(rpm) = 4.4.1-18.1
rpm = 4.4.1-18.1
```

The two output lines indicate that RPM provides the capabilities (most of which are filenames, in this case) `config(rpm)` and `rpm` itself, both at version 4.4.1-18.1.

To determine which RPMs depend on a given capability, use `--whatrequires capability` to list all packages requiring capability. For example, the following command shows all the packages that require the RPM capability the RPM package provides. Although potentially confusing, keep in mind that the name of the package can be used as a capability.

```
$ rpmquery --whatrequires rpm
rpm-python-4.4.1-18.1
up2date-4.4.17-1
yum-2.3.2-2
rpm-build-4.4.1-18.1
rpm-devel-4.4.1-18.1
createrepo-0.4.2-2
```

The command output shows six packages requiring the `rpm` capability. One of the things this tells you is that these six packages depend on RPM, so before you can remove the RPM package, therefore, you have to remove the six packages that depend on it. Not that you should ever remove RPM from a Fedora Core or RHEL system, of course.

The options for querying RPM dependency information offer system administrators valuable information about the relationships between the many RPMs that constitute an installed Red Hat system.

What's in That RPM?

You will often find it useful or necessary to determine the contents of an RPM, whether it is installed or not. A number of the options listed in Table 30-2 make this possible. The possibilities range from simply displaying the package name and version numbers all the way to listing detailed information about each file an RPM installs. In fact, you can list all installed RPMs using the `-a` option. Most queries, though, fall somewhere between these extremes and query a limited subset of packages or a limited selection of package characteristics or files.

The simplest query option, `rpm -q` or `rpmquery`, shows only an RPM's name and version number, as the following command illustrates:

```
$ rpmquery jwhois
jwhois-3.2.2-1.1
```

If you want more — and more descriptive — information, add `-i`:

```
$ rpm -qi jwhois
Name           : jwhois                      Relocations: (not relocatable)
Version        : 3.2.2                      Vendor: Red Hat, Inc.
Release        : 6.FC3.1                    Build Date: Tue 09 Nov 2004 09:28:33
3 PM EST
Install Date: Thu 27 Jan 2005 08:36:32 PM EST    Build Host: tweety.build.red
hat.com
Group          : Applications/Internet        Source RPM: jwhois-3.2.2-6.FC3.1.sr
c.rpm
Size           : 200210                      License: GPL
Packager        : Red Hat, Inc. <http://bugzilla.redhat.com/bugzilla>
URL            : http://www.gnu.org/software/jwhois/
Summary        : Internet whois/nickname client.
Description    :
A whois client that accepts both traditional and finger-style queries.
```


The output wraps because several lines are longer than 80 characters. The `-i` option results in a more comprehensive listing describing the RPM. Two of the entries may require additional explanation. The Group label organizes RPM packages by function. Descriptors, which are Applications and Internet in the example, are separated by `/` and become increasingly specific as you move from left to right. Unfortunately, the values in the Group field are not standardized, and vary from vendor to vendor and even among RPM packagers. The current list of groups (see `/usr/share/doc/rpm-4.3.2/GROUPS`) in Fedora Core and RHEL is:

- Amusements/Games
- Amusements/Graphics
- Applications/Archiving
- Applications/Communications
- Applications/Databases
- Applications/Editors
- Applications/Emulators
- Applications/Engineering
- Applications/File
- Applications/Internet
- Applications/Multimedia
- Applications/Productivity
- Applications/Publishing
- Applications/System
- Applications/Text
- Development/Debuggers
- Development/Languages
- Development/Libraries
- Development/System
- Development/Tools
- Documentation
- System Environment/Base
- System Environment/Daemons
- System Environment/Kernel
- System Environment/Libraries
- System Environment/Shells
- User Interface/Desktops
- User Interface/X
- User Interface/X Hardware Support

To list all of the files in an installed RPM, use the `-l` option:

```
$ rpmquery -l jwhois
/etc/jwhois.conf
/usr/bin/jwhois
/usr/bin/whois
/usr/share/doc/jwhois-3.2.2
/usr/share/doc/jwhois-3.2.2/COPYING
/usr/share/doc/jwhois-3.2.2/NEWS
/usr/share/doc/jwhois-3.2.2/README
/usr/share/doc/jwhois-3.2.2/TODO
/usr/share/info/jwhois.info.gz
[...]
```

The `-f` file option approaches package listing from another direction. For any given file, you can find out which RPM installed it using `-f file`, where `file` contains the full path specification. The following command illustrates one way to use this option:

```
$ rpmquery -f /usr/bin/find
findutils-4.1.20-7
```

Not much to it, right? Well, suppose that you do not know the path to an application binary, just its name. In such a case, take advantage of shell commands and standard Linux utility programs. For example, the next command uses the `which` command and the bash shell's command substitution to resolve a binary's name to a full path before invoking `rpm`:

```
$ rpmquery -f $(which emacs)
emacs-21.3-21.FC3
```

As described in Appendix A, the shell replaces the value of `$(which emacs)` with `/usr/bin/emacs` before passing it to `rpmquery -f`. The result is the same as the output of `rpmquery -f /usr/bin/emacs`. You can also use the command `rpmquery -f `which emacs``, which is more shell neutral.

TIP By default, the `which` command works only with executable files available in the directories listed in the `$PATH` environment variable.

So far, every RPM query option discussed applies to installed packages. As it happens, many of them can be used on uninstalled packages, but only if the query specifies `-p`, which tells RPM to query an uninstalled package. Suppose, for example, that you just downloaded the `fortune-mod-1.2.1-1.i386.rpm` package. To list its files, use `-l`, as explained earlier, and `-p`:

```
$ rpm -qp fortune-mod-1.2.1-1.i386.rpm
/etc/profile.d/fortune.sh
/usr/bin/randstr
/usr/bin/rot
/usr/bin/strfile
/usr/bin/unstr
...
```

The output, truncated to conserve space, is identical to that of an installed package.

Formatting Query Output

Inveterate tweekers and hard-core tinkers appreciate the `--qf` option because it allows custom formatting of RPM query output. On the downside, `--qf` might not work with all query options, and RPMs rarely contain all the information that can potentially be displayed. The general form of a query using query format strings is:

```
rpmquery --qf 'format_str' [query_opts]
```

`format_str` is the workhorse of custom query formatting. A `format_str` must contain at least one tag; all other components are optional. Optional elements include literal text, directives to control the output's width and justification, control character sequences, output modifiers, and array iterators. A tag is a predefined token or symbol representing a piece of information. Examples include `SUMMARY`, `DESCRIPTION`, `NAME`, and `VERSION`, but there are many more; as of version 4.3.2, RPM understood 149 tags! Each tag must be embedded in a `%{ }` construct, for example: `%{SUMMARY}` or `%{NAME}`.

TIP Type `rpm --querytags` to view the entire list of tags RPM understands.

There are a couple of details to bear in mind. First, `format_str` should be delimited by single quotes (`'`), also called apostrophes or *strong quotes*, or by regular double quotation marks (`"`), often called *weak quotes*. Format strings used in shell scripts should be embedded between strong quotes to protect them from effects of shell expansion, described in Appendix A. Second, make sure to type `--qf` (notice the double hyphen); `-qf` means something else entirely, as described in the previous section. To avoid confusion or the possibility of a typing error, consider using the long option `--queryformat`, a synonym of `--qf`.

To keep this chapter from turning into a book about RPM, I discuss only a few format string elements. The most important are directives to control the minimum width and justification of the displayed fields and escape sequences. To specify the width of a field, place a number between a tag's percent sign and its opening brace, for example `%20{NAME}`. By default, output is right-justified, so to force left justification, prefix the field width with `-`, for example `%-20{NAME}`. The escape sequences are the same as those discussed in Chapter 28, such as `\n` for a newline and `\t` for a tab.

To illustrate, try the following examples and compare their output. The first example is the output of an unmodified query:

```
$ rpmquery setup bc hdparm popt
setup-2.5.36-1
bc-1.06-17.1
hdparm-5.7-2
popt-1.9.1-21
```

You have seen this sort of query output already. It is simple and informative but not terribly attractive. The next command uses two tags, `NAME` and `VERSION`, to specify the output fields:

```
$ rpmquery --qf '%{NAME}%{VERSION}' setup bc hdparm popt
setup2.5.36bc1.06hdparm5.7popt1.9.1$
```

Blech! This looks worse than the first example because all of the output runs together, including the command prompt. Yet, it serves as a starting point. First, separate the fields using field-width specifications:

```
$ rpmquery -q --qf '%-20{NAME}%10{VERSION}' setup bc hdparm popt
setup      2.5.36bc      1.06hdparm      5.7popt      1.9.1$
```

Each `NAME` field is 20 characters wide and left-justified. The `VERSION` column is 10 characters wide and is right-justified (the default). Judicious use of the `\t` and `\n` escape sequences solves the jumbled output problem:

```
$ rpmquery --qf '%-20{NAME}\t%10{VERSION}\n' setup bc hdparm popt
setup      2.5.36
bc          1.06
hdparm      5.7
popt        1.9.1
```

`\t`, the tab character, separates the name and version number fields; `\n`, the newline, puts the command prompt back on its own line, where it belongs.

This short discussion is only a taste of the capabilities of query formatting. Nevertheless, it provides a solid foundation for creating richer, more visually appealing query output. It is worth pointing out that the query format capability lets you create custom queries that are simply impossible using any other query option available. So, if you need RPM information you cannot obtain using the standard query options, use `--qf` to create a custom query that displays the information you need, and only that information.

Package Installation and Removal

Although RPM's query feature is one of its most powerful features, it earns its keep because of its package-management features. This section summarizes how to install, remove, and upgrade software packages using RPM.

Installing RPMs

The basic syntax for installing an RPM is:

```
rpm -i [options] package [...]
```

`package` is the complete name of the RPM to install and `options` refines the installation process. Table 30-3 lists commonly used options values. See the `rpm` man page for a comprehensive listing.

Although they appear similar, `--force` and `--nodeps` serve different purposes. `--nodeps` only disables dependency checks. Use it only if you are certain that a dependency conflict will not cause problems later on. `--force` forces a package's installation regardless of all potential problems. As a result, some situations may require using `--force` and `--nodeps` together. Common uses of `--force` include installing a package when one or more of its files conflict with files installed by another package or when any other severe installation failure would occur. Do not use `--force` to install older packages over new packages; use the `--oldpackage` option for that. To overwrite dependencies, similarly, you should use `--nodeps` whenever possible. The `--force` is a blunt instrument that might not have the results you want, expect, or intend.

The following command demonstrates installing an RPM:

```
# rpm -ivh fortune-mod-1.2.1-1.i386.rpm
Preparing...                               ##### [100%]
 1:fortune-mod                             ##### [100%]
```

Table 30-3 Common RPM Installation Options

OPTION	DESCRIPTION
<code>--force</code>	Install the package even if it is already installed, install an older package version, and replace files already installed. <code>--force</code> also ignores dependencies.
<code>-h</code>	Print up to 50 hash marks (#) to illustrate the progress of the installation.
<code>--nodeps</code>	Do not perform a dependency check before installing or upgrading a package.
<code>--test</code>	Do not install the package or update the database, just identify and display possible conflicts or dependency errors.
<code>-v</code>	Be slightly verbose and show some useful information during the installation.

The next example shows the error generated by trying to install a package already installed and how to use `--force` to ignore the error:

```
# rpm -ivh fortune-mod-1.2.1-1.i386.rpm
Preparing...                               ##### [100%]
package fortune-mod-1.2.1-1 is already installed
# rpm -ivh --force fortune-mod-1.2.1-1.i386.rpm
Preparing...                               ##### [100%]
  1:fortune-mod                             ##### [100%]
```

`--force` caused RPM to ignore the conflict and perform the installation. To avoid encountering such conflicts, use the `--test` option, as shown in the next command, to perform a “dry run” installation to catch any problems:

```
# rpm -ivh --test fortune-mod-1.2.1-1.i386.rpm
Preparing...                               ##### [100%]
package fortune-mod-1.2.1-1 is already installed
```

As you can see in the preceding example, adding `--test` to the command line generated an error message. What you cannot see is that neither RPM’s databases nor any files changed. Testing a package installation using `--test` is great protection against the heartburn caused by installing incompatible software.

Upgrading RPMs

The options for upgrading existing RPMs come in two flavors, `-U`, for *upgrade*, and `-F`, for *freshen*. What is the difference between upgrading a package and freshening it? Upgrading a package, using `-U`, installs it even if an earlier version is not currently installed, but freshening a package, using `-F`, installs it only if an earlier version is currently installed. Other than this subtle but important difference, `-U` and `-F` are identical to `-i`, even down to the options they accept (see Table 30-3).

TIP The upgrade (`-U`) option is almost always the method to use because it simply does the right thing. The only exception is when installing multiple versions of the kernel, in which case you do want to have several versions of the same package(s) installed.

The following sequence of commands illustrates how to upgrade an RPM and the difference between the `-U` and `-F` options:

```
# rpm -Fvh fortune-mod-1.0-13.i386.rpm
# rpm -q fortune-mod
package fortune-mod is not installed
```

Hmm. Nothing happened. The `rpm` command line used `-F`, so it did not install the `fortune-mod` package because an earlier version did not exist.

```
# rpm -Uvh fortune-mod-1.0-13.i386.rpm
Preparing...                               ##### [100%]
 1:fortune-mod                             ##### [100%]
```

With `-U`, RPM “upgraded” the `fortune-mod` package, even though an earlier version was not installed.

```
# rpm -Fvh fortune-mod-1.2.1-1.i386.rpm
Preparing...                               ##### [100%]
 1:fortune-mod                             ##### [100%]
```

This time, the freshen operation succeeded.

Removing RPMs

Removing or deleting RPMs and their contents is easy, perhaps frightfully so. The general form of the command is:

```
rpm -e package [...]
```

The `-e` option is a mnemonic for expunge. `package` is the name, only, of the RPM to remove. Multiple packages can be removed simultaneously by listing each package on the command line. For example, the following command removes the `fortune-mod` and `whois` RPMs:

```
# rpm -e fortune-mod whois
```

Notice that successful removal generates no additional output.

Verifying RPMs

Verifying an RPM compares the current status of files installed by an RPM to the file information recorded at the time the RPM was installed, such as file sizes and MD5 checksum values, and reports any discrepancies. The general form of the `verify` command is:

```
rpm -V package [...]
```

The `-V` option requests RPM to verify the status of files in `package`, the name of the RPM to verify. As with many other RPM operations, multiple packages can be verified simultaneously. Table 30-4 explains the file characteristics that RPM evaluates when verifying an RPM.

Table 30-4 Information Evaluated during RPM Verification

CHARACTERISTIC	DESCRIPTION
MD5 checksum	The file's MD5 checksum (calculated using the md5sum command)
File size	The file's size, in bytes
Modification time	The date and time the file was last modified
Device	The device file or files in the case of drivers and hardware devices
User	The file's owner, such as root or bin
Group	The file's group
Mode	The file's permissions and type

If none of the characteristics listed in Table 30-4 have changed for any of the RPM's files since they were installed, RPM displays no information, as the following example shows:

```
$ rpm -V jwhois
```

In this example, the `rpm` command generates no output save for the shell command prompt, so the `whois` package's files remain unchanged from their initial state at installation.

If, on the other hand, any of the tracked file characteristics have changed, the output will resemble the following:

```
$ rpm -V fortune-mod
S.5....T   /usr/bin/rot
.....T   /usr/bin/strfile
....UG.   /usr/bin/unstr
.M..... /usr/man/man1/randstr.1
missing   /usr/man/man1/unstr.1
```

At the very least, it should be clear that something is up with the `fortune-mod` package. Each line out of the output consists of eight fields and a filename, separated by white space to format the output. Table 30-5 shows the keys for interpreting verification output.

Table 30-5 RPM Verification Keys

COLUMN	VALUE	DESCRIPTION
1	5	The MD5 checksum has changed
2	S	The file size has changed
3	L	A symbolic link has changed (points to a different file)
4	T	The file's modification time has changed
5	D	The device designation has changed
6	U	The file's user (owner) has changed
7	G	The file's group has changed
8	M	The file's mode (permissions or type) has changed
ANY	.	No change detected in the corresponding characteristic
ANY	?	This characteristic's current status could not be determined (usually because file permissions prevent reading the file)
N/A	missing	The corresponding file does not exist in its default location

Translating the admittedly cryptic output from the `rpm -V fortune-mod` command, you know the following:

- `/usr/bin/rot`'s file size, MD5 checksum, and modification time have changed.
- `/usr/bin/strfile`'s modification time has changed.
- `/usr/bin/unstr`'s user and group ownership has changed.
- `/usr/man/man1/randstr.1`'s mode (either its permission bits, type, or both) has changed.
- `/usr/man/man1/unstr.1` has been deleted, moved, or renamed.

Depending on the file and the local environment, some changes indicate a potential problem, but others do not. If a binary file's size, checksum, modification time, or user or group ownership has changed and you have not manually upgraded the package, this is cause for alarm because under normal circumstances, these characteristics do not change. That is, it is highly likely that the original file has been replaced or modified, perhaps by a cracker, and you should take steps to address this problem immediately. Exercise similar caution if a file is listed as missing. On the other hand, application and system configuration files, for example, files in the `/etc` directory and its subdirectories, change due to edits by administrators and system configuration tools.

TIP Maintaining a log that tracks changes to your system, such as installing, upgrading, and removing RPMs, is a very useful habit to acquire because it records each modification made to the system. Although we used to recommend handwritten logs, it is probably more sensible to use some sort of soft format, such as a Web page on an intranet or another type of document that can be shared, the idea being that multiple administrators might need to access it.

In some situations, such as RPM verification reports, maintenance and administrative logs become an invaluable resource because you can compare log entries to the verification report to evaluate whether a reported discrepancy poses a security threat or is just the result of a long-forgotten file edit or package installation.

Unfortunately, it is impossible to define a general rule that distinguishes a legitimate change from a pernicious one. The best policy to follow is to know your system and to keep careful track of updates and changes so you can identify and respond quickly and appropriately to anomalous and potentially malicious modifications.

Building Packages Using Source RPMs

In the simplest case, building and installing software from SRPMs requires one or possibly two commands. The same unpack/configure/build/install procedure described in the previous section takes place, but RPM handles each of these steps for you. In this section, you will learn how to use the two command cases (building and installing an RPM), and how to invoke each step of the RPM build process. The general form of the command to build a binary RPM from a source RPM is:

```
rpmbuild -b[stage] spec_file [...]
```

Any of the values listed in Table 30-6 is a valid value of `stage`.

Table 30-6 Valid Build Stages for RPM'S -b Mode

STAGE	MNEMONIC	MEANING
a	All	Builds both binary and source RPMs
b	Binary	Builds only a binary RPM
c	Compile	Compiles the source code
i	Install	Installs the files
l	List	Makes sure that all the package files exist
p	Prep	Unpacks the source code and applies any patches
s	Source	Builds only a source RPM

Stages are executed in the order listed, and later stages require preceding ones, with one exception. That is, the **l** (list) step, for example, cannot be performed before the **p** (prep) stage, and the **b** (binary) stage happens after the **p**, **l**, **c**, and **i** (prep, list, compile, and install) stages have been completed. The exception is that building a source RPM (the **s** stage) does not require first building a binary RPM.

Note that the install stage of the RPM build process does not mean that files are moved into the working file system. Rather, files are “installed” in their proper paths underneath RPM’s build directory. For example, if RPM’s build directory is `/var/tmp/myrpm`, the files `/usr/bin/foo` and `/usr/man/man1/foo.1` would be installed underneath `/var/tmp/myrpm`, so their complete paths would be `/var/tmp/myrpm/usr/bin/foo` and `/var/tmp/myrpm/usr/man/man1/foo.1`. This step is necessary because of the way binary RPMs are built and how RPM installs them.

The following two commands illustrate building the `util-linux-2.12a-16` binary RPM from its corresponding SRPM (and assume that the SRPM is already installed using the instructions in the “Installing RPMs” section earlier in the chapter).

```
# cd /usr/src/redhat/SPECS
# rpmbuild -bb util-linux.spec
Executing(%prep): /bin/sh -e /var/tmp/rpm-tmp.57851
+ umask 022
+ cd /usr/src/redhat/BUILD
+ LANG=C
+ export LANG
+ unset DISPLAY
+ cd /usr/src/redhat/BUILD
+ rm -rf util-linux-2.12a
+ /usr/bin/gzip -dc /usr/src/redhat/SOURCES/util-linux-2.12a.tar.gz
+ tar -xf -
...
```

The build process generates quite a bit of output, most of which was deleted in the output listing. The **SPECS** directory contains the **spec** (presumably, short for specification) files that control RPM’s build process. The **rpm** command uses **-bb** to build a binary RPM using the instructions in the `util-linux.spec` file. As the initial few lines of output shows, RPM first decompresses the archive file, using **gzip**, and unpacks the archived files using **tar**. Additional steps apply any necessary patches, configure the package as necessary, invoke the build process, and then “install” the files as explained previously. The following two lines appear near the end of the process:

```
Wrote: /usr/src/redhat/RPMS/i386/util-linux-2.12a-16.i386.rpm
Wrote: /usr/src/redhat/RPMS/i386/util-linux-debuginfo-2.12a-16.i386.rpm
```

They indicate that RPM created two binary RPMs, `util-linux-2.12a-16.i386.rpm` and `util-linux-debuginfo-2.12a-16.i386.rpm`, in the `/usr/src/redhat/RPMS/i386` directory. First, how can one SRPM produce two binary RPMs? This is simply one of RPM's features. More importantly, why would it do so? Typically, one SRPM results in multiple binary RPMs because the binary RPMs are related, but not closely enough to put all of the programs into a single RPM. In this case, `util-linux` and `util-linux-debuginfo` are related, so the packager put them in the same SRPM. However, most people do not need the programs in `util-linux` built with debugging symbols, so it was put in its own RPM.

TIP You do not have to `cd` into the directory to build an RPM. You could just as well use a full pathname in the command, for example, `rpmbuild -bb /usr/src/redhat/SPECS/mount.spec`.

Once the packages are built, you can install them as you would any other binary RPM, as the following command illustrates:

```
# cd ../RPMS/i386
# rpm -ivh mount-2.10r-5.i386.rpm losetup-2.10r-5.i386.rpm
Preparing...                               [100%]
 1:losetup                                [ 50%]
 2:mount                                  [100%]
```

You can also build an RPM in stages by specifying one of the earlier stages. For example, the following command executes only the prep (p) stage against the `mount` SRPM:

```
# rpmbuild -bp util-linux.spec
```

The next command stops the process after the compile (c) stage:

```
# rpmbuild -bc util-linux.spec
```

Again, using the build stages in this manner is not something end users usually need to do, but the capability is there. The reason you might want to build in stages is to monitor each step of the build process to track down problems. The results of one incomplete build invocation overwrite the results of a

USING RPM WITH SOURCE TARBALLS

As it happens, you can also use RPM to build binary RPMs from source tarballs. The key requirement is that the tarball must contain a spec file in order for RPM to know how to build the binary RPM. Software developers who are not familiar with RPM often provide a spec file in their source packages for the benefit of users who prefer to use RPM to build and install software. Indeed, this is a considerable benefit for users of RPM-based systems because building and installing source tarballs bypasses RPM completely — RPM does not track dependencies or requirements of packages installed from source tarballs, so using a spec file embedded in a tarball offers a very handy way to avoid this possibility.

To use this feature specify `-t[stage]` instead of `-b[stage]`, where `stage` is one of the options listed in Table 30-7. So, for example, to build a binary RPM from `util-linux-2.10r-5.tar.gz` using an embedded spec file named `util-linux.spec`, use the following command line:

```
rpmbuild -ta util-linux-2.10r-5.tar.gz
```

This technique is especially useful if you would like to create an SRPM from a tarball for later reuse and, as noted, is an invaluable tool for maintaining an accurate an up-to-date RPM database.

previous one. Thus, if you execute `rpmbuild -bp foo.spec`, somehow change the unpacked files, and then execute another `rpmbuild -bp foo.spec`, you will lose your changes.

Checking Software Versions

Before downloading and installing SuperWidget version 1.3.2-5, you may want to know what version is currently installed in order to avoid “upgrading” to an old, unstable, development, or possibly buggy version. Unfortunately, no single command for obtaining version information works for all software packages. Rather, many methods exist. This section covers the most common methods for locating software version information.

On a Fedora Core or RHEL system, the easiest way to identify software versions is to use RPM’s query capabilities. Suppose, for example, that you want to find out which version of Emacs, a popular editor, is installed on your system. As explained earlier in the chapter, use RPM’s query option, `-q`. For example:

```
$ rpmquery emacs
emacs-21.4-3
```

The output indicates that the Emacs version is 21.4-3. To find all Emacs-related packages, not just the base packages, you can use the `-a` option to `rpmquery` and pipe the output through `grep`. For example:

```
$ rpmquery -a | grep emacs
emacs-common-21.4-3
emacspeak-21.0-2
emacs-leim-21.4-3
gnuplot-emacs-4.0.0-7
iiimf-emacs-12.2-0.7.svn2578
emacs-el-21.4-3
emacs-21.4-3
emacs-nox-21.4-3
```

What exactly does `emacs-21.4-3` mean, though? RPM uses a standardized naming and version-numbering scheme mirroring a very common approach used in the Linux development community. Its general format is `name-major_num.minor_num[.patch_num]-build_num`. Table 30-7 explains the meaning of each element in the format.

So, the package name `emacs-21.4-3` breaks down to the name `emacs`, the `major_num` 21, the `minor_num` 4, no `patch_num`, and the `build_num` 3. The build number means that it is the third version of the Emacs v21.4 that RPM built.

TIP Non-RPM software packages frequently use the same naming scheme as RPM does.

Table 30-7 Common Version-Numbering Elements

ELEMENT	INTERPRETATION
name	The name of the package (<code>balsa</code> , <code>emacs</code> , <code>mozilla</code>).
major_num	The primary version number; changes between major updates to the package.
minor_num	The secondary version number; increments to reflect updates less dramatic than those indicated by <code>major_num</code> .
patch_num	The patch number; usually reflects only the application of bug fixes to a given version. Not all applications use <code>patch_num</code> .
build_num	The build number; an RPM-specific feature indicating the packager's version.

If you do not want to use RPM, or for packages not installed using RPM, other options exist. Many applications accept command line options that cause them to display their version numbers. The most common such options are `-v`, `-V`, `-version`, and `--version`. Programs from the GNU project, in particular, almost always accept `--version`. For example, the next two examples pass `-v` and `--version` to `mutt` (a popular text-based email client) and `emacs`, respectively, to obtain their version numbers:

```
$ mutt -v
Mutt 1.4.1i (2003-03-19)
Copyright (C) 1996-2002 Michael R. Elkins and others.
Mutt comes with ABSOLUTELY NO WARRANTY; for details type `mutt -vv'.
Mutt is free software, and you are welcome to redistribute it
under certain conditions; type `mutt -vv' for details.

System: Linux 2.6.10-1.770_FC3.root (i686) [using ncurses 5.4]
Compile options:
-DOMAIN
-DEBUG
-HOMESPOOL -USE_SETGID -USE_DOTLOCK -DL_STANDALONE
+USE_FCNTL -USE_FLOCK
+USE_POP +USE_IMAP +USE_GSS +USE_SSL +USE_SASL
+HAVE_REGCOMP -USE_GNU_REGEX
+HAVE_COLOR +HAVE_START_COLOR +HAVE_TYPEAHEAD +HAVE_BKGDSET
+HAVE_CURS_SET +HAVE_META +HAVE_RESIZETERM
+HAVE_PGP -BUFFY_SIZE -EXACT_ADDRESS -SUN_ATTACHMENT
+ENABLE_NLS -LOCALES_HACK +HAVE_WC_FUNCS +HAVE_LANGINFO_CODESET
+HAVE_LANGINFO_YESEXPR
+HAVE_ICONV -ICONV_NONTRANS +HAVE_GETSID +HAVE_GETADDRINFO
ISPELL="/usr/bin/ispell"
SENDMAIL="/usr/sbin/sendmail"
MAILPATH="/var/mail"
PKGDATA_DIR="/usr/share/mutt"
SYSCONFDIR="/etc"
EXECSHELL="/bin/sh"
-MIXMASTER
To contact the developers, please mail to <mutt-dev@mutt.org>.
To report a bug, please use the flea(1) utility.

$ emacs --version
GNU Emacs 21.3.1
Copyright (C) 2002 Free Software Foundation, Inc.
GNU Emacs comes with ABSOLUTELY NO WARRANTY.
You may redistribute copies of Emacs
under the terms of the GNU General Public License.
For more information about these matters, see the file named COPYING.
```



Figure 30-1 RHN Alert Notification tool's About dialog box displays its version number.

For better or worse, many applications display more information than simple version numbers, as the example demonstrates, but at least you get the information you sought.

NOTE When invoked as discussed in this section, most applications display their version information and then exit (that is, they do not start).

Many X Window system applications use `-version` to show their version numbers. Similarly, almost every X application provides a dialog box (commonly accessible by selecting Help ⇨ About) that displays a version number. Figure 30-1, for example, shows the Red Hat Network Alert Notification tool's About dialog box, which includes version information.

If none of these suggestions work, try using `-help` or `--help` to generate a short usage summary that lists a program's options. If all else fails, read the package's documentation or manual page to find out how to access its version information, because it is very rare for an application, whether text mode or GUI, not to display version information.

Obtaining Newer Software

Using the Red Hat Update Agent to keep your system up to date with the latest bug fixes and security patches is fine. But where do you go if you want software that isn't maintained by the Fedora Project or Red Hat Software? What's a poor user jonesing for something new and different to do? Suppose, for example, that you need to get RPMs for applications that play or provide support for MP3 audio formats? Naturally, you download it from one of the popular download sites for Fedora and RHEL software. In some cases, you'll need to build it from scratch (as in, "Use the source, Luke.").

Using Third-Party Sites to Find RPMs

For the terminally impatient, here's a list of our favorite sites for Fedora Core and RHEL packages not available as part of the officially blessed distributions:

- **Dag Wieers' Red Hat/Fedora RPM Repository (<http://dag.wieers.com/packages>)** — At the time this was written Dag Wieers had collected 34,103 RPMs ranging from aalib, an ASCII art library, to zziplib, a library for extracting data from ZIP archives. Dag's site also has RPMs for MP3-friendly audio applications. Not wishing to mislead you, many of these RPMs are available for multiple versions of Fedora and RHEL and the now-discontinued desktop products formerly known as Red Hat Linux. Nonetheless, Dag's count is that these 39,572 RPMs consist of 2166 distinct products. Not too shoddy for a noncommercial project.
- **freshrpms.net (freshrpms.net)** — freshrpms.net is the personal itch of Matthias Saou. He started by building RPMs for software that he couldn't find in RPM format. As he describes his project, "consider freshrpms.net as my modest contribution, by helping Red Hat Linux users get easy access to high quality add-on packages of neat software not included in the main distribution. The packages themselves are focused on quality, and not quantity. This means that you won't find every software you can think of here, but what you will find should blend into Red Hat Linux as if it was part of the original distribution (<http://freshrpms.net/about>)." At the time this was written, Matthias had 743 binary and source RPMs for Fedora Core 3 alone.
- **RPM PBone Search (<http://rpm.pbone.net>)** — RPM PBone Search is more of a RPM search tool than an RPM archive site. It helps you locate RPMs that satisfy a given dependency by performing a search for RPMs that contain a filename you provide. It can also perform searches of the RPM "provides" field as well as RPM summary and description fields. RPM PBone Search "is especially helpful for users who install or compile new software but for some reasons haven't installed all of the required [librarie]s or files and are not able to investigate what RPM packet is missing on their system. Additionally, you can narrow search down to a few releases or to one Linux release. 'RPM PBone Search' is also designed to locate RPM files on various FTP servers (<http://rpm.pbone.net/index.php3/stat/5>)." At the time this was written, RPM PBone Search had a total of 603,817 RPMs in its database, so chances are good you'll find what you're looking for.
- **Axel Thimm's RPMs (ATrpms) (<http://atrpms.net>)** — ATrpms initially started out as a repository for software (in RPM format) used

in the natural sciences, especially high-energy physics. Over time, however, the site grew into a more general-purpose RPM repository. Although a count of available RPMs isn't available, ATrpms, unlike some of the other RPM repositories listed here, categorizes the RPMs it indexes according to the age, quality, and stability of packages. You can select packages by Fedora Core or Red Hat version and also by topic or name.

- **rpm.livna.org (<http://rpm.livna.org>)** — rpm.livna.org is an extension of the Fedora Extras project at fedora.us/. rpm.livna.org carries packages that are not part of the Fedora Core and that cannot, for various reasons, be carried on the fedora.us site. In particular, as the rpm.livna.org FAQ explains, it distributes packages that are not acceptable to fedora.us due to licensing or patent restrictions that are inconsistent with Red Hat's policies, such as licenses that prohibit non-commercial use. (See <http://rpm.livna.org/faq.html> for further information.) Packages from rpm.livna.org are not compatible with packages from the other sites in this list with the exception of Fedora Extras from fedora.us or fedoraproject.org.
- **Fedora Extras (fedora.us/)** — If you have an older version of Fedora Core, before Fedora Core 3, this site might be for you. However, be aware that packages from this site are incompatible with packages from the other sites in this list. Moreover, versions of Fedora Core 2 and earlier are now running in maintenance mode, which means that little if any *new* development is taking place for it. All the interest and excitement has shifted to newer releases.
- **rpmfind.net (<http://rpmfind.net>)** — The rpmfind.org Web site is the oldest third-party RPM repository and maintains an enormous number of RPM-based software packages covering all the major distributions and many of the minor ones and spanning the entire range of application categories. It also lists RPMs by platform, such as Intel x86, SPARC, and Macintosh. Its finely tuned search engine makes it easy to find the RPM you want, or at least to reduce the possibilities from some 100,000+ to a more manageable figure. It also is the home site for the rpm2html and rpmfind tools, which you can run on your own machine to streamline the process of finding just the right RPM for your needs.
- **FedoraProject.org (<http://www.fedoraproject.org/wiki/Extras/>)** — This site is the home for Fedora Core Extras, RPMs built to Fedora Core standards but not included in the standard distribution. Be sure to read the instructions for using the Fedora Extras at fedoraproject.org/wiki/Extras_2fUsingExtras.

With the exception of `fedora.us` and `fedoraproject.org`, none of these sites are endorsed by Red Hat or the Fedora Project, so the quality of the RPMs you can get might be mixed.

Using Ibiblio.org

Ibiblio.org is arguably the sire of all of the Linux and open-source software repositories currently available. It began life as Sunsite (strictly speaking, `ftp://sunsite.unc.edu` and, after the World Wide Web became popular, `http://sunsite.unc.edu`), one of a number of information repositories scattered throughout the world, hosted on hardware donated by Sun Microsystems (hence, the name) and provided as a public service by both Sun and the sites' administrators. In the late 1990s, the University of North Carolina renamed Sunsite to Metalab, and the URLs changed to `ftp://metalab.unc.edu` and `http://metalab.unc.edu`. In 1999, Metalab became Ibiblio.org, reflecting its phenomenal growth in size and popularity and its focus on providing the computing public at large access to a comprehensive store of information (and software) on a variety of topics.

NOTE Ibiblio.org (and its predecessors, Metalab and Sunsite) is a public repository for information of all sorts, not just downloadable software. For example, at the time this chapter was written, it featured an exhibit commemorating America's World War II veterans in honor of Memorial Day, including articles, photographs, and multimedia items. At the same time, Ibiblio.org hosted a stunning 55 GB of downloadable Linux software, and Linux software is only one of Ibiblio.org's software collections!

To access Ibiblio.org's Linux archive, point your Web browser at `ftp://www.ibiblio.org/pub/Linux`. Browse through the archives at your leisure to locate the package that interests you. As usual, to download one to your system, click the corresponding link. If you are using the Netscape browser, it may be necessary to hold down the left Shift key and left-click to download the archive file.

TIP Ibiblio.org does not store RPM files in its standard Linux software archive. Rather, the preferred format is compressed tarballs. However, distribution-specific directories (stored under the `/pub/Linux/distributions` directory tree) do contain RPMs if the distribution in question is RPM-based (such as Red Hat Linux and Linux-Mandrake).

If you are uncertain about the contents of a particular directory or its subdirectories, the files `!INDEX`, `!INDEX.html`, `!INDEX.short.html`, and `README` in each directory describe, in varying levels of detail, the contents of each directory and, if applicable, its subdirectories. If you prefer not to browse `Ibiblio.org`'s Linux archive directly, you can use its Linsearch interface, based on the Linux Software Map described in the next section, by pointing your browser at `ibiblio.org/pub/Linux` and selecting the Linsearch link.

TIP The Linux Software Map, less actively maintained than other sites discussed in this chapter, is a database of Linux software. It supports searches by keywords and titles and also lets you browse the entire database by application title.

Installing Software from Source

After downloading either an RPM, SRPM, or a source archive such as a tarball, naturally, you will want to install it. The section titled “Using the Red Hat Package Manager” detailed how to install binary RPMs and how to create binary RPMs from SRPMS. This section shows you how to install software from source code, that is, to unpack, configure, build, and install a software package you download as uncompiled source code.

Before package-management suites such as RPM became popular, you upgraded and installed software by using the following steps:

1. Downloading a *gzipped tarball* (a tar archive compressed using the *gzip* utility).
2. Unpacking it.
3. Configuring it by manually editing one or more header files or using a configure script that automatically customized the package to your system.
4. Executing `make` to build it.
5. Executing `make install` to install it.

Despite RPM's popularity, a considerable amount of software still uses this approach, so this section walks you through the process to familiarize you with it. The method is much simpler than it might seem at first blush. The example used in this section configures, builds, and installs `bc`, an arithmetic-processing language used in the text mode `dc` calculator.

Configuring the Build Environment

Implicit in the previous paragraph is the assumption that your system has a development environment installed. For end users and system administrators, a build environment consists of the compiler, `gcc`, and its supporting libraries, the `make` utility for automating compiler invocations, a few key development libraries (mostly consisting of header files), and the `install` utility for handling the details of copying files and assigning the proper ownership and setting file permissions appropriately.

During installation, select the Development workstation option to install a complete development environment. Otherwise, make sure that at least the following RPMs are installed on your system:

- `bzip2`
- `fileutils`
- `g++` (for C++ programs)
- `gcc` (for C programs)
- `glibc-devel`
- `gzip`
- `kernel-headers`
- `kernel-source`
- `make`
- `ncompress`
- `tar`

Packages that have additional requirements usually state what they are, so read the package documentation to ensure that your system has the required software.

Unpacking the Source Code

After downloading the `bc` package (`bc-1.0.6.tar.gz` in this chapter's source code directory), move it to a location where it will not interfere with the system. For this example, I moved it to `/tmp`. After `cd`ing to `/tmp`, you can use one of two commands to decompress and unpack the archive. The first command combines decompression and unpacking the tar archive in a single command:

```
$ tar xzf bc-1.0.6.tar.gz
```

The `z` option uses `tar`'s built-in `gunzip` routine to decompress the file, `x` extracts the archive files, and `f` specifies the name of the file on which to operate. The second command you can use sends `gunzip`'s output to `tar`'s input using a pipe (`|`), that is:

```
$ gunzip -c bc-1.0.6.tar.gz | tar xf -
```

TIP If you want more feedback from either of these commands, use `tar`'s `v` option to cause it to display the files it is extracting from the archive.

`gunzip`'s `-c` option sends the result of the decompression to standard output; using `-f -` with `tar` tells it to read its standard input; the pipe connects the two commands. In either case, you wind up with a directory named `bc-1.0.6` in the current directory (`/tmp`, in this case). So, `cd` into `bc-1.0.6` to proceed with configuring `bc`.

Configuring the Source Code

Now that the `bc` package has been unpacked, the next step is to configure it for your system. In most cases, customizing a package for your system boils down to specifying the installation directory, but many packages, including `bc`, allow you to request additional customizations. Happily, this is an easy step.

`bc`, like numerous other software packages, uses a `configure` script to automate configuration and customization. A `configure` script is a shell script that makes educated guesses about the correct values of a variety of system-specific values used during the compilation process. In addition, `configure` allows you to specify the values of these same values, and others, by invoking `configure` with command line options and arguments. Values that `configure` “guesses” and that you pass to `configure` on its command line are normally written to one or more makefiles, files that the `make` program uses to control the build process, or to one or more header (`.h`) files that define the characteristics of the program that is built.

To see the items you can customize, execute `./configure --help` in the base directory of the package you are building, as shown in the following example (which has been edited to conserve space).

```
$ ./configure --help
Usage: configure [options] [host]
Options: [defaults in brackets after descriptions]
Configuration:
...
```

```
Directory and file names:
--prefix=PREFIX          install architecture-independent files in
PREFIX                  [/usr/local]
--exec-prefix=EPREFIX    install architecture-dependent files in
EPREFIX                  [same as prefix]
...
Host type:
--build=BUILD            configure for building on BUILD [BUILD=HOST]
--host=HOST              configure for HOST [guessed]
--target=TARGET          configure for TARGET [TARGET=HOST]
Features and packages:
--disable-FEATURE       do not include FEATURE (same as --enable-
FEATURE=no)
--enable-FEATURE[=ARG]  include FEATURE [ARG=yes]
--with-PACKAGE[=ARG]    use PACKAGE [ARG=yes]
--without-PACKAGE       do not use PACKAGE (same as --with-PACKAGE=no)
--x-includes=DIR        X include files are in DIR
--x-libraries=DIR       X library files are in DIR
--enable and --with options recognized:
--with-pkg              use software installed in /usr/pkg tree
--with-libedit          support fancy BSD command input editing
--with-readline         support fancy command input editing
```

The key options in the example output are `--prefix` and the three options that appear under the heading `--enable` and `--with` options recognized, `--with-pkg`, `--with-libedit`, and `--with-readline`. `--prefix` enables you to specify an installation directory other than the default (indicated in brackets, `[]`), `/usr/local`. For this example, the root installation directory is `/tmp/bctest`, specified as `--prefix=/tmp/bctest` on `configure`'s command line. The second group of command line options enables other features. This example uses `--with-readline`, which turns on support for the GNU readline library. The readline library enables command line editing inside the `bc` program, just as the `bash` shell permits editing the shell command line.

After selecting the desired options, run `configure` with the appropriate options, as shown in the following example. (Again, the output has been edited to conserve space.)

```
$ ./configure --prefix=/tmp/bctest --with-readline
creating cache ./config.cache
checking for a BSD compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking whether make sets ${MAKE}... yes
checking for working aclocal... found
checking for working autoconf... found
checking for working automake... found
```

```

checking for working autoheader... found
checking for working makeinfo... found
checking for gcc... gcc
checking whether the C compiler (gcc ) works... yes
...
checking for readline in -lreadline... yes
checking for readline/readline.h... yes
Using the readline library.
updating cache ./config.cache
creating ./config.status
creating Makefile
creating bc/Makefile
creating dc/Makefile
creating doc/Makefile
creating lib/Makefile
creating config.h

```

The lines beginning with checking indicate that configure is testing for the presence of a certain feature such as gcc. Because the command line specified `--with-readline`, the last two checking lines make sure the readline library is installed (checking for readline in `-lreadline... yes`) and that the appropriate header file, `readline.h`, is installed. Once all of the tests are completed, configure uses the test results to create a number of Makefiles and a header file.

TIP If you are in the habit of building software as the root user, stop! It is extremely rare to require root access to build software. The only step that needs root access is the `make install` step, which requires write permissions to the installation directories. We routinely build the kernel and major system applications as mortal users, only using `su` when we are ready to install.

At this point, you are ready to build `bc`.

Building the Software Package

To build `bc`, type **make** and press Enter. The following example shows the end of the build process's output:

```

$ make
...
gcc -DHAVE_CONFIG_H -I. -I. -I.. -I../.. -I../.. -g -O2 -Wall
-funsigned-char -c dc.c
gcc -DHAVE_CONFIG_H -I. -I. -I.. -I../.. -I../.. -g -O2 -Wall
-funsigned-char -c misc.c
gcc -DHAVE_CONFIG_H -I. -I. -I.. -I../.. -I../.. -g -O2 -Wall
-funsigned-char -c eval.c

```



```
gcc -DHAVE_CONFIG_H -I. -I. -I.. -I./.. -I./../h -g -O2 -Wall
-funsigned-char -c stack.c
gcc -DHAVE_CONFIG_H -I. -I. -I.. -I./.. -I./../h -g -O2 -Wall
-funsigned-char -c array.c
gcc -DHAVE_CONFIG_H -I. -I. -I.. -I./.. -I./../h -g -O2 -Wall
-funsigned-char -c numeric.c
gcc -DHAVE_CONFIG_H -I. -I. -I.. -I./.. -I./../h -g -O2 -Wall
-funsigned-char -c string.c
gcc -g -O2 -Wall -funsigned-char -o dc dc.o misc.o eval.o sta
ck.o array.o numeric.o string.o ../lib/libbc.a
make[2]: Leaving directory `/tmp/bc-1.06/dc'
Making all in doc
make[2]: Entering directory `/tmp/bc-1.06/doc'
make[2]: Nothing to be done for `all'.
make[2]: Leaving directory `/tmp/bc-1.06/doc'
make[2]: Entering directory `/tmp/bc-1.06'
make[2]: Leaving directory `/tmp/bc-1.06'
make[1]: Leaving directory `/tmp/bc-1.06'
```

Depending on the size and complexity of the program you are building, `make`'s output might be extensive. In the example shown, you see the final compiler invocations and, most importantly, no errors. Accordingly, the next step is to test the build.

Testing the Build

Many programs, especially those from the GNU projects, include some sort of test suite to validate the program. The idea is to make sure that the program works properly before installing it. In some cases, you execute the `make test` command to run the test suite. In other cases, as with `bc`, a special subdirectory of the build tree, conveniently named `test` or `Test`, contains the test suite. Each package handles testing slightly differently, so read the package documentation. In the case of `bc`, the test suite lives in a subdirectory named `Test`, and a shell script named `timetest` performs the actual test. In this case, `timetest` evaluates how long it takes `bc` to perform certain mathematical calculations, but it also serves to ensure that `bc` built properly. The following commands invoke `bc`'s test suite:

```
$ cd Test
$ ./timetest
```

`timetest` takes at least 10 minutes to run, so have a cup of coffee or your favorite beverage while the test runs. If no errors occur during the test, you are ready to install it.

Installing the Software

In the case of `bc`, as with many, many other programs installed from source, installing the built and tested program is simply a matter of executing the command `make install` in the build tree's base directory (`/tmp/bc-1.0.6`, in this case). Programs that are more complex might have additional commands, such as `make install-docs` to install only documentation, that break up the installation into more steps or that perform only part of the installation. Still other packages might use scripts to perform the installation. Regardless of the process, however, the goal is the same: Install program executables and documentation in the proper directories, create any needed subdirectories, and set the appropriate file ownership and permissions on the installed files.

In the case of the `bc` package, the installation command is a simple `make install`, as shown in the following code:

```
$ make install
...
/bin/sh ../mkinstalldirs /tmp/bctest/bin
mkdir /tmp/bctest
mkdir /tmp/bctest/bin
  /usr/bin/install -c bc /tmp/bctest/bin/bc
...
make install-man1
make[3]: Entering directory `/tmp/bc-1.06/doc'
/bin/sh ../mkinstalldirs /tmp/bctest/man/man1
mkdir /tmp/bctest/man
mkdir /tmp/bctest/man/man1
  /usr/bin/install -c -m 644 ./bc.1 /tmp/bctest/man/man1/bc.1
  /usr/bin/install -c -m 644 ./dc.1 /tmp/bctest/man/man1/dc.1
...
```

The output, edited to conserve space, shows the creation of the installation directory, `/tmp/bctest` (recall the `--prefix=/tmp/bctest` command line option passed to `configure`), a subdirectory for the binary (`/tmp/bctest/bin`) and the subdirectory for the manual pages, `/tmp/bctest/man/man1`. The output also shows the invocation of the `install` program that actually performs the installation. The `-c` option is ignored because it is used for compatibility with `install` programs used on proprietary UNIX systems. The `-m` option sets the file permissions using the octal permission notation. So, `-m 644` makes the files `bc.1` and `dc.1` (which are manual pages) read/write for the file owner and read-only for the file group and all other users.

NOTE For more information about the `install` program, read the manual page (`man install`) or the TeX-info page (`info install`).

At this point, package installation is complete. Although this example of building and installing a package from a source tarball is simple, the basic procedure is the same for all packages: unpack the source archive, configure it as necessary, build it, test the program, and then install it. One final exhortation before proceeding to the next section: Read the documentation! Most software you obtain in source code form includes one or more files explaining how to build and install the software; we strongly encourage you to read these files to make sure that your system meets all the prerequisites, such as having the proper library versions or other software components. The documentation is there to help you, so take advantage of it and save yourself some frustration-induced hair loss!

Summary

This chapter covered a lot of territory. You learned how to use each of RPM's major operating modes, including querying the RPM database; installing, upgrading, and removing RPMs; and maintaining the RPM database. You also learned methods for obtaining the version information of installed software. The chapter listed some popular software repositories and how to use them. Finally, you learned how to build and install software from source using both the traditional tools (`tar`, `gcc`, `make`, `install`) and RPM's higher-level interface to these tools.

Backing Up and Restoring the File System

IN THIS CHAPTER

- Creating a Backup Plan
- Choosing Media for Backups
- Understanding Backup Methods
- Tape Rotation
- Using Backup Tools

In this chapter, you learn how to make backups of your files and restore damaged file systems from backups. It is important to make backups of the file system to avoid the loss of important information in the case of catastrophic hardware or software failure. An efficient backup and restoration process can minimize downtime and prevent the need to recreate lost work. In this chapter, you learn how to choose a backup medium and how to use backup tools. Red Hat Enterprise Linux provides several packages for dealing with the backup and restoration of the file system. The `tar` and `dump` commands provide low-level interfaces to system backups. In addition, sophisticated backup tools, such as AMANDA, can do automatic backups of multiple machines.

Creating a Backup Plan

Determining what to back up on a particular machine depends largely on what data the machine contains and how the machine is used. However, there are some general guidelines that can be useful in determining what to back up.

Generally, temporary and cached files need not be backed up. The contents of the `/tmp` directory, for instance, are usually deleted when the machine is

rebooted. Therefore, it is all right to not back up these files. Also, the cache directory used by Mozilla and found in users' `.mozilla` directory is automatically regenerated by Mozilla if it is deleted. You may find it worthwhile to investigate whether any other packages installed on the machine generate significant amounts of ignorable temporary data.

Depending on the situation, it may or may not be advisable to back up the machine's system files. If the machine is a standard installation of Red Hat Enterprise Linux without any customizations or extra packages installed, the system files can be restored by reinstalling Red Hat Linux. The tradeoff is that reinstalling and reconfiguring a system probably takes more time and attention than restoring the file system from backup. However, this tradeoff may be worthwhile because of the amount of backup media that can be saved. In the particular case that a single Red Hat installation is copied verbatim onto many machines, it may be appropriate to back up the system files of just one of the machines. If the system files are identical across machines, a single backup should restore all of them. In any case it is probably wise to back up at least the `/etc` directory of each machine. Probably the machines have at least some differing configuration information, such as network and hostname settings.

One other thing needs to be backed up and indeed needs to be backed up via a special method: database files. Doing a straight `tar` from database files won't save you from a database crash, because the database files will all be in different states, having been written to backup when opened. Oracle, Informix, Sybase, and so forth all allow the administrator to export the table data in text or other delimited file as well as put the database tablespaces in backup mode. In backup mode, the data to be written goes to a memory cache rather than the file, and transaction logs are updated only when the cache is flushed. This procedure slows things down but makes certain that the database will survive a crash.

The other aspect of the file system, other than the system files that need to be backed up, is the user files. Generally, all user files are stored in subdirectories of the `/home` directory. You should find it easy, therefore, to back up all user files at one time. Even when the entire file system — both system and user files — is being backed up, you should still back them up separately. System and user files can have different relative priority depending on the situation. The user files are important because they may be irreplaceable, whereas many of the system files generally can be replaced by reinstalling Red Hat Linux. On the other hand, restoration of the system files is necessary for the machine to function and provide services, whereas the machine can be totally functional without restoration of the user files. Such priority considerations must be made when designing a backup strategy.

Give special thought to resources that do not easily fall into the system and user categories. Information stored in SQL databases, for instance, is often

technically owned by root or by a special system user, but also often contains irreplaceable content entered by users. This kind of data can often be the most important to back up. You may find it beneficial to investigate which of the installed packages use this kind of data. Other examples besides databases are Web servers and mailing list archivers.

Choosing Media for Backups

A variety of backup media are available on the market today. Which backup media you use depends on a number of factors and the particular needs of the situation. You should consider how often files are backed up, how long the backups need to last, how redundant the backups need to be, and how much money can be allocated to purchasing backup media. Table 31-1 provides a comparison of backup media.

Table 31-1 Comparison of Backup Media

MEDIUM	CAPACITY	RELIABILITY	COST	SPEED
Magnetic tape	High	High	Cheap	Slow
Writable CDs	Medium	Medium	Cheap	Fast
Hard drive	High	High	Expensive	Fast
Floppy disks	Low	Low	Cheap	Slow
DVD	High	High	Medium	Slow
Zip disks	Medium	Low	Medium	Slow
Flash ROM	Medium	High	Expensive	Fast
Removable hard drive (FireWire)	High	High	Expensive	Fast
Removable hard drive (USB)	High	High	Expensive	Medium

Understanding Backup Methods

To save time and money in creating backups and restoring corrupted file systems and in purchasing backup media, it is important to institute a methodology for creating scheduled backups. The number of different backup methodologies is unlimited. How you should perform backups depends on the particular needs

of your institution and computing infrastructure. The scheduling and type of backups depends on the type of backup media being used, the importance of the data, and the amount of downtime you can tolerate.

The simplest backup methodology is creating a full backup. A full backup copies the entire file system to the backup medium. This methodology can be good for small systems in which there is not much data to back up or systems in which the data is very important, where it is changing rapidly, and where historical snapshots of the system at different points in time are useful.

Performing frequent full backups has several disadvantages. Full backups take a long time to perform if there is a lot of data to back up or if the backup medium is slow. To get a clear snapshot of the system, you may need to suspend the execution of processes that modify the file system while the backup process takes place. If backups take a long time, the downtime might be prohibitive. Full backups have no disadvantages when it comes to restoring an entire file system from backup. However, there is a disadvantage when restoring a partial file system from backup. If a sequential medium, such as magnetic tape, is used, it must be searched sequentially to find the files that need to be restored. This process can cause a partial restoration to take as long as a full file system restoration in the worst case. Full backups also take significantly more space to archive than incremental backups. This situation is not too much of a disadvantage if you reuse the same backup media; you can just overwrite the old backup with the new one. However, it is often advisable to keep multiple generations of backups. Sometimes problems with the file system, such as corrupted or erased files, are not detected or reported immediately. If the file system is backed up once a day on the same backup tapes and an accidentally erased file is not found for two days, it cannot be recovered. On the other hand, if the file system is backed up once a week, any files lost between backups cannot be recovered. Keeping multiple full backups also has a disadvantage. If a full backup is made every day, the amount of archive media necessary to store it quickly becomes prohibitive.

The alternative to doing full backups is to do incremental backups. An incremental backup archives only the files that have changed or been added since the last backup. Incremental backups solve all of the disadvantages of full backups. Incremental backups are fast. In fact, the more often you do them, the faster they are because there is less to back up. Since the backups are smaller, searching from a given backup for a particular file is faster, thus making partial restorations faster if you need to restore from a particular known incremental backup archive. Because less is backed up each time, less media is used, so either less backup media needs to be bought or a longer history can be kept in the same amount of backup media. In the latter case, backups are more robust against lost or damaged files that are not discovered for a while.

Using incremental backups has disadvantages as well. While incremental backups are faster for retrieving individual files, they are slower for restoring entire file systems. To explain this problem, imagine that you have a week-long backup cycle. On the first day of the week, you make a full backup. The rest of the week, you make an incremental backup. If a file system is erased accidentally on the last day of the week (right before a new full backup is to be made), you have to start at the last full backup and then load in a whole week of tapes to entirely restore the file system. If you made a full backup every day, you would have to load only the full backup, then you would be done restoring the file system.

When to use full backups and when to use incremental backups depends on the particular data stored on the machines, the way the machines are used, and how much money can be allocated to buying backup media. After you have decided on a backup methodology, you must configure your tools to use this methodology. Full and incremental backups can be implemented in scripts on top of the primitive backup tools such as `tar`. More advanced tools such as `dump` and `AMANDA` have built-in support for backup levels and scheduling of various kinds of backups. `AMANDA` even has a complex configuration language that lets you specify all kinds of details about the various types of backups you might want to do, the length of your backup cycle, and what files should be excluded from backup (such as private or temporary files).

Another thing to consider is the criticality of the system. If the system must be up at all times and downtime is a critical situation, then full backups are necessary to minimize downtime. One strategy for backing up critical machines is to create a separate volume group on mirrored disks solely for backups and use it as an intermediate area to copy files to prior to writing them to tape. A compressed `tar` file can be created on disk and then be written to tape faster than a normal `tar` file can. Also, since a backup exists on disk, the tape archive is only used as a last resort if the disk archive fails. This strategy is similar to the one that the `AMANDA` automated backup utility uses to take into account faulty backup devices or media. Even if the tape drive fails, the backup on disk can be written to tape when the problem has been solved.

Tape Rotation

Another consideration after you select a backup method is a proper tape rotation schedule. A well-thought-out schedule can lower your media costs and increase the life of your tapes, while ensuring that every file is protected. Several popular tape rotation methods are currently in use.

Grandfather-father-son (GFS) is probably the most common tape rotation scheme. The grandfather backup is a monthly full backup, the father is a weekly full backup, and the son is a daily incremental backup. It is usually a good idea, and more secure, to store at least the full monthly backups off-site. In the event of a catastrophe at your location, a fire that damaged or destroyed your on-site backups, for example, you would be able to restore your data from tapes stored off-site.

TIP For a detailed explanation of tape rotation methods, a good place to look is the Seagate Web site: seagate.com/products/tapesales/backup/A2g1.html.

Using Backup Tools

Fedora Core and Red Hat Enterprise Linux provide numerous tools for doing file system backups. There are tools for interacting with backup media, such as `ftape` for manipulating tapes drives and `cdrecord` for writing to CD drives. Command line tools such as `tar` and `dump` allow for low-level control of file system backups and also easy automation through scripting. Using only shell scripts and periodic scheduling through cron jobs, you can develop a robust automated backup solution for many situations. Graphical tools also exist to create a more user-friendly interface for performing manual backups. Advanced backup tools that can be configured to fully automate the process of backing up multiple machines do exist.

Command Line Tools

Fedora Core and Red Hat Enterprise Linux provide a number of command line tools for performing backups and restoring from backups. The tools for interacting directly with backup media are `mt-st` and `cdrecord`. The standard tools for creating archives are `tar` and `dump` for tape archives and `mkisofs` for CD archives. Each command provides a different interface and a number of options.

Using `mt-st`

The `mt-st` package is a collection of command line tools for accessing and managing magnetic tape drives (the `mt` part of the package) and also includes the module that is used to control the SCSI tape drive (the `st` part of the package), or an IDE drive in SCSI emulation mode. This package is installed by default

on Fedora Core and Enterprise Linux and is intended to control tape drives connected to a SCSI bus. You can check to be sure the package is installed by entering the following at a command line interface.

```
rpm -q mt-st
```

The command will return `mt-st <version number>` if the package is installed and nothing if the package is not installed. In the event that the package is not installed, find it on the installation CDs and install it using the following command:

```
rpm -Uvh mt-st<version number> (Be sure to use the proper version number)
```

To be able to communicate with your tape drive, the `st` module must be loaded. You can determine whether the `st` module is loaded by entering the following command:

```
/sbin/modinfo st
```

If the module is installed you will see output similar to this:

```
filename:      /lib/modules/2.6.10-1.770_FC3/kernel/drivers/scsi/st.ko
author:        Kai Makisara
description:    SCSI Tape Driver
license:        GPL
parm:          buffer_kbs:Default driver buffer size for fixed block mode (KB;
32)
parm:          max_sg_segs:Maximum number of scatter/gather segments to use
(256)
parm:          try_direct_io:Try direct I/O between user buffer and tape drive
(1)
parm:          try_rdio:Try direct read i/o when possible
parm:          try_wdio:Try direct write i/o when possible
vermagic:      2.6.10-1.770_FC3 686 REGPARM 4KSTACKS gcc-3.4
depends:        scsi_mod
srcversion:     0ECB594BCDEAA75405B3302
```

If the module is not installed, you will see a module not found message. You can install the module using the following command:

```
insmod st
```

After you install the module, you should reboot the server so the `st` module can identify and connect the tape nodes, which are `/dev/st#`. When the system finishes booting, you can use the `dmesg` command to get a listing of the tape device. Look for information similar to the following:

```
(scsi0:A:6): 10.000MB/s transfers (10.000MHz, offset 15)
Vendor: COMPAQ      Model: DLT4000      Rev: D887
Type:   Sequential-Access              ANSI SCSI revision: 02
st: Version 20040403, fixed bufsize 32768, s/g segs 256
Attached scsi tape st0 at scsi0, channel 0, id 6, lun 0
st0: try direct i/o: yes (alignment 512 B), max page reachable by HBA 1048575
```

From this output, you can see the SCSI tape drive is identified as `st0`. To communicate with this drive you would use `/dev/st0`. There are actually eight device nodes associated with the tape device. Four of the nodes indicate autorewind nodes and four indicate no rewind nodes. The autorewind nodes are indicated by `st0*`, and the nonrewind nodes are indicated by `nst0*`. You can see a listing of these different nodes by running the `ls -ld` command for the devices. For example, to see the autorewind nodes do this:

```
ls /dev/st0*
```

You will see this output:

```
/dev/st0 /dev/st0a /dev/st01 /dev/st0m
```

To see the no rewind nodes, enter `ls /dev/nst0*`, and you will see the following:

```
/dev/nst0 /dev/nst0a /dev/nst01 /dev/nst0m
```

When you are communicating with the tape device, you use `/dev/st0*` when you want to rewind the tape and `/dev/nst0*` when you do not want to rewind the tape. You will see some examples later in the section on the `mt` and `tar` commands.

The `mt` command is used to perform many tape control functions, such as scanning, rewinding, and ejecting magnetic tapes. Take a look at some examples.

You must be root to access the tape drives. As root, you can test a new magnetic tape by inserting it into the tape drive and using the following command:

```
mt -f /dev/st0 status
```

This command will return output similar to the following:

```
drive type = Generic SCSI-2 tape
drive status = 318767104
sense key error = 0
residue count = 0
file number = 0
block number = 0
Tape block size 0 bytes. Density code 0x13 (DDS (61000 bpi)).
Soft error count since last status=0
```

```
General status bits on (45010000):  
BOT WR_PROT ONLINE IM_REP_EN
```

In addition to giving you status information about the tape, it will rewind it to the beginning. If you run the same command but use `/dev/nst0` instead, you would receive the status information, but the tape does not rewind. There are many options available to you with the `mt` command. The best source of information is the `mt` manual page, which you can access by typing `man mt` at a command line.

Using the *cdrecord* Package

To make backups on CDs under Red Hat Enterprise Linux, you need the `cdrecord` package installed. It contains several commands such as `cdrecord`, `devdump`, `isodump`, `isoinfo`, `isovfy`, and `readcd`. These commands are useful utilities for creating and managing writable CDs.

The disadvantage to making backups on CD is that you must first create a CD image on the file system and then copy the CD image to the actual CD all in one step. This process requires that you have empty space on a single file system partition that is large enough to hold a CD image (up to 650 MB). You create a CD image with the `mkisofs` command:

```
mkisofs -o /tmp/cd.image /home/terry
```

NOTE You can also use `mkisofs` to send content to `stdout` and then feed directly into `cdrecord`. Using this method does run the risk of the output being larger than the CD capacity and possibly buffer underruns on slow systems that don't use burnproof or a similar technology. A good idea is to run the `du -s` command for each directory you want to back up to determine if it will fit on a CD/DVD.

This command makes a CD image file in the `/tmp` directory called `cd.image`. The CD image file contains all the files in the `/home/terry` directory. You must have enough space to make the image file on the partition holding the `/tmp` directory. You can determine how much is free with the `df` command. You can determine how much space the image file is going to take up with the `du /home/terry` command. By default, `mkisofs` preserves the ownership and permissions from the file system in the CD image.

To burn the image file to an actual CD, you must determine which device has the CD drive and its device number, logical unit number, and device ID. You can find this information with the following command:

```
cdrecord -scanbus
```

The output of the `cdrecord` command follows. The line that shows the CD drive type and manufacturer is the one you are interested in. At the beginning of the line is the SCSI ID information shown as 1,0,0, which is the SCSI device number (1), the logical unit number (0), and the device ID (0).

```
cdrecord-Clone 2.01-dvd (i686-pc-linux-gnu) Copyright (C) 1995-2004 Jörg
Schilling
Note: This version is an unofficial (modified) version with DVD support
Note: and therefore may have bugs that are not present in the original.
Note: Please send bug reports or support requests to
http://bugzilla.redhat.com/bugzilla
Note: The author of cdrecord should not be bothered with problems in this
version.
scsidev: 'ATA'
devname: 'ATA'
scsibus: -2 target: -2 lun: -2
Linux sg driver version: 3.5.27
Using libscg version 'schily-0.8'.
cdrecord: Warning: using inofficial libscg transport code version (schily - Red
Hat-scsi-linux-sg.c-1.83-RH '@(#)scsi-linux-sg.c      1.83 04/05/20 Copyright
1997 J. Schilling').
scsibus1:
    1,0,0    100) 'HL-DT-ST' 'RW/DVD GCC-4243N' '1.07' Removable CD-ROM
    1,1,0    101) *
    1,2,0    102) *
    1,3,0    103) *
    1,4,0    104) *
    1,5,0    105) *
    1,6,0    106) *
    1,7,0    107) *
```

You supply the SCSI device number, the device ID, and the logical unit number to the `cdrecord` command, in that order, as part of the `dev` option. A sample `cdrecord` command is as follows:

```
cdrecord -v dev=1,0,0 -data /tmp/cd.image
```

This command does not generally produce a bootable CD. For a CD to be bootable, the image file being recorded onto the CD needs to follow a specific format. Also, your BIOS must support booting from your particular CD-ROM. To produce a bootable image file, you need to follow several steps. First, you need to obtain a boot image. If you have a bootable CD-ROM in the disk drive, the boot image can be written to a file with the following command:

```
dd if=/dev/fd0 of=boot.img bs=10k count=144
```

TIP One handy boot image is the `diskboot.img`, which is on disk one of the Fedora Core and Red Hat Enterprise installation disks.

This command puts the boot image in the file `boot.img`. You must put this somewhere in the directory that you are going to put on the CD. In the example provided, you could create a directory `/home/terry/boot` and place the file there. You also need to give `mkisofs` some extra parameters to have it create a bootable image.

```
mkisofs -r -b /home/terry/boot/boot.img -c
/home/terry/boot/boot.catalog -o /tmp/cd.image /home/terry
```

The `boot.catalog` file need not already exist. It is generated by `mkisofs`. The command line option just tells `mkisofs` where in the image to store the generated file.

Using dump

The `dump` package consists of several commands for backing up and restoring the file system. The `dump` command is used to do backups of either entire partitions or individual directories of `ext2` and `ext3` file systems. The `restore` command is used to restore an entire partition, individual directories, or individual files.

Syntax of the dump Command

The first argument to the `dump` command is a list of options. Following that are all arguments required by the various options in the same order as the options were specified. The last argument is the file system to back up. Table 31-2 lists the available `dump` options.

Table 31-2 Dump Options

OPTION	MEANING	TYPE
B	The number of records per volume	Number
b	The number of kilobytes per dump record	Number
h	The dump level at which to use <code>nodump</code> flags	Number
f	Name of file or device to write to	Filename
d	Tape density	Number
n	Tell <code>dump</code> to send a message when done	None
s	Length of dump tape	Number in feet
u	Record the date of this dump in <code>/etc/dumpdates</code>	None
T	Add only files older than the given time	Time (ctime)

(continued)

Table 31-2 (continued)

OPTION	MEANING	TYPE
W	List the file systems that need to be backed up	None
w	List individual files that need to be backed up	None
0-9	Specify a dump level of 0 through 9	None

CAUTION Using `dump` on a live file system can be dangerous and unreliable.

For more information on this topic, see redhat.com/archives/ext3-users/2003-January/msg00034.html.

Sample dump Command

If you want to see a sample of the output from the `dump` command, try entering the command shown here:

```
dump 0uf /dev/nst0 /dev/hda3
```

This command specifies that the file system on `/dev/hda3` should be backed up on the magnetic tape on device `/dev/nst0`. It specifies that the backup should use backup level 0 (full backup) and write the time of the backup to the `/etc/dumpdates` file. The `/etc/dumpdates` file is used to keep track of the dates when your information was backed up.

Using restore

The `restore` command is used to retrieve files from the backups created with `dump`. You can use `restore` to restore an entire file system or you can use it to interactively select which files you want to restore.

The syntax for the `restore` command is the same as that for the `dump` command, although it has different options. Table 31-3 lists the options.

Table 31-3 Restore Options

OPTION	MEANING	TYPE
r	Restores the entire dump archive	None
C	Compares the files on the file system to those in the dump archive	None
R	Starts the restore from a particular tape in a multivolume sequence	None

Table 31-3 (continued)

OPTION	MEANING	TYPE
x	Extracts only specified files	List of files
t	Lists the contents of the dump archive	List of files
I	Restores files in interactive mode	None
b	Blocks size of the dump in kilobytes	Number
D	Names the file system to be compared against	File system
f	Names dump archive to restore from	Filename
h	Recreates directories but does not restore their contents	None
m	Extracts files by inode number instead of name	None
N	Prints filenames rather than extracting them	None
s	Specifies the tape number to start on when using the R option	Number
T	Specifies where to write temporary files	Directory
v	Specifies verbose mode	None
Y	Does not prompt when bad blocks are encountered	None

Restoring the File System

To restore a damaged or erased file system, you must first recreate the directory or partition that has been lost. If, for instance, you want to recreate the /home directory, which existed by itself on the /dev/hdb1 partition, you could use the following commands:

```
mkfs /dev/hdb1
mount /dev/hdb1 /home
```

Note that this command erases all of the data on the /dev/hdb1 partition. This method of restoration is useful only for restoring all of the files previously archived with dump. If any files have been added, modified, or deleted since the last backup, those changes are lost. Restoring individual files is covered in the section “Using Restore Interactively.” Also, if mkfs is accidentally run on a different partition than the one meant to be restored, all of the data on the partition on which it is mistakenly run is irrevocably erased.

The restore command must be run inside the directory that is going to be restored. So, restore can restore the /home directory with the following commands:


```
cd /home
restore rf /dev/nst0
```

The `r` flag tells `restore` to restore the entire archive rather than just some files. The `f` flag tells `restore` that the archive is located on the device `/dev/rft0`.

Using restore Interactively

The `restore` command, in addition to being used to restore an entire file system, can also be used in an interactive mode, which enables you to restore individual files. The interactive mode is invoked as follows:

```
restore if /dev/nst0
```

This command runs `restore` in interactive mode and specifies that it should restore from the archive on the device `/dev/rft0`. The interactive mode enables you to type options to restore and lets you control `restore`'s behavior. It includes the options shown in Table 31-4.

Table 31-4 Restore Commands

COMMAND	MEANING
<code>add</code>	Adds a file or directory to the list of files to be extracted. If a directory is specified, all contained files, subdirectories, and files contained in subdirectories are extracted. File paths are relative to the current directory being viewed in the dump archive.
<code>cd</code>	Changes which directory within the dump archive is being viewed.
<code>delete</code>	Removes a file or directory from the list of files to be extracted. If a directory is specified, all files in that directory, subdirectories, and files in subdirectories are removed from the list as well. Note that this does not affect what is stored in the dump archive, but rather which files are extracted during the restore.
<code>extract</code>	Extracts all files and directories currently in the list of files to extract and restores them in the file system.
<code>help</code>	Lists available commands.
<code>ls</code>	Lists the contents of the directory currently being viewed in the dump archive. If a directory is specified, the contents of the specified directory are listed rather than the contents of the current directory. Files and directories marked with <code>*</code> in the file listing are currently marked for extraction.
<code>pwd</code>	Prints the path within the dump archive of the directory currently being viewed.

Table 31-4 (continued)

COMMAND	MEANING
quit	Exits the restore program. No other actions are taken by restore.
setmodes	Rather than extracting the files, this sets the permissions on the files in the file system so that they match the permissions of the files in the dump archive that are marked for extraction.
verbose	Switches verbose mode on or off.

Using tar

Fedora Core and Red Hat Enterprise Linux include the GNU version of `tar`. It includes some extensions to the older standard versions of `tar`, including *multi-volume archiving*, an automated process in which `tar` prompts for new media to be inserted whenever it runs out of space. The `tar` program is a utility originally designed for making magnetic tape backups, but is useful for any kind of archiving purpose. When making archives, it is important to specify a leading `./` for files. That creates a relative path, which will be necessary when restoring the files later.

The `tar` command requires one command option followed by any number of optional options. Table 31-5 lists the command options.

Table 31-5 `tar` Options

COMMAND	EXPLANATION
A	Appends the contents of the given <code>tar</code> files to the specified tar archive.
d	Finds differences between what's in the tar archive and what's in the file system.
j	Filters the archive through the <code>bzip</code> filter. Used to compress or decompress files ending with <code>bz2</code> extension.
r	Appends the given files to the specified tar archive.
t	Lists the contents of the specified tar archive.
u	Appends the given files to the specified tar archive, but only if they are newer than the files in the tar archive.
x	Extracts the given files from the specified tar archive.
z	Filters the archive through the <code>gzip</code> filter. Used to compress or decompress files ending with <code>gz</code> extension.

In addition to specifying a command, you must specify a device or file to act as the destination of the tar archive.

Creating a tar Archive

The `tar` command was originally designed to create tape archives; in fact `tar` is a contraction of tape archive. But, you can create tar archives and save them anywhere on your system, not just to a tape drive. To create a tar file use the following command syntax.

```
tar cvf (name of tar file to create) (list of files to add)
```

If you wanted to create a tar file of the files `testing1.txt`, `testing2.txt`, and `testing3.txt` and place them into a tar archive called `testing.tar`, you would issue this command:

```
tar cvf testing.tar testing1.txt testing2.txt testing3.txt
```

In this example, you could use wildcards to make your job easier by using `testing*.txt` instead of typing the filenames individually.

You can also use `tar` to back up entire directories by issuing the directory name you want to tar instead of filenames. For example, if you wanted to tar a directory called `ch4` to the tape drive, you would issue this command:

```
tar cvf /dev/st0 ch4
```

This command would create a tar file called `ch4.tar` that would contain the contents of `ch4`, including all files as well as any subdirectories and rewind the tape to the beginning.

Extracting a Tar Archive

To extract a tar archive, use the following syntax:

```
tar x (name of tar file)
```

To extract the tar file created in the previous section, issue this command:

```
tar x testing.tar
```

This would extract the files from the tar file and place them in the current directory. If you extract a tar file containing a directory and subdirectories, if any, the directory and its subdirectories will be extracted into their original directory structure at the location from where the `tar` command was issued. For example, to extract the sample directory you created previously, you would type the following:

```
tar xvf /dev/st0
```

This command would extract the sample directory, and any files or subdirectories it contained, from the tape and place it at the location from where the `tar` command was run.

Advanced Tools

This section discusses a number of advanced backup tools, including AMANDA, the `amdump` test, and `pax`.

Using AMANDA

The Advanced Maryland Automatic Network Disk Archiver (AMANDA) package is a set of tools for doing backups of multiple machines over the network. Using AMANDA, you can configure your Red Hat Linux machine to be a centralized backup server for the other machines in the network, including Windows systems. AMANDA is included with Fedora Core and Red Hat Enterprise Linux. To use AMANDA, install the following packages:

- `amanda`
- `amanda-client`
- `amanda-server`
- `Gnuplot`

You need to install the `amanda-server` and `gnuplot` packages only on the machine that is going to be the backup server. However, you must install `amanda-client` on any machine that you want to back up using AMANDA. You must install the base `amanda` package on both the client and server machines. The `amanda` package contains several commands, shown in Table 31-6.

Table 31-6 AMANDA Commands

COMMAND	USE
<code>amdump</code>	Normally executed periodically by a cron job, this utility is run on the AMANDA server. It requests backups from the various AMANDA clients.
<code>amflush</code>	If <code>amdump</code> has trouble writing backups to tape, they are kept in temporary storage space on disk until the problem is corrected. After the problem is fixed, this command is run to write the data in the temporary storage space to the tapes.
<code>amcleanup</code>	If the AMANDA server crashes during the running of <code>amdump</code> , this utility should be run to clean up after the interrupted <code>amdump</code> .

(continued)

Table 31-6 (continued)

COMMAND	USE
<code>amrecover</code>	This utility provides a way to select which tapes should be used to recover files.
<code>amrestore</code>	This utility is used to restore individual files or directories or entire partitions from AMANDA backups.
<code>amlabel</code>	This utility is used to write an AMANDA label onto a tape. You must use this command to label tapes before they can be written to with <code>amdump</code> .
<code>amcheck</code>	This utility should be run before <code>amdump</code> to verify that the correct tape is in the drive.
<code>amadmin</code>	This utility does various administrative tasks.
<code>amtape</code>	This utility is used for low-level tape control, such as loading and ejecting disks.
<code>amverify</code>	This utility checks AMANDA tapes for errors.
<code>amrmtape</code>	This utility deletes a tape with a particular label from the AMANDA tape database.
<code>amstatus</code>	This utility reports on the current status of a running <code>amdump</code> program.

Installing AMANDA

After installing the necessary RPMs, some additional installation is required to get AMANDA running. You must create subdirectories in the `/etc/amanda` and `/usr/admn/amanda` directories for each backup schedule you are going to run. For instance, if you plan to run a backup schedule called `test`, you must execute the following commands:

```
mkdir -p /etc/amanda/test
mkdir -p /usr/admn/amanda/normal
```

You also need to create some temporary space for AMANDA to keep files, which it is in the process of backing up. So if, for instance, you want to create this space as a directory on your root partition, you can use the following command to make an `amanda` directory:

```
mkdir /amanda
```

Configuring AMANDA

To configure AMANDA, you must make changes to the `amanda.conf` file and put it in the subdirectory in `/etc/amanda` that you created. A sample `amanda.conf` file is created during AMANDA installation. So in the example, for instance, it would be called `/etc/amanda/test/amanda.conf`. The `amanda.conf` file has many options, shown in Table 31-7, but has defaults for most of them.

Table 31-7 `amanda.conf` Options

OPTION	EXAMPLE	MEANING
<code>org "name"</code>	<code>org "Tristero"</code>	This option specifies the name used in reports generated by AMANDA.
<code>mailto "accounts"</code>	<code>mailto "root example"</code>	This option specifies account names that Amanda should put in charge of the backup process.
<code>dumpuser "account"</code>	<code>dumpuser "Amanda"</code>	This option specifies the user account that the AMANDA dump process should run as.
<code>inparallel number</code>	<code>inparallel 5</code>	This entry specifies the number of <code>amdump</code> processes that can run simultaneously.
<code>netusage num unit</code>	<code>netusage 1000 Kpbs</code>	This entry indicates the bandwidth that AMANDA is allowed to consume while doing backups. It should be set such that even if all of the allocated bandwidth is consumed there is still enough bandwidth for other tasks that might operate at the same time as the AMANDA backup process.
<code>dumpcycle num unit</code>	<code>dumpcycle 1 week</code>	This option specifies the length of the backup cycle.
<code>runspcycle num</code>	<code>runspcycle 7</code>	This option specifies the number of backups that should be done during a single dump cycle. So, with a dump cycle of 1 week and seven runs per cycle, AMANDA makes one full backup and six incremental backups every week.
<code>tapespercycle num unit</code>	<code>tapespercycle 7 tapes</code>	This option specifies how many tapes are available for use in a single backup cycle.
<code>runtapes num</code>	<code>runtapes 1</code>	This option specifies how many tapes are available for use in each backup.
<code>tapedev "device"</code>	<code>tapedev "/dev/rft0"</code>	This option specifies the device name of the tape device.

The `amanda.conf` file also has some complex options, which consist of blocks with multiple subfields. The `holdingdisk` block defines a temporary storage space for holding data that is being backed up. You can define multiple `holdingdisk` blocks in a single file. The definition has the following format:

```
Holdingdisk name
{
    directory "name"
    use num unit
}
```

Example `holdingdisk` block:

```
Holdingdisk example
{
    directory "/example"
    use 4 Gb
}
```

The `tapetype` block defines a particular kind of magnetic tape that might be used in backups. It defines properties about the tape such as length and speed. The `tapetype` definition has the following format:

```
Define tapetype name
{
    comment "freeform string"
    length num unit
    filemark num unit
    speed num unit
}
```

Example `tapetype` definition:

```
Define tapetype EXAMPLE
{
    comment "These are fictional numbers."
    Length 5000 mbytes
    Filemark 100 kbytes
    Speed 500 kbytes
}
```

The `interface` block defines a particular network interface that can be used for communication between an Amanda server and client. The interface definition specifies how much bandwidth can be used on that particular network interface. The syntax of the definition is as follows:

```
Define interface name
{
  comment "Freeform string"
  use num unit
}
```

Example interface definition:

```
Define interface eth0
{
  comment "This sets the bandwidth usage of the Ethernet network
  interface"
  use 500 kbps
}
```

The `dumptype` block defines a particular kind of dump. The entries in the `disklist` file refer to these definitions. A corresponding `dumptype` block must exist in the `amanda.conf` file for it to be referenced in the `disklist` file. The `dumptype` block specifies certain properties of the kind of dump, such as which program to use for dumping, whether to compress backups, and which files should not be backed up.

The `dumptype` block has many options, shown in Table 31-8, which define how the dump works.

Table 31-8 `dumptype` Options

OPTION	EXPLANATION
<code>auth</code>	This option specifies which authorization method should be used between the client and the server. This option can be set to either <code>bsd</code> or <code>krb4</code> and defaults to <code>bsd</code> .
<code>comment</code>	This option is a freeform string and is ignored.
<code>comprate</code>	This option specifies the compression rates for backed up files in terms of how the size of the compressed file should compare to the size of the uncompressed file. This option can either be a single value or two values separated by a comma. The first value specifies the compression rate for full backups. The second value specifies the compression rate for incremental backups and is assumed to be the same as the first value if omitted.
<code>compress</code>	This option specifies the method to be used for compressing the data. The options are presented in the Table 31-9. The default compression type is "client fast."
<code>dumpcycle</code>	This option specifies the number of days in the backup cycle. A full backup is performed at the beginning of each backup cycle.

(continued)

Table 31-8 (continued)

OPTION	EXPLANATION
<code>exclude</code>	This option specifies which files should not be included in the backup. This option works only when the backup program being used is <code>tar</code> . When used with <code>dump</code> or <code>samba</code> it is ignored. The possible values for <code>exclude</code> are a quote wildcard pattern or the list keyword followed by a quoted filename. If the list keyword is used, the filename should refer to a file on the client machine, which contains a list of wildcard patterns to match. Wildcard patterns are listed one per line. Any files matched by either the quoted patterns or any of the patterns in the specified file are excluded from the AMANDA backups.
<code>holdingdisk</code>	This option specifies whether the holdingdisk should be used for temporarily storing files that are going to be dumped. The default is <code>yes</code> .
<code>ignore</code>	This option specifies that this dump type should not actually be backed up even if the <code>disklist</code> file specifies that it should.
<code>index</code>	This option specifies whether to keep an index of files that have been backed up. The default is <code>no</code> .
<code>kencrypt</code>	This option specifies whether the connection between the client and the server should be encrypted. The default is <code>no</code> .
<code>maxdumps</code>	This option specifies how many simultaneous instances of the <code>amdump</code> process can be run. The default is <code>1</code> .
<code>priority</code>	This option specifies the priority of the dump. When AMANDA runs out of tape or is otherwise unable to write backups for some reason, all the data that can be kept on the holdingdisk is put there in order of highest priority dump type to lowest priority. The possible values for the priority of a dump are <code>high</code> , <code>medium</code> , and <code>low</code> . The default is <code>medium</code> .
<code>program</code>	This option specifies which program should be used for making the backup dump. The possible values are <code>DUMP</code> and <code>GNUTAR</code> . The default is <code>DUMP</code> . You must change this to <code>GNUTAR</code> if you wish to use the <code>exclude</code> option.
<code>record</code>	This option specifies whether the date of the dump should be written to the <code>/etc/dumpdates</code> file. The default is <code>yes</code> .
<code>skip-full</code>	This option specifies that when Amanda is scheduled to do a full backup it should refrain from doing so. This option is useful if you want to use AMANDA for incremental backups or to use some other method for full backups.

Table 31-8 (continued)

OPTION	EXPLANATION
<code>skip-incr</code>	This option specifies that when AMANDA is scheduled to do an incremental backup it should refrain from doing so. This option is useful if you want to use AMANDA for full backups but to use some other method for incremental backups or if you do not want to do incremental backups at all.
<code>starttime</code>	This option specifies that the starting time of the dump should be delayed.
<code>strategy</code>	This option specifies the dumping strategy that should be used for this kind of dump. The various available dump strategies are listed in Table 31-10. The default strategy is <code>Standard</code> .

Table 31-9 AMANDA Compression Types

TYPE	EXPLANATION
<code>none</code>	This option specifies that no compression should be used on AMANDA backups.
<code>client best</code>	This option specifies that the client should use the compression algorithm that results in the highest compression levels.
<code>client fast</code>	This option specifies that the client should use the fastest compression algorithm.
<code>server best</code>	This option specifies that the server should use the compression algorithm that results in the highest compression levels.
<code>server fast</code>	This option specifies that the server should use the fastest compression algorithm.

Table 31-10 AMANDA Dumping Strategies

STRATEGY	EXPLANATION
<code>standard</code>	This option specifies that AMANDA should use the standard dumping strategy, which includes both full and incremental backups.
<code>nofull</code>	This option specifies that AMANDA should use level 1 incremental backups always and never do full backups. This is useful when a set of machines all have the same base installation and setup with only minor differences that do not change rapidly. Amanda then saves space by backing up only the changes that occur over time.

(continued)

Table 31-10 (continued)

STRATEGY	EXPLANATION
<code>noinc</code>	This option specifies that incremental backups should never occur and that AMANDA should always do full backups. This is useful if it is important to make the restoration of a machine as swift and easy as possible. However, it makes backups much slower and requires much more storage space for the backups.
<code>skip</code>	This option specifies that the dump type should never be backed up either with full backups or incremental backups. The dump type is ignored even if it occurs in the <code>disklist</code> file.

You need to adapt the `amanda.conf` file to your system. Most important, you need to correctly specify the paths to where you will be backing up the data, such as a disk archive or a tape drive devices, the type of tape drives, and the path to the directory that AMANDA can use as temporary space.

You must also create a `disklist` file that specifies which partitions to back up. In the example setup this file would be stored as `/etc/amanda/test/disklist`.

The format of the `disklist` file is a series of entries, one per line, in the following format:

```
Hostname device dumptype
```

The `disklist` file has the arguments shown in Table 31-11.
The following is an example `disklist` file:

```
Blanu.net /home/blanu/public_html normal
Tristero.sourceforge.net /cvsroot/tristero incremental
Baldwinpage.com /var/www/htdocs/bruno/ normal
```

Table 31-11 `disklist` Arguments

ARGUMENT	EXPLANATION
<code>hostname</code>	This argument specifies the hostname of the AMANDA client to be backed up. For the AMANDA client to enable a connection from the AMANDA server, the hostname of the AMANDA server must be in that client's <code>.amandahosts</code> file.
<code>device</code>	This argument specifies the name of the directory to be backed up.
<code>dumptype</code>	This argument specifies the name of the <code>dumptype</code> definition in the <code>amanda.conf</code> file, which defines the properties associated with this type of dump.

AMANDA Client Configuration

To enable the AMANDA backup servers to connect to the clients to request backups, you must create on each client an `.amandahosts` file in the `/root` directory of the machine. The file consists simply of the names of the server machines that are allowed to connect to the client to request backups.

Here is an example `.amandahosts` file:

```
Blanu.net
Thalassocracy.org
Tristero.sourceforge.net
Baldwinpage.com
```

You are wise to set the permissions of this file to 600 using `chmod`. That ensures that only root can modify the file and other users cannot add hosts to the file, thus bypassing the permission system and gaining access to the full file system.

Performing Backups with AMANDA

To perform a backup, you simply run `amdump` with the name of the backup that you want to run. The configuration information and list of partitions to back up are read from the configuration files in the particular subdirectory in `/etc/amanda` that you created for this particular backup type. In the examples in this section the test directory was created, so you run the command

```
amdump TEST
```

The `amdump` commands then go through the list of the partitions specified in `amdump` and back up each of them, in order, to the tape drives specified in the associated `amanda.conf` file. The partitions in the `disklist` file should be specified in order of importance so that in case of problems the most important files are more likely to have already been backed up. The results of the `amdump` operation, including any errors, are written to the `/usr/adm/Amanda/test` directory.

The `amdump` command is typically run as a cron job, so it can run as an automated process. If you wanted your backup to run every night at 1 a.m. you would need the following crontab entry.

```
0 1 * * 1-5 /usr/sbin/amdump test
```

Summary

In this chapter, you learned how to back up and restore your file system. You learned how to choose which files are important to back up and to choose a backup medium, a backup method, and a tape rotation schedule appropriate for the needs of your situation. You also learned how to use low-level archiving tools such as `tar` and `dump` to produce archives and file system data and to restore corrupted file system data from archives. In addition, you learned how to configure and use AMANDA, an advanced archiving tool.

Performance Monitoring

IN THIS CHAPTER

- System Performance Monitoring Tools
- Measuring Memory Usage
- Viewing Running Tasks
- Monitoring I/O Activity
- Using `sar`

This chapter describes some of the tools you can use to monitor the status and performance of your Fedora Core or RHEL system. Utilities like `free`, `top`, and `ps` provide basic information about the status of the system at given points in time. For ongoing monitoring, you would use tools like `iostat`, `vmstat`, and `sar`.

System-Performance-Monitoring Tools

The first group of tools this chapter discusses enables you to take snapshots of system performance at a given point in time. You can use this data to create baseline metrics of your system's performance. This historical data serves as a guide against which you measure the impact of changes you make. You can use a variety of tools, many more, in fact, than this chapter covers. The six you'll look at are listed here in alphabetical order:

- **`free`** — Reports the amount of free and used memory in the system
- **`iostat`** — Provides detailed CPU and I/O usage information
- **`sar`** — Collects, saves, or reports on a comprehensive list of system activity data

- **slabtop** — Reports kernel memory usage
- **top** — Displays a real-time list of running processes
- **vmstat** — Shows virtual memory and I/O system usage

One of the things you will notice is that each utility has some overlap with other utilities. For example, `free` and `vmstat` both report on virtual memory usage, although `vmstat` provides considerably more detail than does `free`. Likewise, `vmstat` and `iostat` can both provide I/O (input/output) usage data; again, `iostat`'s I/O analysis is more complete than `vmstat`'s is. The following sections disregard these areas of overlap and focus on what each utility does best. For instance, you won't see any discussion of `vmstat`'s disk I/O-specific features, nor will you read much about `iostat`'s ability to report on running processes (an area in which it overlaps with `top`).

Measuring Memory Usage

Even on systems that seem to have ample physical RAM, it is still a good idea to know how much memory is in use and how much is available. Excessive memory consumption, perhaps due to a memory leak in a running program, can slow a system down and eventually force a reboot to reclaim the “lost” memory. At the highest level, you can use `free` command to show a quick report of how much memory is in use and how much is free. `vmstat` shows more detail about memory usage, especially swap usage. The `slabtop` command shows you how the kernel itself is allocating memory.

Memory Usage as Seen by Users and Processes

You can use two commands to obtain summary information about the system's memory usage. The `free` command shows information about the amount of memory that is used and unused, including both physical RAM and swap space. `vmstat` shows the same information in greater detail.

`Free`'s syntax is:

```
free [-b|-k|-m] [-o] [-s secs] [-t]
```

Invoked without command line arguments, `free`'s output looks like the following:

```
$ free
```

	total	used	free	shared	buffers	cached
Mem:	515800	500652	15148	0	0	255356
-/+ buffers/cache:		245296	270504			
Swap:	1052248	536	1051712			

If you want the output to be displayed in bytes, rather than kilobytes, use the `-b` option; use `-m` to display the output in megabytes; `-k` displays the output in kilobytes, the default. If you're math challenged, the `-t` option adds a line to the bottom of the output showing totals values.

The `-o` option disables the `-/+ buffers/cached:` line, which shows adjustments made to the used and free physical RAM. These adjustments are necessary if you want to know how much RAM is actually in use and how much RAM the kernel has set aside for its own use. The kernel keeps a certain amount of RAM available for I/O and memory buffers to facilitate I/O. The amount of buffer memory varies over time as it is used and released. From the point of view of the system as a whole, RAM used as buffer memory is always "in use," even if the kernel has not allocated it at a given point in time. From the kernel's point of view, however, unused buffer memory is just that, unused (or free). Without the `-o` option, you see memory usage from the kernel's point of view. With the `-o` option, you can visualize memory consumption from the view of the system as a whole. In the `free` example just shown, just over 263 Mb (270,504 Kb) is "free" from the kernel's point of view. If you use the `-o` option, you won't see the amount of amount of RAM allocated as kernel buffer memory.

The other columns of output show the amount of memory allocated as shared memory (System V IPC shared memory, to be precise), additional non-specific buffer memory, and the amount of cached data in memory. The shared memory column should be disregarded because it is no longer used.

The final option that might prove useful is the `-s secs` option, which causes `free` to redisplay its report every `secs` seconds. The following example shows `free`'s output immediately before and during a kernel compilation:

```
$ free -s5 -m -o
```

	total	used	free	shared	buffers	cached
Mem:	503	496	7	0	0	253
Swap:	1027	0	1027			

	total	used	free	shared	buffers	cached
Mem:	503	495	8	0	0	253
Swap:	1027	0	1027			

	total	used	free	shared	buffers	cached
Mem:	503	494	8	0	0	243
Swap:	1027	0	1027			

	total	used	free	shared	buffers	cached
Mem:	503	489	14	0	0	244
Swap:	1027	0	1027			

This example used the `-m` option to display the output in megabytes, the `-o` option to turn off the buffer adjustment, and the `-s5` option to refresh the display

every five seconds. The kernel compilation started between the first and second updates. One of the features you'll notice in the bold-faced section is that the amount of cached data fell when the kernel build process started. Presumably, this occurred because data the kernel needed had to be read from disk, forcing a certain amount of cached data to be flushed.

`vmstat` digs deeper into memory usage than `free` and pays particular attention to virtual memory (swap) usage. If your system is constantly swapping, disk I/O will slow to a crawl and the system will seem slow to respond to user input. `vmstat` makes it possible for you to detect the problem. You can then use `top` or one of the other utilities discussed in this chapter to identify what is causing the excessive swapping. First, however, `vmstat`'s syntax, bearing in mind that this discussion ignores options not related to virtual memory:

```
vmstat [-S k|K|m|M] [-a] [-n] [secs [cnt]]
```

```
vmstat [-S k|K|m|M] -m
```

```
vmstat [-S k|K|m|M] -s
```

To change the display unit, which defaults to bytes, use `-S k` for units of 1000 bytes, `-S K` for true kilobytes (1024 bytes), `-S m` for units of 1,000,000 bytes, or `-S M` for true megabytes (1,048,576 bytes). The examples in the text use `-S K`. Certain `vmstat` reports can be refreshed every `secs` seconds and, if `cnt` is specified, will refresh `cnt` times every `secs` seconds before `vmstat` terminates. In its simplest usage, `vmstat`'s output looks like the following:

```
$ vmstat -S K
procs -----memory----- ---swap-- ----io---- --system-- ----cpu----
 r  b  swpd   free   buff  cache   si   so    bi    bo    in    cs us sy id wa
 2  0    852  16132      0 249232    0    0    33    26   10   82 94  2  4  0
```

This information shows only the average usage since the system was booted. To get information about current usage, you must request a refreshing display using `secs` and, if you multiple reports, `cnt`, as shown in the following example:

```
$ vmstat -S K 5 5
procs -----memory----- ---swap-- ----io---- --system-- ----cpu----
 r  b  swpd   free   buff  cache   si   so    bi    bo    in    cs us sy id wa
 3  0    852   4880      0 250120    0    0    33    26   15  135 93  3  4  0
 2  0    852  14628      0 250308    0    0     0    30 1132  405 94  6  0  0
 3  0    852  14168      0 250444    0    0     0    31 1131  418 93  7  0  0
 4  0    852   6780      0 250528    0    0     0    35 1130  375 94  6  0  0
 4  0    852  11856      0 247484    0    0     0   156 1149  422 93  7  0  0
```

What information is shown? In the `procs` section, the `r` column shows the number of processes that are ready to run and waiting for their turn to run on the CPU and the `b` column shows the number of processes that are blocked, or sleeping, and thus not ready to run. In the first example, therefore, two processes are ready to run and waiting for CPU time and no processes are blocked.

The four columns under the `memory` heading, show the following information:

- **swpd** — The amount of virtual memory in use
- **free** — The amount of physical RAM not in use
- **buff** — The amount of physical RAM used as buffers
- **cache** — The amount of physical RAM used as cache

As you can see in the second example, created during a kernel compile, the amount of free and cache memory fluctuates constantly; the more active the system, the greater the fluctuation.

If you specify `-a`, `inact` and `active` replace the `buff` and `cache` columns under the `memory` heading. `inact` displays the amount of inactive memory and `active` displays the amount of active memory. *Inactive memory* is the buffer memory the `free` command shows as free (unused) when buffer adjustments are enabled; *active memory* is memory that is both allocated and in use and maps to the used buffer memory reported by the `free` command. The following `vmstat` example shows the effect of the `-a` option:

```
$ vmstat -S K -a 5 5
procs -----memory----- --swap-- -----io----- --system-- -----cpu-----
 r  b   swpd   free  inact active   si   so    bi    bo   in    cs us sy id wa
 4  0   1500  13132 114240 346908    0    0    33    26   17   143 93  3  4  0
 3  0   1500  14412 114308 345512    0    0    71   116 1172   486 91  9  0  0
 4  0   1500  10704 114400 349152    0    0    22    36 1220   849 92  8  0  0
 4  0   1500  20944 114416 338936    0    0    10    82 1145   521 94  6  0  0
 3  0   1500  12240 114484 347584    0    0    13    49 1342  1437 92  8  0  0
```

Under the `swap` heading, `si` shows the amount of memory that has been read in from the swap device (or devices, if there are multiple swap files or partitions) and `so` the amount of memory that has been written out to a swap device. As you can see in the example just shown, swap usage on this system, even during a kernel compile, is negligible.

In the `io` section, the columns `bi` and `bo` show the number of disk blocks (in units of 1024 bytes) read from and written to, respectively, the system's block devices. Under the `system` heading, `in` lists the number of interrupts received per second and `cs` shows the number of context switches per second. Values under the `cpu` heading, finally, show the disposition of CPU usage, with each column expressed as a percentage of total CPU time (due to rounding, the values might not add to 100 percent). The specific columns are:

- **us** — The percentage of CPU time spent running user, or nonkernel, code
- **sy** — The percentage of time spent executing system, or kernel, code
- **id** — The percentage of CPU time that the CPU is idle
- **wa** — The percentage of CPU time spent waiting for I/O to complete

Examining Kernel Memory Usage

The memory usage information discussed so far examined memory from the point of view of the user or running processes. You haven't seen with any amount of detail how the kernel itself is using memory. The last `vmstat` option, `-m`, gives you a window into the kernel's internal memory usage. The `-m` option causes `vmstat` to display kernel slab usage. *Slabs* are caches of frequently used kernel memory objects, such as inodes, directory entries, file pointers, and random blocks of memory of specific sizes, such as 8192 bytes, 4096 bytes, and so on. Rather than use `vmstat` to view slab usage, however, you should use `slabtop`, which does for slabs what the `top` command does for processes, namely, show slab usage in a real-time updated format. `Slabtop`'s syntax is:

```
slabtop [-d secs] [-s sort] [-o]
```

`-d secs` specifies the number of seconds to pause between updates. `-o` tells `slabtop` to display its output once and then exit. The `-s sort` option sets the sort order, which defaults to the number of slab objects descending order, for the displayed slabs to sort. `sort` can be one of the values listed in Table 32-1.

Table 32-1 slabtop Sorting Criteria

CRITERIA	ORDER	DESCRIPTION
a	Ascending	Sort by the number of active objects
b	Ascending	Sort by the number of objects per slab
c	Descending	Sort by cache size
l	Descending	Sort by the number of slabs
v	Descending	Sort by the number of active slabs
n	Ascending	Sort by the slab name
o	Descending	Sort by the number of objects (this the default sort order)
p	Descending	Sort by the number of pages per slab
s	Descending	Sort by the object size
u	Descending	Sort by cache utilization

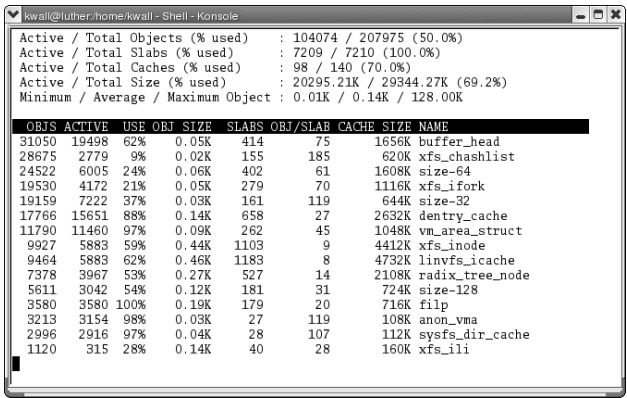


Figure 32-1 Viewing slabtop’s default output.

Invoked with no options, `slabtop`’s output resembles Figure 32-1. The slab cache listing, updated every three seconds by default, shows detailed slab cache information. The top five lines show summary information for the number of individual objects, the total number of slabs containing objects, the number of slab caches, and slab size statistics. The bottom portion of the display shows the specifics for each type of slab cache sorted in descending order by the object type. You can change the sort order at runtime by pressing the key associated with the sort criteria that interests you (see Table 32-1).

TIP The `p` sort option for sorting `slabtop`’s output by the number of pages per slab does not appear to function in `slabtop` version 3.2.3. However, you can view this information using the following `sort` invocation:

```
$ sort -k6,6 -nr < /proc/slabinfo | cut -f1 -d:
size-131072 (DMA)      0      0 131072    1 32
size-131072           0      0 131072    1 32
size-65536 (DMA)      0      0 65536     1 16
size-65536            4      4 65536     1 16
size-32768 (DMA)      0      0 32768     1  8
size-32768           49     49 32768     1  8
size-16384 (DMA)      0      0 16384     1  4
size-16384            3      3 16384     1  4
tcpv6_sock            1      5  1376     5  2
task_struct          120    120  1392     5  2
...
```

The sixth column shows the number of pages per slab. To view the output sorted in ascending order, omit `-r`:

```
$ sort -k6,6 -n < /proc/slabinfo | cut -f 1 -d:
# name                <active_objs> <num_objs> <objsize> <objperslab>
```

```
<pagesperslab>
slabinfo - version
anon_vma          3069   3213    32  119    1
arp_cache         2      20   192   20    1
as_arq            0      0    64   61    1
avc_node          12    600    52   75    1
bdev_cache        14     18   608    6    1
bio               287   287    96   41    1
biovec-1          293   452    16  226    1
biovec-16         260   260   192   20    1

...
```

Again, the sixth column shows the number of pages per slab.

If you run `slabtop` on a kernel that was compiled with the configuration option `CONFIG_DEBUG_SLAB` enabled, you will see additional slab cache statistics. The first line of the output will include (statistics) and the real-time display will show five additional columns:

- The maximum number of active objects in the slab
- The number of times objects have been allocated
- The number of times new pages have added to the cache (cache growth)
- The number of times unused pages have been removed from the cache (cache reaping)
- The number of errors allocating new pages to the cache

Unless you run a debugging kernel and are actively working on the kernel, you won't need this additional information. Nevertheless, you will at least know how to produce this information if someone asks for it.

Viewing Running Tasks

In many cases, you will be less concerned about how much memory a process is using and more concerned about what processes are running, or perhaps more likely, what processes are running out of control. The canonical tools for viewing running processes are `ps` and `top`. `ps` gives you a snapshot view of the currently active processes, and `top` gives you a real-time updated display of running processes.

Getting Started with ps

The implementation of `ps` that is used on Linux systems (from the `procps` suite) is a classic example of a Linux command gone horribly wrong. It has far too many options, a number of which are redundant. On the other hand, it lacks built-in email functionality, so `ps` might yet be salvageable. Seriously, `ps` is a powerful tool for viewing the current process list. Refer to the section titled “Obtaining Process Information” in Chapter 28 for discussions about additional process management programs.

Tables 32-2 through 32-5 borrow the layout of `ps`’s syntax description from its manual page and organize each group of options into tables based on the options’ purpose. `ps` supports both Unix98 options, which are preceded by a hyphen (-), and BSD options, which lack the initial -. Where the functionality is identical or very similar, the BSD options have been omitted. In some cases, apparently identical Unix98 and BSD are listed because the BSD option shows different output from the similarly invoked Unix98 option.

The options from this list that you’ll most likely use are `-e` to select all process, `r` to select only running processes, and `-N` to reverse the meaning of the selection criteria. `-N` helps you express selection criteria for which there are no command line options. For example, if you want to see all processes *except* those owned by the users `bubba` and `root` (see the `-u`, `-U`, and `U` options in Table 32-3), you might use the following command:

```
$ ps U bubba U root -N
  PID TTY          STAT TIME  COMMAND
 3947 ?            Ss   0:00  portmap
 4142 ?            Sls   0:00  ntpd -u ntp:ntp -p /var/run/ntpd.pid
 4179 ?            Ss   0:03  xfs -droppriv -daemon
 4217 ?            Ss   0:00  dbus-daemon-1 --system
```

Table 32-2 Basic Process Selection

OPTION	DESCRIPTION
<code>-N</code>	Negates the selection criteria specified with other options
<code>-a</code>	Selects all processes with a TTY except session leaders
<code>-d</code>	Selects all except session leaders
<code>-e</code>	Selects all processes
<code>T</code>	Selects all processes on the invoking terminal
<code>r</code>	Selects only running processes
<code>x</code>	Selects processes without controlling TTYs

By default, `ps` selects all processes with the same effective user ID (*euid*) as the current user and shows the PID, TTY, accumulated CPU time, and the program name. However, if your `ps` invocation includes BSD-style options (options that lack an initial `-`) the display will also include the process status and show the complete command line used, as shown in the previous example. If you replace the BSD-style `U` option with `-u`, the output looks slightly different:

```
$ ps -u bubba -u root -N
  PID TTY          TIME CMD
 3947 ?        00:00:00 portmap
 4142 ?        00:00:00 ntpd
 4179 ?        00:00:03 xfs
 4217 ?        00:00:00 dbus-daemon-1
```

This characteristic of `ps` can be aggravating, so if `ps` seems to refuse to display the output you want or expect, make sure you are not mixing Unix98 and BSD options.

Table 32-3 Process Selection

OPTION	DESCRIPTION
<code>-C command</code>	Selects by the command name matching pattern <i>command</i>
<code>-G rgid name</code>	Selects by real group ID (RGID) <i>rgid</i> or group <i>name</i>
<code>-p pid</code>	Selects by PID <i>pid</i>
<code>p pid</code>	Selects by PID <i>pid</i> and displays the program executed
<code>-U ruid name</code>	Selects by real user ID (RUID) <i>ruid</i> or user <i>name</i>
<code>-u euid name</code>	Selects by effective user ID (EUID) <i>euid</i> or user <i>name</i>
<code>U name</code>	Selects processes for user <i>name</i> and the program executed

Table 32-4 Standard Output Formats

OPTION	DESCRIPTION
<code>-f</code>	Displays full listing
<code>-j</code>	Displays output in jobs format
<code>j</code>	Displays output in job control format
<code>-l</code>	Displays output in long format
<code>l</code>	Display long output format
<code>s</code>	Displays output in signal format
<code>v</code>	Displays output in virtual memory format

The options for selecting output formats make it much easier for you to view the process information that you want. Here again, the distinction between the standard Unix98-style options and BSD-style options rears its ugly head. Notice in the following example the difference between the Unix98 `-j` option, which displays the selected processes in jobs format, and the BSD `j` option, which shows the selected processes in job control format:

```
$ ps -j
  PID  PGID   SID TTY          TIME CMD
12002 12002 12002 pts/2    00:00:00 bash
12111 12111 12002 pts/2    00:00:00 ps

$ ps j
  PPID   PID  PGID   SID TTY          TPGID STAT   UID   TIME COMMAND
11457 11458 11458 11458 pts/1    11992 Ss      500   0:00 /bin/bash
11458 11478 11478 11458 pts/1    11992 S       500   0:08 jpilot
11458 11992 11992 11458 pts/1    11992 S+      500   0:00 more ps.txt
12001 12002 12002 12002 pts/2    12113 Ss      500   0:00 /bin/bash
12002 12113 12113 12002 pts/2    12113 R+      500   0:00 ps j
```

The Unix98 format produced by `-j` is less informative than the BSD output format produced by `j`, consisting of only the PID, the process group ID (PGID), the session ID (SID), the TTY on which the program is running, the cumulative CPU time, and the bare command name. In many situations, the Unix98 output will be all you need. The BSD output is much more informative (at least in this case), showing all of the processes owned by the current user, not just the processes running on the current TTY (which is pts/2).

The difference or lack of difference between Unix98 and BSD `ps` options is even more apparent with the `-l` and `l` options, both of which display information in so-called long format:

```
$ ps -l
 F S   UID    PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
 0 S   500  12002 12001  0  76   0 -  1488 wait  pts/2    00:00:00 bash
 0 R   500  12112 12002  0  78   0 -   887 -    pts/2    00:00:00 ps

$ ps l
 F   UID    PID  PPID  PRI  NI   VSZ  RSS WCHAN  STAT TTY          TIME COMMAND
 0   500  11458 11457   15   0  4852 1492 wait  Ss  pts/1    0:00 /bin/bash
 0   500  11478 11458   16   0 17164 9044 -    S  pts/1    0:08 jpilot
 0   500  11992 11458   16   0  4284  620 -    S+ pts/1    0:00 more ps.txt
 0   500  12002 12001   17   0  5952 1452 wait  Ss  pts/2    0:00 /bin/bash
 0   500  12114 12002   19   0  2548  624 -    R+ pts/2    0:00 ps l
```

The chief difference in the two long format output styles is that, as before, the BSD-style output includes more information and the different values for process priorities. You can see that the bash process whose PID is 12001 has a priority of 76 in the Unix98 output format and 17 in the BSD format.

Table 32-5 Modifying Output Format

OPTION	DESCRIPTION
C	Uses raw CPU time for %CPU instead of a decaying average
c	Shows the true command name
e	Shows environment after the command
f	Displays process hierarchy as ASCII art (forest)
h	Suppresses header lines (repeats header lines in BSD personality)
-H	Shows process hierarchy (forest)
S	Includes some dead child process data (as a sum with the parent)
-w	Displays output in wide format
w	Displays output in wide format

The options for modifying the output format make it possible to do some very interesting things with `ps`. For example, to see a complete list of process in a nicely formatted ASCII art tree, try the following command:

```
$ ps f -ej
 5216  5048  5048 ?      S      0:00  \_ /bin/sh /usr/lib/firefox-1.0.3/firefo
 5253  5048  5048 ?      S      0:00  |  \_ /bin/sh /usr/lib/firefox-1.0.3/ru
 5258  5048  5048 ?      Sl   285:27 |      \_ /usr/lib/firefox-1.0.3/firefo
11457  5048  5048 ?      S      0:03  \_ konsole
11458 11458 11458 pts/1  Ss     0:00  |  \_ /bin/bash
11478 11478 11458 pts/1  S      1:58  |      \_ jpilot
14700 14700 11458 pts/1  S+     0:00  |      \_ less ps.txt
12001  5048  5048 ?      S      0:03  \_ konsole
12002 12002 12002 pts/2  Ss     0:00  |  \_ /bin/bash
14709 14709 12002 pts/2  R+     0:00  |      \_ ps f -ej
14575  5048  5048 ?      S      0:00  \_ kio_file [kdeinit] kio_file file /tmp
 5051  5048  5048 ?      S      0:00  dcoptserver --nosid
```

The value in viewing running processes in a tree format is that you can instantly see the parent-child relationships between processes without having to connect the dots using each process's PPID (parent PID). This also highlights another side effect of `ps`'s multiple personalities: there's usually more than one way to get identical or nearly identical results. For example, the following command shows another way to view a process tree:

```
$ ps axfj
 5048  5216  5048  5048 ?      -1 S      500   0:00  \_ /bin/sh /usr/lib/fi
 5216  5253  5048  5048 ?      -1 S      500   0:00  |  \_ /bin/sh /usr/li
 5253  5258  5048  5048 ?      -1 Sl   500 285:27 |      \_ /usr/lib/fi
 5048 11457  5048  5048 ?      -1 S      500   0:03  \_ konsole
```

```

11457 11458 11458 11458 pts/1    14700 Ss      500  0:00 | \_ /bin/bash
11458 11478 11478 11458 pts/1    14700 S      500  1:58 | \_ jpilot
11458 14700 14700 11458 pts/1    14700 S+     500  0:00 | \_ less ps.txt
5048 12001 5048 5048 ?          -1 S      500  0:03 \_ konsole
12001 12002 12002 12002 pts/2    14717 Ss      500  0:00 | \_ /bin/bash
12002 14717 14717 12002 pts/2    14717 R+     500  0:00 | \_ ps afjx
5048 14575 5048 5048 ?          -1 S      500  0:00 \_ kio_file [kdeinit]
5048 14713 5048 5048 ?          -1 S      500  0:00 \_ kio_pop3 [kdeinit]
1 5051 5048 5048 ?          -1 S      500  0:00 dcoptions --nosid

```

To get a sense of how processes are using memory, use the BSD-style `v` option, illustrated in the following example:

```

$ ps v
  PID TTY          STAT TIME  MAJFL  TRS   DRS  RSS %MEM COMMAND
11458 pts/1        Ss    0:00    6   577  4274 1400  0.2 /bin/bash
11478 pts/1        S      1:59   22  483 16944 7028  1.3 jpilot
12002 pts/2        Ss    0:00    2   577  5374 1480  0.2 /bin/bash
14700 pts/1        S+    0:00    3    96  3075  588  0.1 less ps.txt
14767 pts/2        R+    0:00    0    60  2943  624  0.1 ps v

```

By way of explanation, the columns that address memory usage are:

- **MAJFL** — Shows the number of major faults, which occur when data the CPU needs isn't resident in L1 or L2 cache and must be retrieved from main memory
- **TRS** — Shows the text resident size, the amount of memory used by a program's text segment, which contains initialized data
- **DRS** — Shows the disk resident size, the amount of physical RAM the process has consumed that is currently swapped to disk
- **RSS** — Shows the resident set size, the amount of physical RAM consumed that isn't swapped to disk
- **%MEM** — Shows the percentage of total physical RAM the process consumes

`ps` is feature-rich, embarrassingly so. It probably deserves a chapter of its own, but we have other utilities to cover. Time spent with the manual page and experimenting with `ps`, especially comparing the output of similar options, will be rewarded with a knowledge of `ps` (and your system's running processes) that will earn you serious geek points at the next Linux installfest you attend.

Using top

Although `ps` is amply capable of showing you what processes are running at a given point in time, `top` excels at showing how processes behave *over* time. Its user interface enables you to sort the output in a variety of ways. In addition,

you can use `top` to send signals to running processes and change their priorities. If you have a set of `top` options you always use, you can set these in a configuration file that `top` reads when it starts.

To get started, `top`'s basic syntax is:

```
top [-i] [-s] [-S] [-u user] [-d secs] [-n count] [-p pid]
```

See the `top` manual page for a complete list of all the command line options. The `-d secs` option sets the delay in seconds between each `top` update (the default is 3 seconds). `top` will update continuously in real time unless you specify `-n count`, which limits `top` to `count` updates before it exits. If you don't want to see idle processes in the listing, specify the `i` option. The `-s` option starts `top` in secure mode. This option is primarily used to prevent stray keystrokes from harming a running system. It only makes sense to use it when starting `top` as root because mortal users can affect only their own processes. If you want to see the CPU time consumed by all processes and their children, specify `-S` to see the time expressed as a cumulative sum. If you are interested in only the processes owned by a specific user, use the `-u user` option. To monitor a specific process or set of processes, use the `-p pid` option. To monitor multiples processes, you can specify multiple `pids` in a comma-separated list.

Figure 32-2 shows the default `top` display when no options are specified.

```

top - 23:15:33 up 10 days, 22 min, 5 users, load average: 1.45, 1.74, 1.51
Tasks: 131 total, 2 running, 129 sleeping, 0 stopped, 0 zombie
Cpu(s): 16.7% us, 5.7% sy, 3.7% ni, 73.7% id, 0.0% wa, 0.3% hi, 0.0% si
Mem: 515308k total, 502964k used, 12344k free, 0k buffers
Swap: 1052248k total, 34272k used, 1017976k free, 215560k cached

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM    TIME+  COMMAND
 3078 root        15   0 107m   51m 3876  S 15.6 10.3 564.27  X
 9718 kwall       25  10 41740 24m  11m  S  1.7  4.8 13:14.90 rhn-applet-gui
17973 kwall       15   0 111m   40m  18m  S  1.7  8.1 28:54.04 firefox-bin
 9702 kwall       15   0 34420 17m  13m  S  1.0  3.4  0:54.33 kwin
 9723 kwall       25  10 41736 24m  10m  S  1.0  4.8 13:00.11 rhn-applet-gui
 9724 kwall       25  10 41740 24m  11m  S  1.0  4.8 12:37.89 rhn-applet-gui
 9729 kwall       25  10 41748 24m  11m  S  1.0  4.8 12:48.87 rhn-applet-gui
 9707 kwall       15   0 37968 20m  15m  S  0.7  4.1  1:57.91 kicker
10073 kwall       15   0 62476 40m  26m  S  0.7  8.1  3:03.54 kmail
19196 kwall       15   0 36004 17m  13m  S  0.7  3.4  0:40.45 konsole
 660 kwall       15   0 36848 18m  13m  S  0.7  3.6  0:04.11 konsole
1789 kwall       15   0 6072 1408 1208  S  0.7  0.3  0:17.81 wget
2301 kwall       16   0 2016 1008 780  R  0.7  0.2  0:00.11 top
 9674 kwall       15   0 57188 19m  15m  S  0.3  3.8  2:58.64 kded
   1 root        16   0 1744  416 368  S  0.0  0.1  0:02.51 init
   2 root        34  19   0   0   0  S  0.0  0.0  0:33.17 ksoftirqd/0
   3 root        RT   0   0   0   0  S  0.0  0.0  0:00.00 watchdog/0
   4 root        10  -5   0   0   0  S  0.0  0.0  0:03.70 events/0
   5 root        14  -5   0   0   0  S  0.0  0.0  0:00.02 khelper
   6 root        10  -5   0   0   0  S  0.0  0.0  0:00.00 kthread

```

Figure 32-2 The default `top` display.

`top` packs a lot of information into a small amount of space. The screen is divided into two windows: the upper window, which shows summary information and the lower window, which shows process-specific information. The blank line at the bottom of the summary window is a status line and serves as a user input area for the few interactive commands that require user input. The summary window should look familiar because the first line shows `uptime`-style information while the third, fourth, and fifth lines show memory consumption à la the `free` command.

Top's content window is where the action is. In its default configuration, `top` shows the columns listed in Table 32-6.

By default, `top` sorts the process display in descending order of CPU (%CPU) usage. To change the sort field, press `O` to open the Current Sort Field dialog box and press the letter corresponding to the column by which you want the display sorted, and then press Enter. For example, to sort by memory usage, press `O n` Enter.

TIP To make running processes stand out, press `B` in the main `top` display to cause running processes to appear in bold face or press `b` to highlight the running processes.

Table 32-6 `top`'s Default Display Columns

COLUMN	DESCRIPTION
PID	The process ID
USER	The user name
PR	The process' priority
NI	The process' nice value
VIRT	The amount of virtual memory the process uses
RES	The amount of physical RAM the process consumes
SHR	The amount of shared memory the process has created
%CPU	The percentage of CPU time consumed by the process
%MEM	The percentage of total memory allocated to the process
TIME+	The total CPU time consumed by the process, not including CPU time consumed by child processes
COMMAND	The binary name that started the process

There are a number of other interactive commands you can use to modify `top`'s display while it is running. For newcomers, the most important keystroke is `q`, which exits `top`. Table 32-7 lists additional interactive keystrokes.

Table 32-7 `top` Interactive Keystrokes

KEYSTROKE	DESCRIPTION
<	Moves the sort column left one column
>	Moves the sort column right one column
b	Toggles bold display
c	Toggles display of command name
d	Sets the update interval
f	Adds or removes columns
i	Toggles display of idle tasks
k	Kills a task as specified by its PID
l	Toggles the load average summary
m	Toggles memory information summary
n	Sets the number of processes to display
o	Changes column display order
O	Selects the sort field
q	Quits <code>top</code>
R	Toggles normal or reverse sort
r	Renices a task as specified by its PID
S	Toggles display of cumulative CPU time
s	Sets the update interval
t	Toggles CPU statistics summary
u	Shows process owned by a specific user
W	Creates a <code>.toprc</code> file in the current user's home directory
x	Toggles highlighting the sort column
y	Toggles highlighting the running tasks
z	Toggles color display

The `W` command creates `$HOME/.toprc`, which `top` reads in subsequent sessions to restore your preferred settings. If you find `top`'s default display too cluttered, use the `i` command to toggle idle tasks off, which hides them. Press `i` again to restore idle processes to the display. Experiment with `top`'s display settings until you find a display that suits your needs, then save your preferences using the `W` command.

To kill a task using `top`:

1. In the main window, type `k`.
2. Type PID of the task you want to kill and press Enter.
3. Type the numeric or symbolic name of the signal you want to use (**15**, **SIGTERM**, is the default), and press Enter. Press Enter without typing a signal to accept the default signal.
4. `top` kills the corresponding process.

Notice that you can send any signal using `top`, not just a signal that terminates a process `SIGTERM`. Table 32-8 shows all of the signals that `top` understands. You can use either the numeric value, such as 15, or the symbolic name, `SIGTERM`. In addition, the symbolic name can be either the full name, such as `SIGHUP`, or the name without the `SIG` prefix, such as `HUP`.

Table 32-8 Signals Recognized by `top`

VALUE	NAME	VALUE	NAME	VALUE	NAME
1	SIGHUP	2	SIGINT	3	SIGQUIT
4	SIGILL	5	SIGTRAP	6	SIGABRT
7	SIGBUS	8	SIGFPE	9	SIGKILL
10	SIGUSR1	11	SIGSEGV	12	SIGUSR2
13	SIGPIPE	14	SIGALRM	15	SIGTERM
17	SIGCHLD	18	SIGCONT	19	SIGSTOP
20	SIGTSTP	21	SIGTTIN	22	SIGTTOU
23	SIGURG	24	SIGXCPU	25	SIGXFSZ
26	SIGVTALRM	27	SIGPROF	28	SIGWINCH
29	SIGIO	30	SIGPWR	31	SIGSYS

Monitoring I/O Activity

Memory and CPU usage serve as important indicators of your system's health and its overall performance and efficiency, but they are not the only measures. A common truism in performance-tuning literature and practice is that your system is only as fast as its slowest component. In most systems, the slowest component is the I/O subsystem. Memory, CPU, bus, and network speeds long ago surpassed the capability of disk devices (with the exception of certain solid state disks and "disklike" devices such as compact flash memory) to handle the data throughput possible with contemporary speeds. Accordingly, you need a good tool that helps you identify and isolate where I/O bottlenecks are occurring. `iostat`, discussed in this section, and `sar`, discussed in the next section, are just the tools to use. How you proceed *after* you've fingered the performance culprit is a different issue, of course; diagnostics identify only the problem.

NOTE In order to use the `iostat` command, the `sysstat` package must be installed. `vmstat`, on the other hand, is installed by the `procps` package. Confusing? You bet!

As its name suggests, `iostat` reports I/O statistics for disk devices and disk partitions. It also reports CPU performance data, but this information is not important to the present topic. `iostat`'s syntax is:

```
iostat [-d] [-k] [-t] [-x] [{dev|ALL}] [-p [{dev|ALL}]] [secs [cnt]]
```

Refer to the `iostat` manual page for complete syntax and usage details, especially for CPU utilization options not covered here. The `-d` option invokes `iostat`'s disk utilization mode; all the examples in this section use it because doing so disables CPU utilization reports. `iostat`'s default output should resemble the following:

```
$ iostat -d
Linux 2.6.10-1.770_FC3.root (beast.example.com)      05/05/2005

Device:            tps    Blk_read/s    Blk_wrtn/s    Blk_read    Blk_wrtn
hda                 1.30         11.46         9.79      3996171     3415664
hdb                 0.46          2.99         4.06      1043976     1416023
hdd                 0.00          0.00         0.00         680          0
```

The first report `iostat` prints lists summary data collected since the last boot. To get current information, use the `secs` and/or `cnt` options. `secs` specifies the delay between updates and `cnt`, if specified, specifies how many

updates to show before `iostat` exists. If `cnt` is not specified, the updates are continuous. The following example takes two samples five seconds apart:

```
$ iostat -d 5 2
```

Device:	tps	Blk_read/s	Blk_wrtn/s	Blk_read	Blk_wrtn
hda	1.30	11.46	9.80	4001643	3423981
hdb	0.46	3.01	4.29	1052328	1499821
hdd	0.00	0.00	0.00	1208	0

Device:	tps	Blk_read/s	Blk_wrtn/s	Blk_read	Blk_wrtn
hda	0.20	0.00	3.20	0	16
hdb	1.20	0.00	22.40	0	112
hdd	0.00	0.00	0.00	0	0

You'll learn what `iostat`'s other options do in a moment. What exactly does `iostat`'s output mean?

- **Device** — Indicates the disk device or partition.
- **tps** — Shows the number of transfers per second sent to the device. A *transfer* in this context refers to any I/O request and has no specific size because multiple read or write requests can be contained in a single transfer.
- **Blk_read/s** — Shows the number of 512-byte blocks read per second from the device.
- **Blk_wrtn/s** — Shows the number of 512-byte blocks written per second to the device.
- **Blk_read** — Shows the total number of 512-byte blocks read from the device.
- **Blk_wrtn** — Shows the total number of 512-byte blocks written to the device.

Thus, since the system booted, the second `iostat` command shows the following for `/dev/hda`:

- The average number of transfers per second is 1.2.
- The average number of blocks read per second is 11.46, or approximately 5867 bytes per second.
- The average number of blocks written per second is 9.80, or approximately 5018 bytes per second.
- The total number of blocks read is 4,001,643, which totals 2,048,841,216 bytes or about 1.9 GB.
- The total number of blocks written is 3,423,981, which totals 1,753,078,272 bytes or about 1.6 GB.

The block size `iostat` uses is shown in terms of disk sectors. Thus, when you are analyzing a physical disk, blocks are 512 bytes, but when you are looking at a disk partition, the measurement appears in kilobytes (1024 bytes). Both values are independent of a file system's block size, which might be 512 bytes, 1024 bytes, or some other value.

To see the values in kilobytes (KB) rather than bytes, use the `-k` option:

```
$ iostat -d -k
Linux 2.6.10-1.770_FC3.root (beast.example.com)      05/05/2005

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
hda                 1.29         5.72         4.90    2008233    1720898
hdb                 0.46         1.50         2.15     528420     755286
hdd                 0.00         0.00         0.00         604         0
```

TIP In many cases, you can merge command line options to reduce typing.

For example, `iostat -d -k` and `iostat -dk` are equivalent.

While I/O statistics for an entire disk are useful, it might be more helpful to be able see I/O statistics on a per-partition basis. To obtain this level of granularity, use the `-p dev` option, where `dev` is the device in which you are interested. For example, to see the I/O statistics for all of the partitions on `/dev/hda`, use the following command:

```
$ iostat -dk -p /dev/hda
Linux 2.6.10-1.770_FC3.root (beast.example.com)      05/05/2005

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
hda                 1.29         5.71         4.90    2008777    1723297
hda3                 1.36         5.61         4.35    1974399    1531225
hda2                 0.16         0.09         0.55     33064     191752
hda1                 0.00         0.00         0.00     1230       320
```

As you can see, of the three partitions on `/dev/hda`, `hda1` (the `/boot` file system) has hardly been used at all, as you would expect, and `hda3` has been the most heavily used, which also make sense because it contains the `/` file system.

But wait! There's more! If you specify `-x` (which you can't use with `-p`), `iostat` shows extended statistics for each disk (but not for each partition):

```
$ iostat -dk -x
Linux 2.6.10-1.770_FC3.root (beast.example.com)      05/05/2005

Device:  rrqm/s wrqm/s  r/s   w/s  rsec/s wsec/s   rkB/s   wkB/s avgrq-sz
avgqu-sz  await  svctm  %util
hda       0.03   0.21   0.71  0.58  11.40   9.79    5.70    4.90   16.42
```

0.01	10.40	4.46	0.58							
hdb		0.01	0.11	0.17	0.30	3.01	4.31	1.51	2.16	15.80
0.01	11.18	2.32	0.11							
hdd		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	100.67
0.00	638.17	638.17	0.00							

Even in standard 80-character wide terminal the display wraps because it is too long. For each disk, the extended statistics include:

- **rrqm/s** — The average number of read requests merged and sent to the disk per second
- **wrqm/s** — The average number of write requests merged and sent to the disk per second
- **r/s** — The average number of read requests sent to the disk per second
- **w/s** — The average number of write requests sent to the disk per second
- **rsec/s** — The average number of sectors read from the disk per second
- **wsec/s** — The average number of sectors written to the disk per second
- **rkB/s** — The average number kilobytes read from the disk per second
- **wkB/s** — The average number of kilobytes written to the disk per second
- **avgrq-sz** — The average size in sectors of requests sent to the disk
- **avgqu-sz** — The average queue length of requests sent to the disk
- **await** — The average time in milliseconds that elapsed between issuing an I/O request and the request completing
- **svctm** — The average time in milliseconds spent servicing a request
- **%util** — The percentage of CPU time during which I/O requests were being sent to the disk

These statistics show you the most interesting and useful I/O statistics because they show you how well your I/O subsystem is performing and identify I/O bottlenecks. For example, the more read and write requests that are merged (see **rrqm/s**, **wrqm/s**, **r/s**, and **w/s**), the more efficient the I/O subsystem is because read and write requests are being batched together and filled all at once. On the other hand, if too many requests are being merged, I/O *throughput*, the amount of data actually being read from or written to disk, can decrease. Throughput is measured by sectors read and/or written per second (**rsec/s** and **wsec/s**) and by the actual byte count of reads and writes per second (**rkB/s**, and **wkB/s**).

To get a sense for the overall task load, look at the wait time (**await**) and CPU utilization values (**%util**). If **await** is high, the system might be I/O bound because too much time elapses between when an I/O request is issued

and when that request completes. Similarly, if the utilization value is high, the system as a whole is saturated because I/O operations represent a (dis)proportionately high amount of the system's activities, either performing actually I/O or waiting for I/O to complete.

Using sar

All the tools discussed so far have been single-purpose utilities that spotlight specific areas of the performance and monitoring spectrum: `free`, `vmstat`, and `slabtop` for memory consumption; `ps` and `top` for CPU usage; and `iostat` for disk utilization. Although many administrators prefer such single-purpose tools, a certain utility and convenience exists in being able to access all of a system's performance metrics using one program. This section discusses `sar` (pronounced like "car"), which stands for system activity report. `sar` is a popular tool that provides a single interface for collecting, storing, and analyzing system monitoring and performance data.

`sar` and `sadc` are, like `iostat`, installed with the `sysstat` package. Unfortunately, `sysstat` is not installed as part of some of the default Fedora Core installation profiles, notably the Fedora Core workstation profile. If you want to use `sar`, make sure the `sysstat` package is installed.

On Fedora Core and RHEL system, `sar` works in conjunction with the `sadc` program that is started at boot time by the `sysstat` service. `sadc`, which stands for system activity data collector, samples a wide variety of system data every 10 minutes and writes it in binary format to `/var/log/sa/sadd`, where `dd` is a two-digit zero-padded value corresponding to the current day of the month. `sar` reads the data files created by `sadc` and uses that data to create its reports. The virtue of this approach is that you can review `sar`'s reports for a given day on any system and at any time, provided you have access to the raw data files.

NOTE A quick `ls` in `/var/log/sa` shows files named `sardd`. These files are daily reports created by the `sa2` command, which is a shell command that invokes `sar` using the `-f` option. See `/usr/lib/sa/sa2` for more details.

Because `sar` can be used to monitor multiple subsystems, the text looks at each subsystem separately. Before diving in to the subsystem-specific modes, however, you'll want to know some of the general command line options that `sar` supports. Probably the most useful option is `-f ifile`, which lets you tell `sar` the input file it should read. (`ifile` is one of the binary-formatted files described in the previous paragraph.) This is the option to use if you want to see statistics for a day other than the current one. If you want to take a fresh

sample of data and save it to an output file, use the `-o ofile` option, replacing `ofile` with the name of the file you want. Finally, to force `sar` to use current data rather than read stored data, place an interval value (in seconds) and a count value at the end of the command line. For example, the following command displays CPU usage every 10 seconds, exiting after the fifth update:

```
$ sar 10 5
Linux 2.6.10.779_FC3.root (beast.example.com)      05/08/2005

10:13:00 PM      CPU      %user      %nice      %system      %iowait      %idle
10:13:10 PM      all       57.10       26.10       16.80         0.00         0.00
10:13:20 PM      all       59.34       23.38       17.28         0.00         0.00
10:13:30 PM      all       57.96       23.52       18.52         0.00         0.00
10:13:40 PM      all       56.04       25.87       18.08         0.00         0.00
10:13:50 PM      all       57.86       26.53       15.62         0.00         0.00
Average:         all       57.66       25.08       17.26         0.00         0.00
```

To save this output in a text file, use the following command:

```
$ sar -o cpu.rpt 10 5
```

Monitoring Memory with sar

You can use `sar` to monitor physical and virtual memory consumption several different ways. Invoke `sar -B` to see paging; run `sar -r` option to review RAM usage; execute `sar -W` option to view swap activity. In this case, paging refers to memory that the CPU has had to page in from disk. Here's a sample report:

```
$ sar -B 5 5
Linux 2.6.10-1.770_FC3.root (beast.example.com)    05/08/2005

10:26:49 PM  pgpgin/s  pgpgout/s  fault/s  majflt/s
10:26:54 PM    170.40   1257.20   5136.00    0.00
10:26:59 PM    121.04   1921.84   4886.17    0.00
10:27:04 PM    111.20   1015.60   5352.00    0.00
10:27:09 PM    151.20   2305.60   4623.00    0.00
10:27:14 PM    129.60   7243.40   1854.80    0.00
Average:       136.69   2749.06   4370.19    0.00
```

The columns displayed include:

- **pgpgin/s** — The number of kilobytes paged in from disk per second
- **pgpgout/s** — The number of kilobytes paged out to disk per second

- **fault/s** — The number of page faults, which occur when the CPU reads data that is not in an active memory page
- **majflt/s** — The number of major page faults, which occur when the CPU reads a memory page that must be loaded from disk

The `-r` option is a great replacement, if slightly harder to read, for the `free` command because it shows you with less ambiguity how your system's RAM and swap space is being used. Its output might resemble the following:

```
$ sar -r 5 5
Linux 2.6.10-1.770_FC3.root (beast.example.com)      05/08/2005

10:27:32 PM kbmemfree kbmemused  %memused kbbuffers  kbcached kbswpfree
kbswpused  %swpused  kbswpcad
10:27:37 PM      5936    509864     98.85         0    241324    1051524
724      0.07        144
10:27:42 PM     13120    502680     97.46         0    243484    1051524
724      0.07        144
10:27:47 PM      9732    506068     98.11         0    241992    1051524
724      0.07        144
10:27:52 PM      5768    510032     98.88         0    232348    1051524
724      0.07        144
10:27:57 PM     25692    490108     95.02         0    234284    1051524
724      0.07        144
Average:      12050    503750     97.66         0    238686    1051524
724      0.07        144
```

- **kbmemfree** — The amount of free (unallocated) RAM, in kilobytes
- **kbmemused** — The amount of used RAM, in kilobytes
- **%memused** — The percentage of total RAM in use
- **kbbuffers** — The amount of physical RAM the kernel is using as buffer space, in kilobytes
- **kbcached** — The amount of physical RAM the kernel is using to cache data, in kilobytes
- **kbswpfree** — The amount of unused swap space, in kilobytes
- **kbswpused** — The amount of used swap space, in kilobytes
- **%swpused** — The percentage of total swap used in use
- **kbswpcad** — The amount of cache memory in use by memory pages that are *also* still available from swap, in kilobytes

Monitoring CPU Usage with sar

sar provides several options for viewing CPU utilization data. The `-q` option shows run queue data on a per second basis; the `-u` option shows traditional CPU usage data; and `-w` shows the number of context switches per second. Context switches occur each time a CPU stops executing one process, saves the process's state, and starts executing another process. The more context switches the CPU has performed, the more work it is doing (well, duh). For example, the following output was created on a system running the GCC test suite and compiling the kernel:

```
$ sar -w 5 5
Linux 2.6.10-1.770_FC3.root (luther.kurtwerks.com)      05/09/2005

10:53:30 PM    cswch/s
10:53:35 PM      539.00
10:53:40 PM      947.40
10:53:45 PM     1085.40
10:53:50 PM     3245.78
10:53:55 PM     7778.24
Average:         2720.77
```

Notice how the number of context switches varies widely.

The run queue statistics show a different and more accurate picture of CPU activity:

```
$ sar -q 5 5
Linux 2.6.10-1.770_FC3.root (luther.kurtwerks.com)      05/09/2005

10:56:10 PM    runq-sz  plist-sz    ldavg-1    ldavg-5    ldavg-15
10:56:15 PM         11      142        8.84        6.80        4.34
10:56:20 PM         9      145        8.85        6.84        4.36
10:56:25 PM        11      144        8.86        6.88        4.39
10:56:30 PM        10      143        9.19        6.98        4.43
10:56:35 PM        10      143        9.18        7.01        4.46
Average:         10      143        8.98        6.90        4.40
```

What are you looking at?

- **runq-sz** — Displays the number of processes waiting for CPU time
- **plist-sz** — Displays the total number of processes in the process list
- **ldavg-1** — Displays the load average during the past minute
- **ldavg-5** — Displays the load average during the last 5 minutes
- **ldavg-15** — Displays the load average during the last 15 minutes

The *load average* is the number of processes that are ready to run but that cannot because they are waiting for CPU time. You can see in the example output that the run queue is fairly steady at an average of 10 processes waiting to run of a total of 143 processes. However, the load averages are gradually increasing, indicating that overall system usage is high and constant, rather than spiking temporarily.

The `-u` option, finally, shows you what kind of code the CPU is executing and how many processes are waiting for I/O to complete:

```
$ sar -u 5 5
Linux 2.6.10-1.770_FC3.root (luther.kurtwerks.com)      05/09/2005

11:05:14 PM      CPU      %user      %nice      %system      %iowait      %idle
11:05:19 PM      all      65.40      17.00      17.60         0.00         0.00
11:05:24 PM      all      67.47      17.17      15.37         0.00         0.00
11:05:29 PM      all      67.74      17.03      15.23         0.00         0.00
11:05:34 PM      all      64.20      17.00      18.80         0.00         0.00
11:05:39 PM      all      63.20      15.00      21.80         0.00         0.00
Average:         all      65.60      16.64      17.76         0.00         0.00
```

In a multiprocessor system, the data would be displayed for all CPUs individually.

- **%user** — The percentage of CPU time spent running user, or nonkernel, code (application-level code)
- **%nice** — The percentage of CPU time spent running user-level code with priorities modified using the `nice` (or `renice`) commands
- **%system** — The percentage of time spent executing system, or kernel, code
- **%iowait** — The percentage of CPU time that the CPU was idle while waiting for I/O to complete
- **%idle** — The percentage of CPU time that the CPU was idle and not waiting for I/O to complete

To summarize, `sar` is the Swiss army knife of system monitoring utilities. Unlike the other programs and command discussed in this chapter, `sar` usually displays reports using data collected and stored in the file system. As a result, `sar`'s impact on system performance is minimal. If you run it as shown in this section, specifying an interval between updates and/or the number of updates to display, expect a modest performance hit.

Summary

As a system administrator, one of your primary responsibilities is to maintain a responsive, smoothly running system for users. When performance begins to falter, you need to have tools at your disposal to isolate, identify, and diagnose the problem. `free`, `vmstat`, and `slabtop` give you excellent visibility into the use and disposition of user- and kernel-level memory stores. `ps` and `top` are excellent tools for visualizing what processes are running and for isolating processes that might be the root of your system's performance problem. `iostat`, similarly, makes it easy to see whether and where I/O bottlenecks are hampering performance. If you prefer to use a single tool and don't want to impact your system's performance while monitoring it, `sar` combines the capabilities of all these utilities into a single program and reads historical data collected by a data collection daemon.

PART

Five

System Security and Problem Solving

Chapter 33: Exploring SELinux Security

Chapter 34: Implementing Network Security

Chapter 35: Troubleshooting and Problem Solving

Exploring SELinux Security

IN THIS CHAPTER

- Understanding SELinux
- Using SELinux
- Finding More Information about SELinux

This chapter is a primer on Security-Enhanced Linux, or SELinux. SELinux is a new and much stronger security model for Fedora Core- and RHEL-based systems. Although it is most appropriate for use in situations that demand strict security, SELinux can also be used in server and desktop environments. SELinux is also significantly different from traditional Linux and UNIX security models, so it takes some time to change certain assumptions about system access and to unlearn certain practices (mostly finger habits) that are incompatible with the security model that SELinux enforces. The material in this chapter is introductory and intended to familiarize you with SELinux and the mandatory access control model on which it is built, and to show you how you can get started using SELinux features.

Understanding SELinux

SELinux is a set of patches to the Linux kernel and key supporting utilities that implement mandatory access control and role-based access control. Created by the United States National Security Agency (NSA) as part of ongoing research into information security, mandatory access control and role-based access control make it possible to implement strict information separation at the kernel level. The purpose is to maintain and enforce confidentiality and data integrity

on multiuser systems by giving the kernel the ability to enforce access control based on policy. One common scenario in which mandatory access control and role-based access control can be used occurs when you need to restrict root's omnipotence. Policy-based access control, enforced in the kernel, makes it possible to prevent root from accessing certain files, for example, while still maintaining full ability to administer the system. Another common use for access control is to limit the damage done by flawed or malicious code. A typical exploit method is to create a buffer overflow in an application running with root privileges that results in executing arbitrary code, such as command shell. While SELinux cannot prevent the buffer overflow, it *can* limit the damage caused by unauthorized access to a root shell.

Even though SELinux was created by the NSA, which is responsible for the United States' signal intelligence activities, it is freely available under the same terms and conditions as Linux itself, the General Public License. Initially released on 22 December 2000 for the 2.2.12 kernel, SELinux has constantly been improved and kept up to date with developments in the mainstream kernel. Surprisingly, SELinux has been accepted by the Linux community and incorporated into most of the major Linux distributions, including Fedora Core, RHEL, SUSE Linux (now known as Novell Enterprise Desktop), Debian, and a version of Gentoo Linux known as Hardened Gentoo).

NOTE SELinux patches were not present in RHEL 3, only RHEL 4.

Mandatory and Role-Based Access Control

The standard Linux security model is referred to as discretionary access control, or DAC, because file access and other resource-related decisions are made based on user identity and a relatively small set of object permissions (read, write, and execute for a given user, a given group or groups, and/or other users). In a DAC-based security model, therefore, programs executed by a specific user have extremely wide discretion to do (practically) anything to that user's resources. Moreover, the root user and programs running as the root user have almost total discretion over *all* resources owned by *all* users. The implication for flawed or malicious software is clear: effectively complete control over files and other resources it controls via the user identity associated with the program. In a less hostile security environment, DAC provides an acceptable level of security against unintentional or accidental damage and a reasonably robust defense against casual attempts to exploit it. DAC also relies on the good will and common sense of users to exercise their powers responsibly, assuming, for example, that a user won't make the company payroll files readable by all users.

Mandatory access control, or MAC, makes no assumptions about users' responsibilities but does assume a hostile security environment. MAC gives security and system administrators full control over all of the resources on a system and, if properly implemented, prevents users from making risky security-related decisions. MAC works by defining a policy and then enforcing that policy on objects on the system. To take a simple example, one policy might be, "No file can be made world-readable." To enforce this policy, users would be prevented from setting the world read bit even on files they own and programmatic attempts (using the `chown` system call, for example) to make world-readable files would also be denied. Even the root user would be subject to this restriction. Policy enforcement happens at the kernel level and is dependent *not* on the user's identity alone but on any security-related information deemed relevant.

SELinux's MAC implementation makes it possible to create fine-grained permissions for both subjects and objects. *Subjects* include users, programs, processes, and threads, that is, for any entity or actor. *Objects* refer to files and devices, or more generally, anything to which you can do something or on which you can act.

Role-based access control (RBAC) is a specific type of MAC. RBAC works by grouping subjects into abstract roles. Broadly understood, a *role* consists of a discrete set of actions that a subject having that role is permitted to perform. The policy enforcement occurs when a role is cross-referenced against a list or table of objects on which the role can act. This cross-referencing is known as type enforcement because policy enforcement depends on the type of process (or subject) that seeks to perform some action and the type of object on which the action would be performed. Each process type, known as a *domain*, has a carefully defined set of objects on which it can act and an equally carefully defined set of actions that it can take. Admittedly complicated, type enforcement allows for fine-grained control over the users, programs, and processes on a Fedora Core or RHEL system.

WHY DID THE NSA CHOOSE LINUX?

Because they can. Seriously, the NSA moved their information security research to Linux for (at least) five reasons:

- 1. Previous research had been conducted on niche or research operating systems that didn't reflect real-world usage.**
- 2. They weren't seeing the results they wanted.**
- 3. The National Security Council was interested in Linux.**
- 4. The NSA was interested in mutual technology transfer opportunities.**
- 5. Linux was perceived to be the best alternative.**

SELinux Policies

Traditional MAC implementations have suffered from a key shortcoming that limited its popularity and usability: inflexibility. That is, traditional MAC implementations have been able to enforce only a single access policy. SELinux introduced the notion of multiple MAC policies and allowed administrators to define their own policies. The original SELinux policy suffered from a similar shortcoming and the resulting NSA policy was too strict. SELinux's developers eventually created *two* policies: the strict policy (contained in the Fedora Core and RHEL package `selinux-policy-strict`) that was consistent with the original and restrictive NSA guidelines, and the targeted policy (contained in the Fedora Core and RHEL package `selinux-policy-targeted`). The key difference between the strict and the targeted policy is that the targeted policy focuses on restricting and securing a carefully defined set of daemons whose compromise or failure would result in significant integrity, stability, or security compromise. The balance of the policy allows normal usage under traditional Linux security, that is, as if SELinux is not enabled. For Fedora Core 4, the targeted policy covers 80 daemons, which is too long a list to include here, so please refer to the Fedora Core or RHEL release notes for a complete list of the daemons included in the targeted policy.

Using SELinux

By default, SELinux as configured on Fedora Core and RHEL systems is uses the targeted policy and is enabled by default when the system boots. If you disabled SELinux during installation, you can reenale it using the Security Level Configuration tool. To do so, type **system-config-securitylevel** in a terminal window or other command prompt or select Red Hat ⇄ System Settings ⇄ Security Level. Figure 33-1 shows the Security Level Configuration dialog box.

To enable SELinux, click the SELinux tab. See Figure 33-2.

Check the Enabled (Modification Requires Reboot) check box. If the Policy Type isn't targeted, use the drop-down box to select the targeted policy. Click the Relabel on next reboot to relabel files controlled by SELinux appropriately. Click OK to make your change take effect. Finally, reboot.

CAUTION Relabeling the file system can take a very long time.

Changing the SELinux policy on a running system is one of the few situations in which you need to restart a running Linux system. Changing the SELinux policy requires a reboot because restarting the system under a new policy allows all system processes to be started in the proper context and reveals any problems in the policy change.

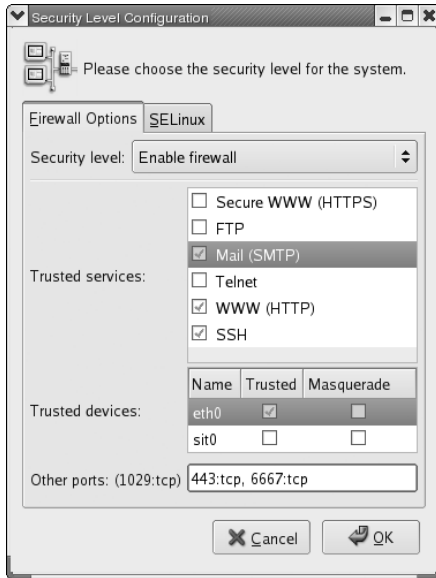


Figure 33-1 The Security Level Configuration tool.

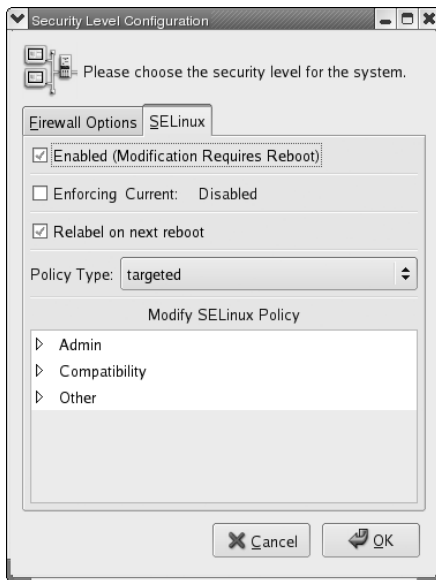


Figure 33-2 The SELinux tab.

After rebooting, use the `sestatus` command to ensure that your changes take effect:

```
# sestatus
SELinux status:      enabled
SELinuxfs mount:     /selinux
Current mode:        permissive
Mode from config file: permissive
Policy version:      18
Policy from config file:targeted

Policy booleans:
allow_yppbind         active
dhcpd_disable_trans   inactive
httpd_disable_trans   inactive
httpd_enable_cgi      active
httpd_enable_homedirs active
httpd_ssi_exec        active
httpd_tty_comm        inactive
httpd_unified         active
mysqld_disable_trans  inactive
named_disable_trans   inactive
named_write_master_zonesinactive
nscd_disable_trans    inactive
ntpd_disable_trans    inactive
portmap_disable_trans inactive
postgresql_disable_transinactive
snmpd_disable_trans   inactive
squid_disable_trans   inactive
syslogd_disable_trans inactive
use_nfs_home_dirs     inactive
use_samba_home_dirs   inactive
use_syslogng          inactive
winbind_disable_trans inactive
ypbind_disable_trans  inactive
```

The first few half dozen lines of output indicate the general status of SELinux. The output you see should be identical, with the exception of the Policy version line, which might be different on your system. The output under the Policy booleans heading shows the daemons and domains (types) for which SELinux policy is enabled. Type enforcement is enabled on policies labeled active and disabled on policies labeled inactive. As you can see in the output in the previous listing, the targeted policy is quite limited in what it protects.

Check the system log file, `/var/log/messages`, for messages that read `avc: denied` (that's two spaces between `avc:` and `denied`). Such messages indicate potential problems that might interfere with smooth system usage:

```
# grep 'avc: denied' /var/log/messages
May 16 22:46:59 luther kernel: audit(1116298019.278:0): avc: denied { read }
for pid=3993 exe=/sbin/portmap name=libnsl.so.1 dev=hda3 ino=111865608 scontext=
user_u:system_r:portmap_t tcontext=system_u:object_r:file_t tclass=lnk_file
May 16 22:46:59 luther kernel: audit(1116298019.285:0): avc: denied { read }
for pid=3993 exe=/sbin/portmap name=libnsl-2.3.5.so dev=hda3 ino=109095436 scont
ext=user_u:system_r:portmap_t tcontext=system_u:object_r:file_t tclass=file
May 16 22:46:59 luther kernel: audit(1116298019.314:0): avc: denied { getattr }
for pid=3993 exe=/sbin/portmap path=/lib/libnsl-2.3.5.so dev=hda3 ino=1090954
36 scontext=user_u:system_r:portmap_t tcontext=system_u:object_r:file_t tclass=
file
May 16 22:46:59 luther kernel: audit(1116298019.323:0): avc: denied { execute }
for pid=3993 path=/lib/libnsl-2.3.5.so dev=hda3 ino=109095436 scontext=user_u
:system_r:portmap_t tcontext=system_u:object_r:file_t tclass=file
May 16 22:47:09 luther kernel: audit(1116298029.443:0): avc: denied { read }
for pid=4172 exe=/usr/sbin/ntpd name=libcap.so.1.10 dev=hda3 ino=109636543
scontext=user_u:system_r:ntpd_t tcontext=system_u:object_r:file_t tclass=file
May 16 22:47:09 luther kernel: audit(1116298029.482:0): avc: denied { getattr
} for pid=4172 exe=/usr/sbin/ntpd path=/lib/libcap.so.1.10 dev=hda3 ino=1096
36543 scontext=user_u:system_r:ntpd_t tcontext=system_u:object_r:file_t tclass=
file
May 16 22:47:09 luther kernel: audit(1116298029.492:0): avc: denied { execute }
for pid=4172 path=/lib/libcap.so.1.10 dev=hda3 ino=109636543 scontext=
user_u:system_r:ntpd_t tcontext=system_u:object_r:file_t tclass=file
May 16 22:47:09 luther kernel: audit(1116298029.503:0): avc: denied { read }
for pid=4172 exe=/usr/sbin/ntpd name=libc.so.6 dev=hda3 ino=51207831 scontext
=user_u:system_r:ntpd_t tcontext=system_u:object_r:file_t tclass=lnk_file
May 16 22:47:09 luther kernel: audit(1116298029.515:0): avc: denied { write }
for pid=4172 exe=/usr/sbin/ntpd name=log dev=tmpfs ino=8273 scontext=user_u:
system_r:ntpd_t tcontext=user_u:object_r:device_t tclass=sock_file
May 16 22:47:09 luther kernel: audit(1116298029.527:0): avc: denied { sendto }
for pid=4172 exe=/usr/sbin/ntpd path=/dev/log scontext=user_u:system_r:ntpd
_t tcontext=user_u:system_r:unconfined_t tclass=unix_dgram_socket
```

Messages that read `avc: denied` mean that the SELinux policy in effect is not allowing the named executable, such as `/usr/sbin/ntpd` or `/sbin/portmap`, to take some action. The denied action appears in braces. For example, consider the following `avc: denied` message:

```
May 16 22:47:09 luther kernel: audit(1116298029.527:0): avc: denied { sendto }
for pid=4172 exe=/usr/sbin/ntpd path=/dev/log scontext=user_u:system_r:ntpd
_t tcontext=user_u:system_r:unconfined_t tclass=unix_dgram_socket
```

In this message, the `/usr/sbin/ntpd` program (`exe=/usr/sbin/ntpd`) is attempting to use the `sendto` system call (`{ sendto }`) to write to the device-special file `/dev/log` (`path=/dev/log`), but SELinux is denying permission to do so. The specific `ntpd` process in question is process 4172 (`pid=4172`).

Use the system for a while, attempting especially to exercise the daemon executables (shown in boldface in the listing). If you discover a feature on

which you rely is not working properly, refer to the section titled “Modifying the Targeted Policy” to learn how to disable policy enforcement. After you have satisfied yourself that everything is working correctly, start the Security Level Configuration tool again, and click the Enforcing check box to start enforcing the new security level. (See Figure 33-3.)

Alternatively, you can use the command `setenforce 1` or `setenforce Enforcing` at a command prompt:

```
# setenforce 1
```

This command enables SELinux policy enforcement. You can use the `getenforce` command to see the current enforcement status:

```
# getenforce
Enforcing
```

If you change your mind, use the `setenforce` command with an argument of 0 or `Permissive` to disable policy enforcement:

```
# setenforce Permissive
# getenforce
Permissive
```

Although not strictly required, it might be wise to reboot the system again.

Enabling SELinux Manually

To enable SELinux manually, use the following steps:

1. Edit `/etc/selinux/config` and change the policy type to read as follows:

```
SELINUXTYPE=targeted
```

2. While editing `/etc/selinux/config`, make certain that the `SELINUX` variable is set to `permissive`, that is:

```
SELINUX=permissive
```

This line starts SELinux with the correct policy but still enables you to log in if there is an unanticipated problem.

3. Execute the following command to force the initialization scripts to relabel the system when it reboots:

```
# touch /.autorelabel
```

Relabeling the system refers to setting certain extended file system attributes that SELinux uses to enforce policy. These extended file attributes are an important part of SELinux’s protection (and a significant source of trouble if the attributes are incorrectly set).

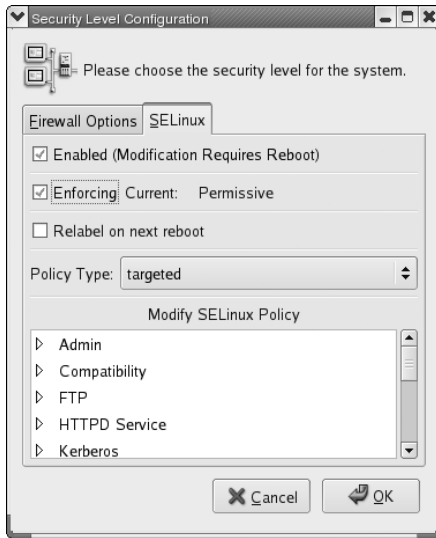


Figure 33-3 Enabling enforcement of the SELinux targeted security policy.

4. Verify that the changes took effect with the `sestatus` command:


```
# sestatus -v
```
5. While the system is still running in permissive mode, check `/var/log/messages` for `avc: denied` messages to identify possible problems that might interfere with smooth system usage under the new policy.
6. When you are satisfied that the system runs in a stable manner under the new policy, enable enforcing by changing the `SELINUX` line in `/etc/selinux/config` to `SELINUX=enforcing`. You can either reboot or execute **setenforce 1** to turn enforcing on in real time.

Modifying the Targeted Policy

To modify the targeted SELinux policy, use the Security Level Configuration tool. One reason to modify SELinux policy is to eliminate the `avc: denied` messages described earlier. These messages indicate that programs are violating (rather, *attempting* to violate) some aspect of the targeted policy. For example, the following code shows that the `ntpd` program is triggering policy violations:

```
# grep 'avc: denied' /var/log/messages | grep ntpdate
May 16 22:47:09 luther kernel: audit(1116298029.443:0): avc: denied { read }
for pid=4172 exe=/usr/sbin/ntpd name=libcap.so.1.10 dev=hda3 ino=109636543
scontext=user_u:system_r:ntpd_t tcontext=system_u:object_r:file_t tclass=file
May 16 22:47:09 luther kernel: audit(1116298029.482:0): avc: denied { getattr
} for pid=4172 exe=/usr/sbin/ntpd path=/lib/libcap.so.1.10 dev=hda3 ino=1096
```

```

36543 scontext=user_u:system_r:ntpd_t tcontext=system_u:object_r:file_t tclass=
file
May 16 22:47:09 luther kernel: audit(1116298029.503:0): avc: denied { read }
for pid=4172 exe=/usr/sbin/ntpd name=libc.so.6 dev=hda3 ino=51207831 scontext
=user_u:system_r:ntpd_t tcontext=system_u:object_r:file_t tclass=lnk_file
May 16 22:47:09 luther kernel: audit(1116298029.515:0): avc: denied { write }
for pid=4172 exe=/usr/sbin/ntpd name=log dev=tmpfs ino=8273 scontext=user_u:
system_r:ntpd_t tcontext=user_u:object_r:device_t tclass=sock_file
May 16 22:47:09 luther kernel: audit(1116298029.527:0): avc: denied { sendto }
for pid=4172 exe=/usr/sbin/ntpd path=/dev/log scontext=user_u:system_r:ntpd
_t tcontext=user_u:system_r:unconfined_t tclass=unix_dgram_socket

```

To get rid of these messages and fix `ntpd`, start the Security Level Configuration tool and click the SELinux tab. As you can see in Figure 33-4, a number of policy options are present in the Modify SELinux Policy box.

Expand the SELinux Service Protection item by clicking the triangle next to that item in the list. Find the Disable SELinux protection for `ntpd` daemon item and click the check mark next to it. (See Figure 33-5.)

By disabling the SELinux policy for the NTP daemon, you will eliminate the policy violations that appear in your system log. You are also preventing SELinux from protecting NTP against compromise, so proceed cautiously. Click OK to save your changes. You can make similar changes to a variety of other daemons protected by SELinux and in doing so will be creating your own customized version of the targeted security policy.

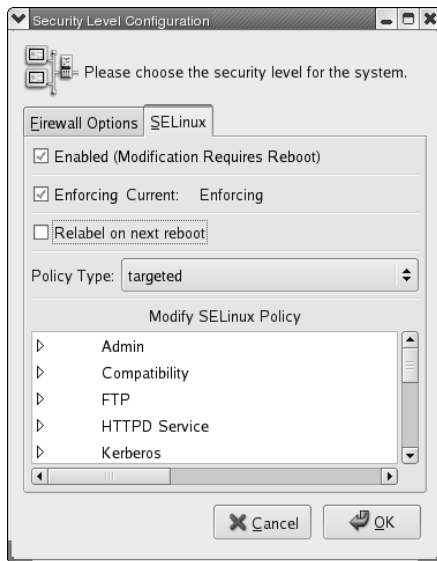


Figure 33-4 Viewing editable SELinux security policies.

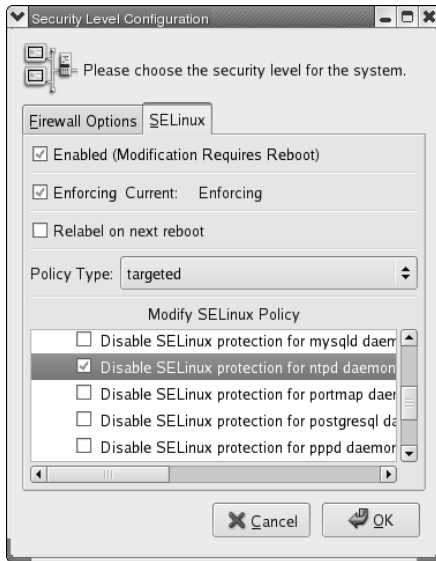


Figure 33-5 Viewing editable SELinux security policies.

Finding More Information about SELinux

This chapter has only scratched the surface of SELinux. For additional information about the SELinux extensions, the following sources of information will prove useful:

- NSA Resources
 - *NSA SELinux Information* — nsa.gov/selinux
 - *NSA SELinux FAQ* — nsa.gov/selinux/info/faq.cfm
- Fedora Project and Red Hat Resources
 - *The Fedora Core SELinux FAQ* — <http://fedora.redhat.com/docs/selinux-faq>
 - *Running Apache on SELinux* — <http://fedora.redhat.com/docs/selinux-apache-fc3>
 - *Fedora Core SELinux Mailing List* — <https://listman.redhat.com/mailman/listinfo/fedora-selinux-list>

- Community Resources

- *SELinux community page* — <http://selinux.sourceforge.net>
- *UnOfficial FAQ* — crypt.gen.nz/selinux/faq.html
- *Writing SE Linux policy HOWTO* — https://sourceforge.net/docman/display_doc.php?docid=21959&group_id=21266
- *Getting Started with SE Linux HOWTO* — https://sourceforge.net/docman/display_doc.php?docid=20372&group_id=21266
- *IRC Channel* — <irc://irc.freenode.net#fedora-selinux>

Summary

This chapter introduced you to SELinux, an extremely robust and complex security enhancement to Linux contributed by the National Security Agency. SELinux represents a significant departure from the traditional Linux security model, so it is important to understand not only the weaknesses of the traditional model that SELinux tries to address but also how SELinux uses mandatory access control, role-based access control, and type enforcement to increase the level of data integrity and information assurance possible on Linux systems. Because SELinux is complicated, the Security Level Configuration tool is a real bonus because the tool makes modifying the policies possible without having to learn yet another arcane configuration language. You will also find it helpful to know where you can find more information about SELinux configuration and usage.

Implementing Network Security

IN THIS CHAPTER

- Creating a Firewall
- Installing, Configuring, and Using LDAP
- Installing, Configuring, and Using Kerberos

It's a dangerous Internet out there. Perhaps the most important task and the most daunting challenge confronting any system administrator is to protect systems from external compromise. This chapter shows you how to use `iptables` to create and maintain a firewall on your Fedora Core system to (help) keep the bad guys out while permitting legitimate network traffic into your LAN. You will also learn how to use LDAP and Kerberos to create a more scalable and secure authentication (login) environment.

Creating a Firewall

Firewall is a generic term used to describe methods for selectively permitting or denying access to a network or a server. As generally used, a firewall is software that controls access at the packet level by examining the source and destination IP addresses, including the port number. In some cases, the actual content (or *payload*) of the IP packets will be examined to limit the type of traffic permitted to enter a network (referred to as *ingress control*) or to exit a network (referred to as *egress control*). Firewalls are not necessarily strictly software-based. A number of commercial firewall solutions consist of a dedicated hardware device (sometimes referred to as an *appliance*) running a security-hardened Linux distribution that includes the firewall software and

application software for configuring the firewall and controlling the appliance itself. Regardless of the specific details, however, the goal is the same: to protect a network by controlling the type of traffic that enters it.

The firewall solution this chapter covers is Netfilter, the Linux kernel's built-in packet-filtering software. It is often referred to as iptables because the command used to edit the packet filtering rules is named `iptables`. The standard kernels provided in Fedora Core and RHEL include a full suite of loadable kernel modules that provide a complete packet-filtering toolkit. Fedora Core and RHEL also include an easy-to-use tool for configuring Netfilter rules, the Security Level Configuration tool. To get started, select Main ⇄ System Settings ⇄ Security Level or type **system-config-securitylevel** in a terminal window. The main screen should resemble Figure 34-1.

The firewall might already be enabled depending on the options you chose when you installed Fedora Core or RHEL. If it isn't, select Enable firewall from the Security level drop-down box. When you enable the firewall, you can select the inbound traffic you want to permit for the Trusted services list. The list of trusted services contains common services for which you might wish to enable ingress:

- **WWW (HTTP)** — Web server access
- **FTP** — FTP server access
- **SSH** — SSH-based shell account access
- **Telnet** — Telnet-based shell account access
- **Mail (SMTP)** — Mail server access

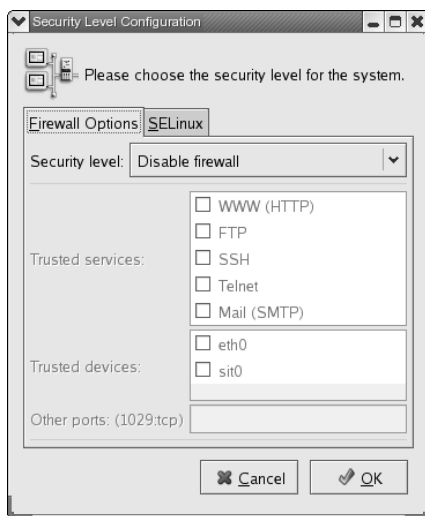


Figure 34-1 The Security Level Configuration tool.

The smart way to approach access control is to use the principle referred to in security circles as *least-access*. This simply means starting from a system to which no access is granted and then to permit the minimum amount of access necessary for a system to serve its purpose. For example, if you have a Web server, you don't need to enable access to the mail server, the FTP server, or the SSH server. The idea behind the principle of least-access is to limit the methods that ne'er-do-wells can attempt to exploit to gain unauthorized access to your system. If you administer your Web server using SSH, naturally, you will need to enable SSH access. In the case of a Web server, it isn't necessary to permit FTP access even if visitors will be downloading files directly from the server using their browser because the Web server will serve the files using its built-in support for the FTP protocol. Figure 34-2 shows the firewall configured to permit WWW (HTTP) and SSH access.

Notice that Figure 34-2 shows the network interface `eth0` as a trusted device. A *trusted device* is a network interface through which traffic to the host is permitted. This is most useful on multihomed hosts, that is, hosts that have multiple network interface cards. For example, Internet gateways often have two Ethernet cards, say, `eth0` and `eth1`. One (suppose it is `eth0`) is Internet-facing and is the interface on which inbound Internet traffic arrives. The other (`eth1` in this case) is usually LAN-facing and is the interface by which LAN clients access the Internet. The gateway will use network address translation (NAT) and IP masquerading to route traffic from the LAN to the Internet (from `eth1` to `eth0`). In this configuration, the firewall would consider `eth0` as a trusted interface and route (carefully controlled) traffic on to `eth1`. However, direct connections to `eth1` from outside of the LAN (that is, from the Internet) would be denied.

In the Other ports text box, add the port number and protocol family of services not listed in the Trusted services list to which you want to permit access. The format of these entries is `port:family`. Multiple entries must be separated by commas. For example, if you are running an IRC server, you have to allow TCP and UDP access to port 6667, so you would enter `6667:tcp 6667:udp` in the Other ports text box. (See Figure 34-3.)

TIP To find out what ports and protocols a given network service uses, you can look at the `/etc/services` configuration file. For example, if you want to know what ports and protocols the IRC daemon, `ircd`, requires, the following `grep` command will show you:

```
$ grep -i ircd /etc/services
ircd          6667/tcp      # Internet Relay Chat
ircd          6667/udp      # Internet Relay Chat
```

After you have configured access to services and trusted devices, click OK to save your changes and exit the Security Level Configuration tool. Click Yes when prompted. (See Figure 34-4.)

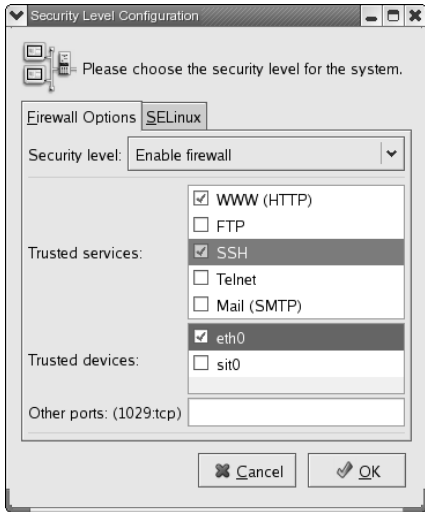


Figure 34-2 Configuring the firewall to allow HTTP and SSH access.

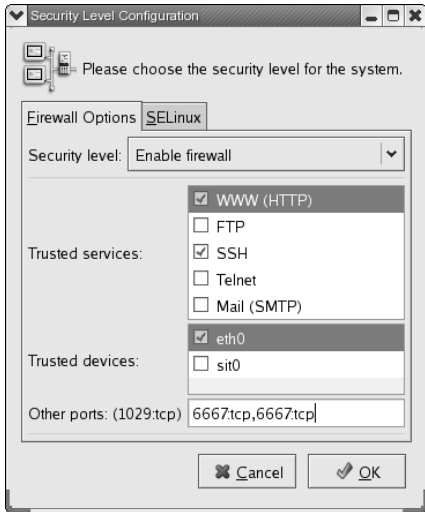


Figure 34-3 Specifying additional port access.

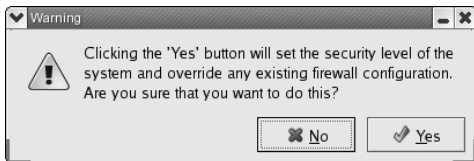


Figure 34-4 Confirming firewall configuration changes.

ABOUT NAT AND IP MASQUERADING

Network address translation, or NAT, maps an IP address used on one network to a different IP address used on another network. As used in this chapter, NAT translates IP addresses used inside a LAN to an IP address of a system that faces the Internet. For incoming traffic destined to a specific IP address inside the LAN, such as the HTTP responses sent to a client browser, NAT unmaps the global IP addresses on incoming packets back to local IP addresses. NAT is used for two main reasons: to conserve and to limit the number of external IP addresses a company needs and to enhance network security. NAT limits the number of externally facing IP addresses a company needs because it directs all Internet-destined traffic through a single “public” IP address. NAT strengthens network security because each outgoing or incoming IP package must pass through a NAT step. The translation step offers the opportunity to qualify or authenticate the request or match it to a previous request.

IP masquerading is a specific type of NAT that permits multiple hosts on a private network (a network using an IP address range that is not accessible across the Internet) to use the Internet via a single IP address that is accessible to the Internet. It is referred to as masquerading because all of the internal IP addresses appear to be a single IP, that of the host providing the externally visible IP address.

To activate your changes, load the new configuration by restarting iptables using the service utility:

```
# service iptables restart
Flushing firewall rules: [ OK ]
Setting chains to policy ACCEPT: filter [ OK ]
Unloading iptables modules: [ OK ]
Applying iptables firewall rules: [ OK ]
```

Once your firewall is up and running, you can test by attempting to access services to which access is denied.

Installing, Configuring, and Using LDAP

LDAP, the Lightweight Directory Access Protocol, provides a hierarchical collection of information that can be accessed over a network. LDAP is an example of a directory service, where the term *directory* refers to a central information resource such as a telephone directory or network-accessible address book. LDAP directories can be thought of as simple, hierarchical databases that are accessed using keys that identify the portions of the directory hierarchy to traverse to locate a specific unit of information. LDAP is analogous to other directory services that are in common use on networked systems today,

such as Novell's NDS and eDirectory, Netscape's Directory Server, or Sun's SunONE Directory Server software.

LDAP is based on the X.500 standard for directory sharing (www.uth.tmc.edu/uth_databases/white_pages/rfc-index.html), but is designed to provide an open, usable model for sharing information over a network that is unencumbered by the complexity of X.500 specifications and implementations. LDAP is referred to as a lightweight directory access protocol because it eliminates the overhead of the seven-layer Open System Interconnect (OSI) network stack model used by the X.500 specification in favor of using the TCP/IP layer directly. LDAP also does not support many of the obscure, rarely used features of the X.500 specification, which simplifies implementation and provides an easily understandable API. For additional introductory information about LDAP and X.500, see <http://asg.web.cmu.edu/cyrus/email/standards-X500.html>.

Like all directory services, LDAP is a client-server technology. Clients can either query or upload information to an LDAP server. In the case of a query, an LDAP server either responds to a query directly or can forward the query to another LDAP server, which repeats the "respond or forward" process. The first LDAP server for Linux systems was developed at the University of Michigan, but most Linux LDAP development now takes place in the OpenLDAP project (openldap.org), which is the source of the software discussed in this section.

Using LDAP provides a number of potential benefits to an organization. It provides a central, network-accessible repository for commonly used information. Its hierarchical nature makes it easy to search for and locate specific information, while also providing a directory that holds many different types of information. The idea of directory elements and attributes is easy to understand and work within light of conceptually similar models for managing information such as Extensible Markup Language (XML). The LDAP protocol is also independent of the underlying storage model used, which makes it easy to map LDAP data into existing databases or migrate to new, smaller database models.

Overview of LDAP Directory Organization

LDAP directories consist of hierarchical entries with specific attributes that each define one particular bit of information about that entry. Each entry can be viewed as an object that contains a number of attribute type/value pairs, much like an element in an XML document or data description. Each entry is defined by its distinguished name (DN), which is a unique identifier for that node in an LDAP directory. In addition to the distinguished name, LDAP nodes also have a relative distinguished name (RDN), which is a unique identifier for that node within the context of its parent. A good analogy for DNs and RDNs is full and relative pathnames in a file system hierarchy. For example,

the distinguished name for the Linux `less` utility would be `/usr/bin/less`, its full pathname. However, its relative distinguished name within the context of the `/usr/bin` directory is `less`, which is enough to uniquely identify it in that context.

The set of entries that make up a specific LDAP hierarchy are defined in a schema for that LDAP directory. A schema is a text file that defines the attributes, types, and values that can belong to each entry in an LDAP directory, the objects that hold these entries, and the hierarchical context for each of these objects in an LDAP directory. On RHEL and Fedora Core systems, a number of basic LDAP schema are provided in the directory `/etc/openldap/schema`.

Schema files are divided into two basic types of entries: `attributetype` definitions and `objectclass` definitions. As the name suggests, each `attributetype` definition describes a specific attribute, defining its name, the syntax for that object, and how attributes of that type are compared. Similarly, each `objectclass` definition describes a single class of entries in an LDAP hierarchy, the context in which they appear in that hierarchy, and any mandatory components of that object.

Listing 34-1 shows an excerpt from the LDAP core schema, defined in the file `/etc/openldap/schema/core.schema` on Fedora Core and RHEL systems. This excerpt shows the `attributetype` definition for the `dc` (`domainComponent`) attribute and the `objectclass` (`dcObject`) that holds entries of that type.

```
# RFC 1274 + RFC 2247
attributetype ( 0.9.2342.19200300.100.1.25
    NAME ( 'dc' 'domainComponent' )
    DESC 'RFC1274/2247: domain component'
    EQUALITY caseIgnoreIA5Match
    SUBSTR caseIgnoreIA5SubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
# RFC 2247
objectclass ( 1.3.6.1.4.1.1466.344 NAME 'dcObject'
    DESC 'RFC2247: domain component object'
    SUP top AUXILIARY MUST dc )
```

Listing 34-1 Sample LDAP schema entries.

Each of these sample entries contains mysterious strings of decimal numbers separated by periods, which are referred to as dotted decimal values. These dotted decimal entries, such as those in the `attributetype`, `SYNTAX`, and `objectclass` definitions, are Object Identifiers (OIDs) that refer to pre-defined X.500 protocol elements, object types, attribute types, and attribute values. These are present in order to maintain compatibility with and leverage the X.500 specification.

The sample `attributetype` definition shown in Listing 34-1 contains the following declarations:

- **NAME** — One or more unique names identifying this attribute.
- **DESC** — Quoted text string that describes the attribute.
- **EQUALITY** — States that comparison of attributes of this type should be done in a case-insensitive fashion, using International Alphabet 5, which is an international standard alphanumeric code used to represent characters within a string. The United States' version of IA5 is more commonly known as American Standard Code for Information Interchange (ASCII).
- **SUBSTRING** — States that substring comparisons of attributes of this type should also be done in a case-insensitive fashion using International Alphabet 5.
- **SYNTAX** — Defines the OID for the syntax definition for the value of an attribute of this type, and that each attribute of this type can only have a single value.

The sample `objectclass` definition shown in Listing 34-1 contains the following declarations:

- **NAME** — A unique name identifying this class of objects
- **DESC** — A quoted text string that describes the attribute
- **SUP** — States that this object appears at the top of an LDAP directory, meaning that it has no superior object, and that it must contain a `dc` attribute

NOTE When discussing or examining LDAP directories, you will often see entries referred to by standard naming attributes such as Common Name (CN) and Organizational Unit (OU). These names are simply the naming attributes of specific types of objects — though they often appear to be basic parts of LDAP syntax, they are really just the names of commonly used attributes.

As you can see, LDAP schema files can be quite complex. They are analogous to the Document Type Definition (DTD) and schema files used to define the structure and types of content that can be found in XML documents. Like XML DTDs and schemes, the easiest way to get started using an LDAP schema is to find one that immediately satisfies your requirements (or, more often, the requirements of the LDAP clients that you want to use). In smaller organizations that use LDAP for specific purposes and with existing LDAP clients, you should easily be able to find an LDAP schema that meets your needs.

As discussed later in this section, OpenLDAP comes with a number of utilities for importing existing data into LDAP directories. Once you've settled on the schema that you want to use, you can use the appropriate utilities to create text files in LDAP Data Interchange Format (LDIF) that you can then import into your LDAP directory server. LDIF files are simple text-format files that make it easy to import new data into an LDAP server, and are analogous to the tab-delimited or comma-separated value (CSV) files used to import new data into databases and spreadsheets. You can also use LDIF files as a convenient mechanism for exporting data from one LDAP server and using this as a server-independent backup mechanism, or for importing that data into a separate LDAP directory server or a replica of your primary server.

OpenLDAP Packages for Linux

The packages that provide OpenLDAP libraries, server processes, utilities, and header files on RHEL and Fedora Core systems are the following:

- **openldap** — Libraries and basic documentation for running OpenLDAP clients and servers on a system
- **openldap-clients** — OpenLDAP clients and associated documentation
- **openldap-servers** — OpenLDAP servers, server status utilities, server configuration files and basic scheme, scripts for migrating data into or out of an OpenLDAP server, and associated documentation

You can find the specific versions of these packages that are installed on your system using the following command:

```
$ rpmquery -a | grep -i ldap
perl-LDAP-0.31-5
openldap-servers-2.2.13-2
compat-openldap-2.1.30-2
openldap-servers-sql-2.2.13-2
mod_authz_ldap-0.26-2
openldap-devel-2.2.13-2
php-ldap-4.3.11-2.4
nss_ldap-220-3
openldap-clients-2.2.13-2
openldap-2.2.13-2
python-ldap-2.0.1-2
```

The version numbers you see might be slightly different. In addition to the required core packages, the following auxiliary packages are available in the distribution media for Fedora Core and RHEL systems:

- **compat-openldap** — Compatibility libraries that may be required by existing LDAP clients.
- **mod_authz_ldap** — An Apache Web server module that enables Apache users to authenticate against an LDAP directory.
- **nss_ldap** — Provides two sets of LDAP clients and associated files: `nss_ldap` and `pam_ldap`. The `nss_ldap` package enables X.500 and LDAP directory servers to be used as the primary source of system configuration files used by the system and the Name Service Switch, such as `/etc/aliases`, `/etc/group`, `/etc/hosts`, `/etc/shadow`, and so on. The `pam_ldap` package provides Linux Pluggable Authentication Modules that support authentication, password changes, Netscape's SSL, NIS's `ypldapd`, and so on.
- **openldap-devel** — Headers, libraries, and associated documentation needed for developing and compiling OpenLDAP clients and servers. This package also includes text versions of all of the X.500 and LDAP RFC (Request for Comment) documents that define these standards.
- **openldap-servers-sql** — A loadable module for the OpenLDAP server that enables it to read information stored in a SQL database.
- **perl-LDAP** — Modules and associated documentation for interacting with LDAP servers from within scripts written in the Perl scripting language.
- **php-ldap** — A dynamic shared object (DSO) module and associated documentation for the Apache Web server that enables PHP applications to interact with an OpenLDAP server.
- **python-ldap** — Provides support for accessing OpenLDAP servers from within scripts written in the Python scripting language.

Core OpenLDAP Server Files, Daemons, and Utilities

This section highlights the most common command server processes, configuration files and directories, and startup scripts associated with an OpenLDAP server. For additional information about any of these processes or files, consult the online reference information for them by using the `man` command.

The following are the most commonly used OpenLDAP binaries on a Fedora Core or RHEL system:

- **ldapadd** — Adds new entries to an OpenLDAP directory. Located in the directory `/usr/bin`.
- **ldapdelete** — Deletes one or more entries from an OpenLDAP directory. Located in the directory `/usr/bin`.

- **ldapmodify** — Modifies existing entries in an OpenLDAP directory. Located in the directory `/usr/bin`.
- **ldapsearch** — Locates entries in an OpenLDAP directory. Located in the directory `/usr/bin`.
- **slapadd** — Imports entries from an LDIF file into an OpenLDAP directory. Located in the directory `/usr/sbin`.
- **slapd** — The LDAP server. Located in the directory `/usr/sbin`.
- **slapcat** — Extracts entries from an LDAP server, creating an LDIF file. Located in the directory `/usr/sbin`.
- **slapindex** — Reindexes an OpenLDAP directory. Located in the directory `/usr/sbin`.
- **slappasswd** — Generates an encrypted password entry that you can use in the slapd configuration file or for storage in your LDAP directory. Located in the directory `/usr/sbin`.

The OpenLDAP server is typically started, stopped, or restarted using the command script `/etc/init.d/ldap`. As with all startup scripts on Fedora Core or RHEL system, you should symlink this file to start up and kill files in the directories associated with your system's multiuser run levels. You can either do this manually or by using the `ckconfig -add ldap` command.

All of the systemwide configuration files for the OpenLDAP and the Linux OpenLDAP daemon are located in the directory `/etc/openldap`:

- **ldap.conf** — Sets default values used by OpenLDAP clients on your system. The values in this file can be overridden by the contents of the user-specific `~/.ldapprc` file, or by a file called `ldapprc` located in the current working directory.
- **schemas** — A directory that contains a number of default schema definitions for your use when creating an OpenLDAP directory.
- **slapd.conf** — Configuration information for the slapd server running on the current system. This file should never be readable by non-privileged users, because it contains password and other security information for your OpenLDAP server.

Configuring and Starting an OpenLDAP Server

Configuring an OpenLDAP server is a simple process, requiring a minimum of two changes to the `/etc/openldap/slapd.conf` configuration file.

NOTE By default, OpenLDAP password are sent in the clear over a network. If possible, you should always initially configure your OpenLDAP server from the system on which it is running.

First, you must change the `suffix` entry so that it correctly identifies your domain. For example, the default entry in `/etc/openldap/slapd.conf` is usually:

```
suffix          "dc=my-domain,dc=com"
```

You must change this to reflect your domain. For example, to set up an OpenLDAP server for the domain `vonhagen.org`, you would change this line to the following:

```
suffix          "dc=vonhagen,dc=org"
```

Next, you must change the `rootdn` entry to reflect the name of a privileged user who has unrestricted access to your OpenLDAP directory. For example, the default entry in `/etc/openldap/slapd.conf` is usually:

```
rootdn          "cn=Manager,dc=my-domain,dc=com"
```

Continuing with the previous example, you would change this to something like the following for the `vonhagen.org` domain:

```
rootdn          "cn=ldapadmin,dc=vonhagen,dc=org"
```

Though this user is the equivalent of the root user as far as OpenLDAP is concerned, the name of this user does not have to be a real user on your system.

TIP For security reasons, if your OpenLDAP directory is going to be accessed from other systems, you may want to use a name other than common defaults such as “root” or “Manager” for this entry. This will make it more difficult for someone to attempt to crack your OpenLDAP installation.

Finally, though optional in some sense, you may want to set a unique password for your OpenLDAP server by modifying the `rootpw` entry in your `/etc/openldap/slapd.conf` configuration file. This enables you to configure, test, and correct your OpenLDAP system over your local network, if necessary. For example, the default entry in `/etc/openldap/slapd.conf` for this entry is the following:

```
rootpw          secret
```

You can provide a clear text or encrypted password as the value for this entry. The default entry in the `/etc/openldap/slapd.conf` file shows the clear-text password `secret`. You can use the `slappasswd` command to generate an encrypted password that you can paste into the `/etc/openldap/slapd.conf` file, as in the following example:

```
# slappasswd
New password:
Re-enter new password:
{SSHA}x0uopfQDBaylPdv3zfjLqOSkrAUh5GgY
```

The `slappasswd` command prompts you for a new password, asks for confirmation, and then displays the encrypted password string preceded by the encryption mechanism used in the password. You then simply replace the value of the existing `rootpw` option with the generated string, as in the following example:

```
rootpw          {SSHA}x0uopfQDBaylPdv3zfjLqOSkrAUh5GgY
```

As mentioned previously, you should enable only the `rootpw` option when initially configuring your OpenLDAP server, and it is only necessary to do so if you must configure your OpenLDAP server over a network. If you do so, it is always a good idea to set a unique, encrypted password for your OpenLDAP server that differs from your standard root password, even though the `/etc/openldap/slapd.conf` file should not be readable by nonprivileged users on your system. Once you have completed your configuration, you should disable this entry by commenting it out. To do so, put a hash mark (#) at the beginning of the line containing the `rootpw` entry.

NOTE OpenLDAP passwords are sent over the network in the clear unless you enable SSL/TLS (Secure Socket Layer/Transaction Layer Security) encryption in your `/etc/openldap/slapd.conf` file. The default file provided with RHEL and Fedora Core systems does not include the entries for doing so, which are the `TLSCipherSuite`, `TLSCertificateFile`, and `TLSCertificateKeyFile` entries. Discussing SSL/TLS encryption for OpenLDAP is outside the scope of this introductory section — for additional information, see a reference such as O'Reilly's *LDAP System Administration* (ISBN: 1-56592-491-6).

After you have modified your `/etc/openldap/slapd.conf` file and have saved your changes, you can start the OpenLDAP server using the `/etc/init.d/ldap` script, as in the following example:

```
# /etc/init.d/ldap start
```

After you have started an OpenLDAP server, the next step is to use the `ldapadd` command to populate it with the data that you initially want it to contain, and to then query its contents using a utility such as `ldapsearch`, to verify the data that you've entered. However, the best test of an OpenLDAP directory is to put it into production using an OpenLDAP client.

LDAP directories are commonly used on Linux systems as a central repository for enterprise-wide phone books, email addresses and contact lists, and as

a repository for authentication information. As an example of populating and using an LDAP directory, the next section explains how to configure your system to use OpenLDAP for login and general authentication information.

Using OpenLDAP for System Authentication

Using LDAP for system authentication provides an easy way of centralizing authentication information in enterprise networks that use LDAP for other purposes or which simply do not want to use other networked authentication mechanisms such as NIS.

There are two steps involved in switching to LDAP authentication:

1. On the LDAP server system, add the appropriate password and group entries for every user in your environment to the LDAP directory.
2. Configure each client system to use LDAP for authentication.

The examples in the following sections assume that you have entered the name `ldap` as a valid entry for your LDAP server in DNS. These steps are discussed in the next two sections.

Adding User, Password, and Group Entries to an LDAP Server

In order to configure your LDAP server to provide authentication information, you must first migrate your existing authentication information to the LDAP server. You do this by preparing LDIF files that hold the contents of your `/etc/passwd`, `/etc/shadow`, and `/etc/group` files, and then importing those files into the LDAP server.

Creating LDIF files from your existing `/etc/passwd`, `/etc/shadow`, and `/etc/group` files is most easily done by using the `migrate_passwd.pl` and `migrate_group.pl` scripts found in the directory `/usr/share/openldap/migration`. These scripts are installed as part of the `openldap-servers` package, and should already be installed on your LDAP server.

TIP If you have multiple password, shadow, and group files on different systems that you want to merge into a single LDAP repository, you can copy them all to your LDAP server system, concatenate them, and sort them to produce single files. You can then edit these files so that they only have single entries for each user and group and install them as the master password, shadow, and group files on your server before running the migration scripts. Verify that these files work correctly after installation and before migrating them to LDAP!

To migrate user, password, and group information into your LDAP server in order to use it as a basis for client system authentication, do the following:

1. Become the root user, and change directory to the directory:

```
$ su root
Password:
# cd /usr/share/openldap/migration
```

2. Edit the file `migrate_common.ph`, which sets variables used by all of the migration scripts. Set the value for the `DEFAULT_BASE` variable to the correct value for your environment. As an example, the correct value for migrating information to the LDAP server used as an example throughout in this chapter is:

```
$DEFAULT_BASE = "dc=vonhagen,dc=org";
```

3. Generate an LDIF file for your user and password information using the `migrate_passwd.pl` script, as in the following example. The `migrate_passwd.pl` script also extracts the necessary password information from your `/etc/shadow` file:

```
$ ./migrate_passwd.pl /etc/passwd passwd.LDIF
```

4. Generate an LDIF file for your group information using the `migrate_group.pl` script, as in the following example:

```
$ ./migrate_group.pl /etc/group group.LDIF
```

5. Import the files that you just created into your LDAP directory using commands like the following:

```
# ldapadd -x -h hostname -D "cn=ldapadmin,dc=vonhagen,dc=org" \
> -w password -f passwd.LDIF
# ldapadd -x -h hostname -D "cn=ldapadmin,dc=vonhagen,dc=org" \
> -w password -f group.LDIF
```

These commands are ordinarily entered on one line — they are split here for formatting reasons.

In these commands, replace `hostname` with the hostname of the system on which your LDAP server is running, make sure that the credentials specified following the `-D` option match those of the root user for your LDAP server, and replace `passwd` with the password you set in the `rootpw` entry, both as defined in your OpenLDAP server configuration file, `/etc/openldap/slapd.conf`.

After following these steps, you are ready to update your client systems to use LDAP authentication (and test them, of course).

Updating Client Systems to Use LDAP Authentication

On each system that you want to use the new LDAP authentication server, you must do the following:

1. Make sure that the `openldap`, `openldap-clients`, and `nss_ldap` packages are installed.
2. Run the `/usr/bin/authconfig` command, and select Use LDAP in the User Information section and Use LDAP for Authentication in the Authentication section of the Authentication Configuration screen. Figure 34-5 shows this screen with the correct options selected.
3. Select Next and press Enter to proceed to the next screen and enter the correct host and distinguished name information for your LDAP server on the LDAP Server screen. Continuing with the example used throughout this section, the following are appropriate host base entries:

```
host ldap.vonhagen.org
base dc=vonhagen,dc=org
```

Figure 34-6 shows this screen with the options appropriate to this example selected. Select OK and press Enter to exit the `authconfig` application.

4. Check the `/etc/ldap.conf` file to make sure that it contains the LDAP settings that you entered in the `authconfig` application. If the values that you entered were not correctly propagated to this file, set them manually.
5. Check the `/etc/pam.d/system-auth` file to make sure that the entries for LDAP authentication information have been correctly added. This file should look something like the following:

```
auth      required      /lib/security/$ISA/pam_env.so
auth      sufficient    /lib/security/$ISA/pam_unix.so likeauth nullok
auth      sufficient    /lib/security/$ISA/pam_ldap.so use_first_pass
auth      required      /lib/security/$ISA/pam_deney.so

account   required      /lib/security/$ISA/pam_unix.so broken_shadow
account   sufficient    /lib/security/$ISA/pam_succeed_if.so uid < 100
quiet
account   [default=bad success=ok user_unknown=ignore]
/lib/security/$ISA/pam_ldap.so
account   required      /lib/security/$ISA/pam_permit.so

password  requisite     /lib/security/$ISA/pam_cracklib.so retry=3
password  sufficient    /lib/security/$ISA/pam_unix.so nullok
use_authok md5 shadow
password  sufficient    /lib/security/$ISA/pam_ldap.so use_authok
password  required      /lib/security/$ISA/pam_deney.so

session   required      /lib/security/$ISA/pam_limits.so
session   required      /lib/security/$ISA/pam_unix.so
session   optional      /lib/security/$ISA/pam_ldap.so
```

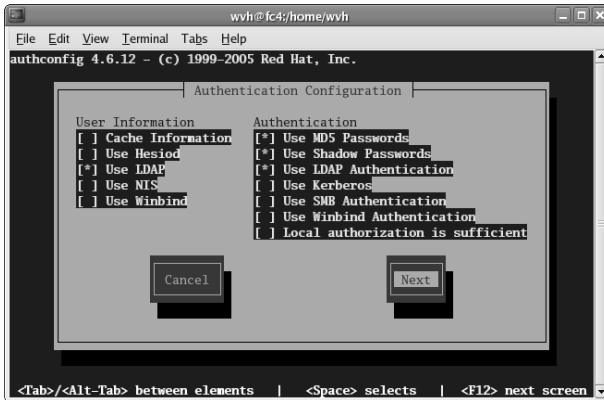


Figure 34-5 Selecting LDAP Authentication in authconfig.

6. Modify your `/etc/nsswitch.conf` file to specify that password, shadow, and group information are first looked for in LDAP. The following are correct entries:

```
passwd: ldap files
shadow: ldap files
group: ldap files
```

This tells the Name Service Switch to check LDAP first and then fall back to checking explicit password, shadow, and group files on the client system.

The next time that you log in on your client system, it will contact your LDAP server for authentication information.

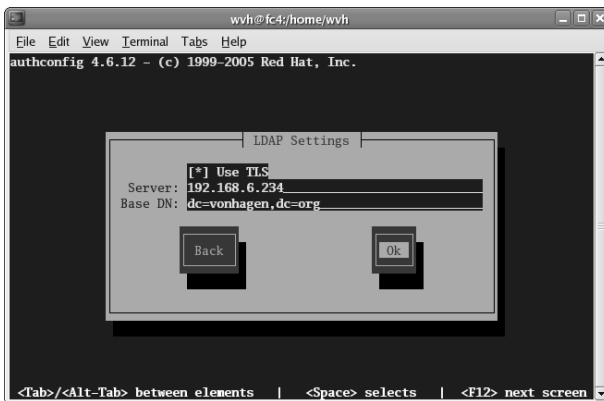


Figure 34-6 LDAP Environment Settings in authconfig.

CAUTION Before logging out of this client system and configuring another, open a new login session to this host using the `telnet` or `ssh` commands to ensure that you can correctly log in using LDAP. If you encounter any problems, do not log out of this system until you have resolved them.

Installing, Configuring, and Using Kerberos

Kerberos is a distributed authentication service that was originally developed at the Massachusetts Institute of Technology (MIT) for use with MIT's Athena project, a distributed computing environment. Kerberos was popularized by its use at MIT and in the AFS distributed file system, developed at Carnegie-Mellon University, Transarc Corporation, and now available as a supported IBM product and an open-source project. The security provided by Kerberos is so well-respected and robust that Kerberos has even been adopted by Microsoft as an underlying authentication model for Windows 2000 and subsequent versions of Windows.

Kerberos is designed to provide secure authentication for client-server applications by using strong cryptography to enable clients to prove their identity to servers over the network. In more advanced Kerberos installations, clients and servers that have used Kerberos to prove their identity to each other can optionally encrypt all subsequent communications to ensure privacy. This requires that all applications that exchange data over the network be either special Kerberos-aware versions or that they link with Kerberos-aware libraries.

In its simplest form, Kerberos works by exchanging encrypted security information between clients, which can be users or machines, the Kerberos Authentication Server, and the resource you are trying to access. The information that is initially exchanged when attempting to prove one's identity is known as a *ticket*. The information used to encrypt tickets and subsequent communications is known as a *key*. After the identity of a client is verified, that client is granted a Kerberos token that can be used to verify the client's identity to any Kerberos-aware service. For security reasons, Kerberos tokens are timestamped so that they automatically expire unless renewed by a user or service.

The timestamps contained within Kerberos tokens (and tickets) can be verified only if the time and data are synchronized across Kerberos clients and servers. Kerberos authentication will fail if client and server clocks become skewed by more than 5 minutes. This is the suggested default value, which you can change in your Kerberos configuration files. We strongly suggest that you run Network Time Protocol (NTP) daemons on all Kerberos clients and servers to ensure that their clocks remain in sync. It is also a good idea to set up replicated time servers for any site using Kerberos so that your site can still synchronize client clocks if you encounter problems connecting to the Internet.

Kerberos is often referred to as a trusted third-party authentication service because each participant in the Kerberos authentication process believes in Kerberos's assessment of the identity of the other parties in that process. Kerberos verifies each participant's assessment using private keys that each participant generates during each stage of the authentication process. Kerberos is very robust and secure because all stages of the authentication process exchange encrypted information. At no point are passwords transmitted over the network without first being encrypted.

Kerberos Terminology, Machine Roles, and Reliability

As a sitewide authentication mechanism, Kerberos introduces some new terms that you must understand in order to use Kerberos effectively. The most basic of these is the idea of a *realm*, which is essentially a set of machines that uses a Kerberos server for authentication and that, therefore, trusts that server. In Kerberos configuration files, your realm is typically identified in uppercase characters to differentiate it from any, usually similar, DNS domain with which it is associated.

Reliability is a critical aspect of a sitewide authentication mechanism. Kerberos environments are protected against the failure of key systems and services by replicating those systems on slave systems. The most critical of these is the Key Distribution Center (KDC) system, the primary system for granting tickets and the system that houses the master copy of the Kerberos database. KDC slaves contain copies of the Kerberos database but cannot perform administrative functions; they only tickets grant in the event that the primary system is unavailable.

TIP As a general rule, all KDC systems should be installed so that they can serve as either a master or a slave. In the event of a hardware problem with your primary KDC systems, this simplifies converting an existing slave to a new master KDC.

Kerberos Packages for Linux

The packages that provide Kerberos and Kerberos-related libraries, server processes, utilities, and header files on RHEL and Fedora Core systems are:

- **krb5-workstation** — Contains basic Kerberos programs (`kinit`, `klist`, `kdestroy`, `kpasswd`) as well as Kerberized versions of the `telnet` and `ftp` applications. This package should be installed on every client of a Kerberos server.
- **krb5-server** — Provides the programs that must be installed on a Kerberos 5 server or server replica.

- **krb5-libs** — Contains the shared libraries required for Kerberos clients and servers.
- **krbafs-utils** — Provides versions of core utilities for the AFS distributed file system that are linked against the krbafs library.
- **krbafs-devel** — Includes header files and static libraries for developing and compiling applications that use the krbafs library
- **krbafs** — Provides the krbafs shared library that enables programs to obtain tokens for the AFS distributed file system using Kerberos 4 credentials without needing access to the official AFS libraries.
- **krb5-auth-dialog** — Contains a pop-up dialog that warns users when their Kerberos tickets are about to expire and enables them to renew them.
- **krb5-devel** — Includes header files and libraries necessary for developing and compiling Kerberos 5 applications.
- **pam_krb5** — Provides a PAM (Pluggable Authentication Module) that enables Kerberos authentication. This package also includes a PAM (`pam_krb5afs`) that can get tokens for the AFS distributed file system.

NOTE The Kerberos packages supplied with Red Hat Linux are not the only Kerberos implementation available. Other freely available versions of Kerberos include the Heimdal project (www.pdc.kth.se/heimdal), the Shishi project (<http://josefsson.org/shishi>), and the original implementation from the Kerberos mothership at MIT (<http://web.mit.edu/kerberos/www>).

Core Kerberos Utilities

This section highlights the most common utilities associated with Kerberos authentication. For additional information about any of these processes or files, consult the online reference information for them by using the `man` command. All of these utilities are located in the `/usr/kerberos/bin` directory on a Fedora Core or RHEL system:

- **kdestroy** — Deletes and tokens owned by the current user.
- **kinit** — Enables you to obtain tokens manually for a specified user.
- **klist** — Lists the tokens of the current, or a specified, user.
- **kdestroy** — Deletes tokens owned by the current user.

The `/usr/kerberos/bin` directory also contains Kerberized versions of common applications such as `ftp`, `rcp`, `rsh`, `telnet`, and so on. On systems that use Kerberos, you should put the `/usr/bin/kerberos` directory in the

default PATH for users before standard system directories such as /usr/bin and /bin.

Installing and Configuring a Kerberos Server

After installing the krb5-libs, krb5-server, and krb5-workstation packages on the system that you are going to use as your primary Key Distribution Center, the first step in configuring your Kerberos environment is to set up your master Key Distribution Center. The process for doing this is the following:

1. Edit the general Kerberos configuration file for your environment, /etc/krb5.conf. This file identifies the KDCs and admin servers in your Kerberos realm and provides default values for your realm, Kerberos applications, and for how your existing hostnames map into your Kerberos realm. A sample /etc/krb5.conf file for the realm VONHAGEN.ORG is the following:

```
[logging]
default = FILE:/var/log/krb5libs.log
kdc = FILE:/var/log/krb5kdc.log
admin_server = FILE:/var/log/kadmind.log
[libdefaults]
default_realm = VONHAGEN.ORG
dns_lookup_realm = false
dns_lookup_kdc = false
ticket_lifetime = 24h
forwardable = yes
[realms]
VONHAGEN.ORG = {
    kdc = kerberos.vonhagen.org:88
    admin_server = kerberos.vonhagen.org:749
    default_domain = vonhagen.org
}
[domain_realm]
.vonhagen.org = VONHAGEN.ORG
vonhagen.org = VONHAGEN.ORG
[kdc]
profile = /var/kerberos/krb5kdc/kdc.conf
[appdefaults]
pam = {
    debug = false
    ticket_lifetime = 36000
    renew_lifetime = 36000
    forwardable = true
    krb4_convert = false
}
```

The defaults provided in the `/etc/krb5.conf` file installed by the `krb5-server` package are reasonable, except that you must change all instances of `EXAMPLE.COM` to the name of your realm and all instances of `example.com` to the name of your domain (`VONHAGEN.ORG` and `vonhagen.org`, respectively, in the previous example). You must also make sure that DNS or `/etc/hosts` entries exist on all clients for the systems that you identify as your default KDC and `admin_server` systems in the `[realms]` section.

2. Edit the Key Distribution Center configuration file, `/var/kerberos/krb5kdc/kdc.conf`. The location of this file is provided in the `[kdc]` section of the `/etc/krb5.conf` file. As with the `/etc/krb5.conf` file, the primary change that you must make to this file is to change the instance of `EXAMPLE.COM` to the name of your realm, which is `VONHAGEN.ORG` in the following example:

```
[kdcdefaults]
acl_file = /var/kerberos/krb5kdc/kadm5.acl
dict_file = /usr/share/dict/words
admin_keytab = /var/kerberos/krb5kdc/kadm5.keytab
v4_mode = nopreauth
[realms]
VONHAGEN.ORG = {
    master_key_type = des-cbc-crc
    supported_encytypes = des3-hmac-sha1:normal arcfour-hmac:normal \
    des-hmac-sha1:normal des-cbc-md5:normal des-cbc-crc:normal \
    des-cbc-crc:v4 des-cbc-crc:afs3
}
```

3. Use the `kdb5_util` command on the Master KDC to create the Kerberos database and your stash file. You will have to enter the master database password twice, for verification purposes. The stash file is a local, encrypted copy of the master key that is used to automatically authenticate the KDC as part of your system's startup sequence.

```
# /usr/kerberos/sbin/kdb5_util create -r VONHAGEN.ORG -s
Loading random data
Initializing database '/var/kerberos/krb5kdc/principal' for realm
'vonhagen.org',
master key name 'K/M@vonhagen.org'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key:
Re-enter KDC database master key to verify:
```

This command creates various files in the directory specified in the `kdcdefaults` section of your `kdc.conf` file: two Kerberos database files (`principal.db` and `principal.ok`), the Kerberos

administrative database file (`principal.kadm5`), the database lock file (`principal.kadm5.lock`), and the stash file (`.k5stash`).

4. Edit the ACL definition file, `/var/kerberos/krb5kdc/kadm5.acl`, changing the default realm (`EXAMPLE.COM`) to the name of the realm that you are creating (`VONHAGEN.ORG`, in this example). The default entry in this file, which begins with `*/admin`, gives any user with an admin instance (such as `wvh/admin`, which you create in the next step) complete access to and control over the realm's Kerberos database. After updating this file for our example realm, this file looks like the following:

```
*/admin@VONHAGEN.ORG *
```

TIP Kerberos administrative permissions are very granular and enable you to grant different levels of administrative privileges to certain users and system administrators. See the Kerberos documentation in `/usr/share/doc/krb5-server-version` for more detailed information about expressing different levels of Kerberos permissions in this file. If you subsequently want to refine the permissions with a user who has an associated `/admin` instance, you should create an entry for that user in the `/var/kerberos/krb5kdc/kadm5.acl` file before the default `*/admin` permissions entry for `/admin` users.

5. Use the `kadmin.local` command to add each of your system administrators to the Kerberos database. The `kadmin.local` command is a Kerberos-aware version of the standard `kadmin` utility that does not first authenticate to a Kerberos database and is, therefore, used for bootstrapping Kerberos on a KDC. Entries in the Kerberos database are known as principals. The following example adds an admin instance for the user `'wvh'`:

```
# /usr/kerberos/sbin/kadmin.local
kadmin.local: addprinc wvh/admin
WARNING: no policy specified for wvh/admin@VONHAGEN.ORG; defaulting
to no policy
Enter password for principal "wvh/admin@VONHAGEN.ORG":
Re-enter password for principal "wvh/admin@VONHAGEN.ORG":
Principal "wvh/admin@VONHAGEN.ORG" created.
```

6. Add a standard user entry for the nonadmin version of the principal that you just created and then exit the `kadmin.local` utility, as in the following example. Adding a standard principal enables default authentication by the associated entity. You will eventually need to create a principal for each user that you want to be able to authenticate using Kerberos. Most sites do this by writing a script that also created Kerberos principals when creating standard user accounts.

```
kadmin.local: addprinc wvh
WARNING: no policy specified for wvh@VONHAGEN.ORG; defaulting to no
policy
Enter password for principal "wvh@VONHAGEN.ORG":
Re-enter password for principal "wvh@VONHAGEN.ORG":
Principal "wvh@VONHAGEN.ORG" created.
kadmin.local: quit
```

7. Now, the fun begins! Start the various Kerberos-related services using the following commands:

```
# /sbin/service krb5kdc start
# /sbin/service kadmin start
# /sbin/service krb524 start
```

At this point, you're ready to install and start a Kerberos client, as explained in the next section. Before doing anything else, you should verify that your server can hand out tickets by using the `kinit` command to explicitly request one for the administrative principal that you created earlier. You can then use the `klist` command to verify its contents, and then destroy the ticket (just to clean up) using the `kdestroy` command. The following example shows this sequence:

```
$ kinit wvh
Password for wvh@VONHAGEN.ORG:
$ klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: wvh@VONHAGEN.ORG
Valid starting      Expires            Service principal
05/03/05 22:09:04   05/04/05 22:09:04   krbtgt/VONHAGEN.ORG/VONHAGEN.ORG
Kerberos 4 ticket cache: /tmp/tkt0
klist: You have no tickets cached
$ kdestroy
```

Enabling Kerberos Clients and Applications

Setting up a system as a Kerberos client is simple:

1. Copy the `/etc/krb5.conf` file from your KDC to the client.
2. Enable a sample application. Use `krb-telnet`, a Kerberos-aware version of the classic `telnet` application, as a test application. The `krb-telnet` server is managed by your system's `inet` daemon. To enable `krb-telnet`, modify the file `/etc/xinetd.d/krb-telnet` changing the `disable` entry from `yes` to `no`, as in the following example:

```
# default: off
# description: The Kerberized telnet server accepts normal telnet
```

```
#           sessions, but can also use Kerberos 5 authentication.
service telnet
{
    Flags            = REUSE
    socket_type      = stream
    wait            = no
    user            = root
    server          = /usr/kerberos/sbin/telnetd
    log_on_failure += USERID
    disable         = no
}
```

3. Restart your system's inet daemon using the following command:

```
# /etc/init.d/xinetd.d restart
```

4. Telnet to your system and make sure that you can log in successfully. After you log in, you can use the `klist` command to verify that you've automatically been granted the appropriate Kerberos tokens, as in the following example:

```
$ klist
Ticket cache: FILE:/tmp/krb5cc_p4979
Default principal: wvh@VONHAGEN.ORG
Valid starting Expires Service principal
05/07/05 10:00:46 05/08/05 10:00:46
krbtgt/VONHAGEN.ORG@VONHAGEN.ORG
Kerberos 4 ticket cache: /tmp/tkt500
klist: You have no tickets cached
```

Congratulations, Kerberos is working! The next step in taking full advantage of Kerberos is to integrate it into your system's login authentication process, as described in the next section.

TIP As mentioned earlier in this section, make sure that the time and date are synchronized between your KDCs and any Kerberos client systems before proceeding with this section. By default, a difference of more than five minutes will cause Kerberos authentication to fail.

Using Kerberos for Login Authentication

If you are going to be using Kerberos for login authentication, testing Kerberos clients such as `krb-telnet`, as described in the previous section, is a great way to make sure that your Kerberos server is working and that your clients can communicate with it successfully. When you're sure that everything's working correctly, integrating Kerberos with the PAMs used for your system's login process is a logical next step.

The `authconfig` applications provided by the RHEL and Fedora distributions simplifies integrating new authentication mechanisms by making all of the necessary modifications to the `/etc/pam.d/system-auth` PAM control file for you. To enable Kerberos authentication across your system, do the following:

1. Run the `/usr/bin/authconfig` command and Use Kerberos in the Authentication section of the Authentication Configuration screen. Figure 34-7 shows this screen with the correct option selected.
2. Select Next and press Enter to proceed to the next screen and enter the name of your realm and the IP addresses or hostnames of your KDC and admin server (which are the same in the examples used in this chapter).

Figure 34-8 shows this screen with the options appropriate to this example selected. Select OK and press Enter to exit the `authconfig` application.

NOTE Using DNS to locate hosts and associated Kerberos realms requires adding special entries to your DNS server configuration files. For more information about this process, see the documentation in `/usr/share/doc/krb5-server-version` for more information.

3. After exiting from `authconfig`, log out and log back in. After you log in, use the `klist` command to verify that you have Kerberos tokens, which will display information identical to that shown at the end of the previous section.

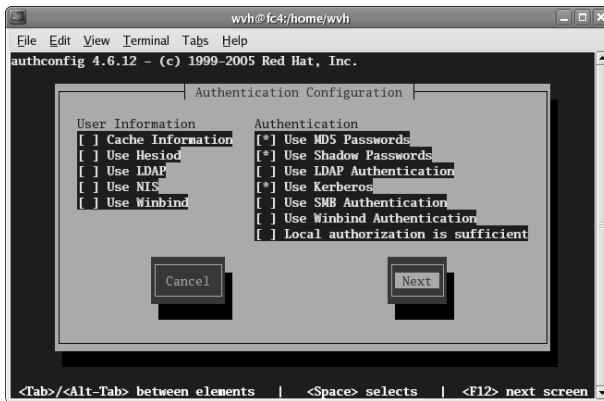


Figure 34-7 Selecting Kerberos authentication in `authconfig`.

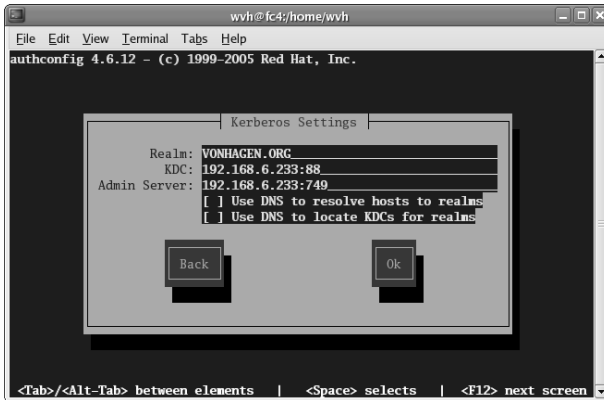


Figure 34-8 Kerberos environment settings in authconfig.

After using authconfig, your `/etc/pam.d/system-auth` file will look like the following:

```
auth      required      /lib/security/$ISA/pam_env.so
auth      sufficient    /lib/security/$ISA/pam_unix.so likeauth
nullok
auth      sufficient    /lib/security/$ISA/pam_krb5.so
use_first_pass
auth      required      /lib/security/$ISA/pam_deny.so
account   required      /lib/security/$ISA/pam_unix.so
broken_shadow
account   sufficient    /lib/security/$ISA/pam_succeed_if.so uid <
100 quiet
account   [default=bad success=ok user_unknown=ignore]
/lib/security/$ISA/pam_krb5.so
account   required      /lib/security/$ISA/pam_permit.so
password  requisite      /lib/security/$ISA/pam_cracklib.so retry=3
password  sufficient    /lib/security/$ISA/pam_unix.so nullok
use_auth tok md5 shadow
password  sufficient    /lib/security/$ISA/pam_krb5.so use_auth tok
password  required      /lib/security/$ISA/pam_deny.so
session   required      /lib/security/$ISA/pam_limits.so
session   required      /lib/security/$ISA/pam_unix.so
session   optional      /lib/security/$ISA/pam_krb5.so
```

TIP If you have any problems with Kerberos login authentication, enable PAM debugging in your `/etc/krb5.conf` file so that you can quickly identify and resolve authentication-related problems with login and other system applications that use PAMs. To do this, simply set the debug entry in the PAM section of the `[appdefaults]` stanza to `true` and restart your Kerberos server.

Summary

In an Internet environment that is increasingly hostile, system administrators must take steps to ensure the security of the systems and LANs they administer. One step is to use Linux's Netfilter packet filtering technology to create a firewall that carefully controls the ingress of Internet traffic into the LAN. A second step is to create a centralized LDAP data store that all systems use for access control and authentication. A third step is to use authentication technology, such as Kerberos, that provides enhanced security via the exchange of encrypted keys, which does not directly transmit authentication data over the Internet and which can also provide encryption support for other types of network communication.

Troubleshooting and Problem Solving

IN THIS CHAPTER

- Troubleshooting Techniques
- Troubleshooting Resources
- Solving Common Problems
- Solving File System Problems
- Solving Networking Problems
- Solving NFS Problems
- Exploring Miscellaneous Problems
- Making an Emergency Boot Disk

Even the best-laid plans rarely survive contact with reality, and Red Hat Enterprise Linux is no exception. This chapter describes various problems that can occur with various subsystems and features of Fedora Core and Red Hat Enterprise Linux and suggests possible solutions. Obviously, we cannot cover everything that might go awry, but we'll cover the problems *we* had and how we solved them. If our crystal ball is working, we may even be able to foresee and help prevent a few problems.

Despite the work and testing that go into preparing each release, unanticipated problems inevitably emerge. Most of these problems result from one of three situations:

- Testing Fedora Core and Red Hat Enterprise Linux for compatibility with every piece of hardware is simply not possible.
- Given the range of hardware available, any given combination of two components, for example, a SCSI disk controller and a SCSI disk, may result in subtle but maddening incompatibilities.

- As a result of the rapid rate of hardware revisions, drivers written for earlier revisions of your hardware might not support the latest hardware version.

This chapter is intended to help you troubleshoot and solve the most common configuration challenges you might encounter when installing and using Fedora Core and Red Hat Enterprise Linux. In particular, the first section of this chapter gives you some general troubleshooting techniques that you can use to solve many problems you may encounter, not just the ones in this chapter. Other sections in this chapter help you resolve problems related to installation, the file system, networking, and booting the system. A final section addresses a few common problems that do not fit into other categories.

Troubleshooting Techniques

Trying to fix a system can seem like a daunting task, especially for a new user or someone without a lot of troubleshooting experience. But, following an established set of troubleshooting steps can go a long way toward making the search for the solution to the problem a lot easier. This section will show you some steps you can follow in a specific order to find and then try to solve the problem.

Step 1: Identify the Problem

This step may seem obvious to you, but is it really? Suppose that you get a call from a user who reports being unable to check email. This is the problem, right? Not being able to check email may not be the real problem, but a symptom, the underlying problem may not be so obvious. What is the problem? Is it the physical network connection, or the DNS server, the default gateway, or perhaps the email client is improperly configured. As you can see, there may be many reasons for not being able to check email and it is important to find and solve the actual problem, not the symptom of the problem. Knowing the symptom is important because it can lead to the actual problem, which you can then solve.

You need to ask yourself, or whoever reported the problem to you, some detailed questions to try to get to the root of the problem. For example, you might ask if it is possible to access Web sites on the Internet. If the answer is yes, you can rule out problems with the physical network connection. You can also rule out problems with the DNS servers and the default gateway as well. By asking just one question you are able to rule out several possible problem areas.

Step 2: Reproduce the Problem

If you discover a problem with your systems, you should always try to reproduce the problem. If a user on your network reports a problem, you should try

to get him or her to reproduce the problem. Many times a user will immediately call to report being unable to do some task without even knowing what exactly they were doing when the problem occurred. Ask users to try the task again and to write down exactly what they were doing when the problem happened. Ask them if they received any error messages and, if so, what the message said. By doing this you will have more information that you can use to help you figure out the problem and its cause. If possible, try to watch users while they are performing the task to determine if they are doing it correctly.

Step 3: Look for Changes

In many cases systems work perfectly for months and suddenly they stop working. A service that worked countless times in the past is not working now. Why not? You need to determine what has changed between the time all was well and the problem was reported. You can ask some questions that may lead you to a solution. Some questions you may ask are:

- **Could you ever perform the task?** This might sound like a silly question, but often users will try to do something the system wasn't set up to do. Perhaps they can do the task at home and decided to try it at work. You may need to explain to the users that they may not be able to do the task and it isn't a problem with their system.
- **If a service was available, when did it become unavailable?** Knowing the answer to this question may lead you to a change that occurred immediately before the problem happened. The cause of the problem may be directly related to the change. This question also leads into the next question.
- **What changed since the service was available?** Did the user install any software on the system? Did you install any software on the system? Many times the change that occurred is responsible for the problem.
- **Are other users affected by the problem?** If the problem is affecting only one user, you can usually isolate the problem to that user's system.

These are just a few of the questions you may ask yourself or your users. There are many other probing questions you can ask at this step and also other steps of the troubleshooting process. The more information you have, the easier your troubleshooting will be.

Step 4: Determine the Most Likely Cause

By now you have gone through three steps in your troubleshooting and should be able to narrow down the problem. Next, you need to decide the possible cause of the problem.

If we continue the earlier example, the user reporting being unable to check email, our questioning so far has eliminated many possible causes. We know that the physical network connections are good and our DNS and router configurations are good as well. If the problem is affecting only one user, we can make a good guess that the problem is related to that specific user's system. Obviously something changed in the configuration of that user's system and we can continue our problem solving there by examining that particular system, specifically the email configuration settings.

Step 5: Implement a Solution

After you have determined the most likely cause of the problem, you need to propose and implement a solution. Continuing with the example of being unable to send email, you have determined the problem is the individual user's system. You need to check the user's system configuration and make the necessary changes to restore the service. After you put your solution in place, you need to test the system to be sure the solution solved the problem. Next, you should be sure that implementing your solution has not caused any additional problems. Sometimes solving one problem can cause other problems, and you need to be sure this is not the case with your solution.

Step 6: Keep Documentation

Any time you have a problem, whether you discovered it yourself or it was reported to you by a user, you should always document it. You should fully describe the problem, the steps you took to investigate it, and the solution you implemented. By keeping good documentation about system problems, you can considerably shorten future incidents of troubleshooting similar problems.

Troubleshooting Resources

Whenever you have a problem to troubleshoot, you usually won't be going it alone. There are many sources of information that you can use to help you find a solution to your problem. This section looks at some of the resources available to you listed in what we feel are the most useful order.

The Internet

A great source of information is the Internet. Many times when I have a problem with my system, I use my favorite search engine and do a search for the problem I am having. A problem that occurs with your Fedora Core or Red Hat

Enterprise Linux system has most likely already occurred on another system somewhere. Why not search for the problem to see how others have determined the cause of the problem and its solution. To quote an old saying “Why reinvent the wheel?” If someone else has had the same or similar problem, why not see how they solved it. In most cases your answer is a search away from being solved.

System Log Files

Sometimes you won’t be able to find a solution to your problem on the Internet. Maybe the problem is too new or too specific to your system’s configuration. In this case, you can use your system log files to try to get more information about the problem.

All log files in Fedora Core and Enterprise Linux are located in the `/var/log` directory. Figure 35-1 shows the `/var/log` directory from a typical Fedora Core system.

The log files are plain text files and can be viewed using the `cat` command or by using any text-editing program. You must be logged in as root to view the contents of the log files. A very useful log file is the kernel startup file, which shows all the devices on the system and their configuration status at system boot. If you are having a problem with your system hardware, this is a good place to start your troubleshooting. Another good log file to view is `/var/log/messages`, which contains many of the boot messages and also up-to-date system log information. Listing 35-1 shows an abbreviated version of the contents of the kernel startup log, which can be viewed by issuing the command `dmesg`.

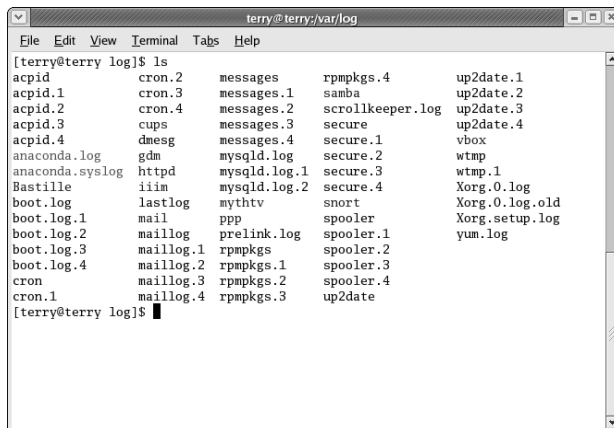


Figure 35-1 The contents of the `/var/log` directory.


```

[root@terry log]# dmesg
Linux version 2.6.10-1.770_FC3 (bhcompile@porky.build.redhat.com) (gcc version
3.4.2 20041017 (Red Hat 3.4.2-6.fc3)) #1 Thu Feb 24 14:00:06 EST 2005
BIOS-provided physical RAM map:
  BIOS-e820: 0000000000000000 - 000000000009fc00 (usable)
  BIOS-e820: 000000000009fc00 - 00000000000a0000 (reserved)
  BIOS-e820: 00000000000e0000 - 0000000000100000 (reserved)
  BIOS-e820: 0000000000100000 - 0000000001fffd0000 (usable)
  BIOS-e820: 0000000001fffd0000 - 0000000001ffff0c00 (reserved)
  BIOS-e820: 0000000001ffff0c00 - 0000000001ffffc000 (ACPI NVS)
  BIOS-e820: 0000000001ffffc000 - 000000000200000000 (reserved)
OMB HIGHMEM available.
511MB LOWMEM available.
Using x86 segment limits to approximate NX protection
On node 0 totalpages: 131024
  DMA zone: 4096 pages, LIFO batch:1
  Normal zone: 126928 pages, LIFO batch:16
  HighMem zone: 0 pages, LIFO batch:1
DMI 2.3 present.
ACPI: RSDP (v000 COMPAQ                                ) @ 0x000f6560
ACPI: RSDT (v001 HP      CPQ0860 0x16080420 CPQ  0x00000001) @ 0x1fff0c84
ACPI: FADT (v002 HP      CPQ0860 0x00000002 CPQ  0x00000001) @ 0x1fff0c00
ACPI: SSDT (v001 COMPAQ  CPQGysr 0x00001001 MSFT 0x0100000e) @ 0x1fff5c3c
ACPI: DSDT (v001 HP      nx7000 0x00010000 MSFT 0x0100000e) @ 0x00000000
ACPI: PM-Timer IO Port: 0x1008
Built 1 zonelists
Kernel command line: ro root=/dev/VolGroup00/LogVol10 rhgb quiet
Initializing CPU#0
CPU 0 irqstacks, hard=c03d3000 soft=c03d2000
PID hash table entries: 2048 (order: 11, 32768 bytes)
Detected 598.268 MHz processor.
Using pmtmr for high-res timesource
Console: colour VGA+ 80x25
Dentry cache hash table entries: 131072 (order: 7, 524288 bytes)
Inode-cache hash table entries: 65536 (order: 6, 262144 bytes)
Memory: 514364k/524096k available (2045k kernel code, 9128k reserved, 655k data,
160k init, 0k highmem)

<-----snip----->

ieee1394: Initialized config rom entry `ip1394'
ohci1394: $Rev: 1223 $ Ben Collins <bcollins@debian.org>
ACPI: PCI interrupt 0000:02:00.0[A] -> GSI 10 (level, low) -> IRQ 10
ohci1394: fw-host0: OHCI-1394 1.0 (PCI): IRQ=[10]  MMIO=[90200000-902007ff]  Max
Packet=[1024]
md: Autodetecting RAID arrays.
md: autorun ...
md: ... autorun DONE.
ieee1394: Host added: ID:BUS[0-00:1023]  GUID[00023f4c4a405266]
SELinux: initialized (dev ramfs, type ramfs), uses genfs_contexts
NET: Registered protocol family 10

```

Listing 35-1 The contents of the kernel startup file dmesg. (*continued*)

```

Disabled Privacy Extensions on device c03697c0(lo)
IPv6 over IPv4 tunneling driver
divert: not allocating divert_blk for non-ethernet device sit0
ACPI: AC Adapter [C134] (on-line)
ACPI: Battery Slot [C11F] (battery present)
ACPI: Power Button (FF) [PWRF]
ACPI: Lid Switch [C136]
ibm_acpi: ec object not found
ACPI: Video Device [C0D0] (multi-head: yes rom: no post: no)
EXT3 FS on dm-0, internal journal
kjournald starting. Commit interval 5 seconds
EXT3 FS on hda2, internal journal
EXT3-fs: mounted filesystem with ordered data mode.
SELinux: initialized (dev hda2, type ext3), uses xattr
SELinux: initialized (dev tmpfs, type tmpfs), uses transition SIDs
Adding 1048568k swap on /dev/VolGroup00/LogVol01. Priority:-1 extents:1
SELinux: initialized (dev binfmt_misc, type binfmt_misc), uses genfs_contexts
    
```

Listing 35-1 (continued)

Both Fedora Core and Enterprise Linux provide a log viewer that you can use to look at your logs. You need to be running the X Window system to use the log viewer. To use the log viewer in Enterprise Linux choose Applications ⇨ System Tools ⇨ System Logs from the desktop menu. In Fedora Core choose Desktop ⇨ System Tools ⇨ System Logs. If you are not logged in as root, you will be prompted to enter the root password. You will see the System Logs main window, as shown in Figure 35-2. In Figure 35-2 the kernel startup log (dmesg) is shown.

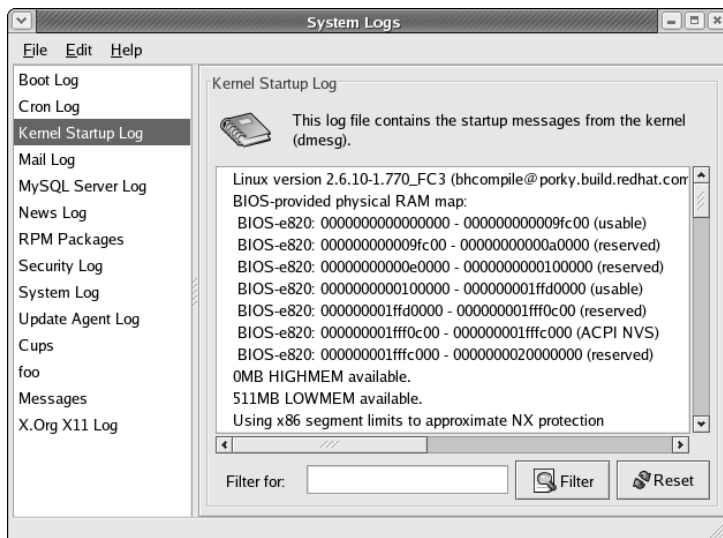


Figure 35-2 The System Logs window lets you choose which log file to view.

To find out what the log file is used for and to view its contents, click the file name. A brief description appears on the top right of the System Logs window. The contents of the selected log file will appear in the large text box under the description of the log file. You can add other log files to the log viewer by doing the following:

1. Click Edit ⇨ Preferences to open the Preferences dialog box shown in Figure 35-3.
2. Click Add, then enter the name you want to be displayed in the log list, a description, if desired, and the path to the location of the log file. For example, to add the Samba log file to the list of logs, enter Samba Log for the name and `/var/log/samba/smbd.log` for the location.
3. Click OK and then Close. The Samba Log entry will appear in the list of log files.

Many of the programs installed on your system will create a directory in `/var/log` to contain their log files. For example, the samba server places its log files in `/var/log/samba`. To view any of the program-related log files change into the appropriate directory to view the files using one of the methods explained.

README Files

README files typically contain information that is very important to know about the software or hardware but for whatever reason isn't included in the documentation. Many times the information in the README is meant to cover the exact problem you may be having.

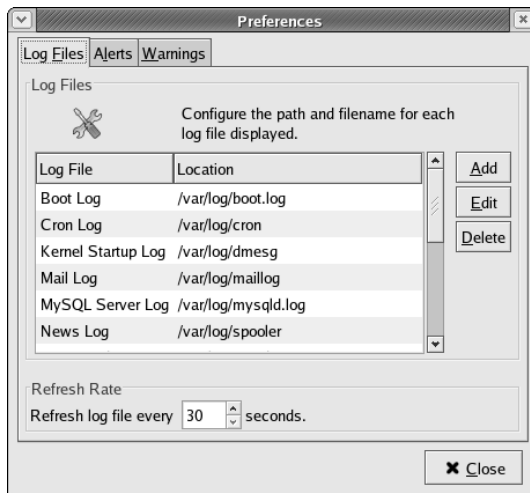


Figure 35-3 The System Logs Preferences dialog box.

You can find installation and configuration instructions, default settings, and other useful tips. You should always read these files whenever you are installing software or hardware on your system, not just when you are having a problem. By reading the README before attempting to install or configure the software or hardware it refers to, you can often avoid the problem to begin with. Many of the system's README files are located in the `/usr/share/doc` directory in a subdirectory named for the program.

Solving Common Problems

The situations discussed in this section include some common problems encountered by new users. Some of the topics covered include not being able to log in to the system and dealing with forgotten passwords. We also look at the most common hardware-related gotchas as well as commands that don't work.

Unable to Log In

A very common problem for a lot of users is not being able to log in. This problem can be caused by a number of problems, but the most common reason is that the user has forgotten his password. Another reason that a user might not be able to log in is because she doesn't have an account on the system that she is trying to use. In either case, you can log in as the root user and reset a user's password, or you can create an account for a user who doesn't have one.

Resetting a User's Password

To reset a user's password, you have to change it to a different password. Do this as follows:

1. Log in as the root user, open a terminal window, and enter the following command:

```
passwd <user name>
```
2. Type a new password for the user, and press Enter.
3. Type the same password again, and press Enter.
4. Log out as root and try logging in as the user to be sure that the login now works.

Creating a User Account

Another reason that a user might not be able to log in is because the user does not have an account. This is easily remedied by creating an account for the user. To create a user account, do the following:

1. Open a terminal window and log in as the root user.
2. At the command prompt, type **useradd *username*** and then press Enter.
3. Type the command **passwd *username***, and then press Enter.
4. Type the password for the user when prompted, and then press Enter.
5. Log out as root and try logging in as the user to be sure that the login now works.

Lost or Forgotten Root Password

If a user forgets his password, you can always log in as root to change the user's password. But what can you do if you forget the root user's password? You won't be able to do a lot of administrative tasks if you can't become the root user when necessary. Fortunately, the procedure for resetting the root password is not difficult:

1. Reboot your system.
2. When the GRUB boot prompt appears, highlight the kernel image that you want to boot and then press **e**.

This opens the GRUB edit window for this kernel.

3. Place the cursor on the second line, which should look similar to the following:

```
kernel /vmlinuz-2.4.22-1.2115.nptl ro root=LABEL=/1
```

4. At the end of the line, insert a space and then type the word **single**.
5. Press Enter and then press **b** to boot the kernel.

After the system boots into single-user mode, it opens a terminal window with a root shell prompt of `init-2.05b#`.

6. At the prompt, type **passwd root** and then press Enter.
7. Type the new password for root and then press Enter.
8. Type **exit** and then press Enter. The system reboots, and you know the new root password. Try not to forget this one.

CD-ROM Drive Not Detected during Installation

Before beginning your installation, you should check the hardware compatibility list at the Red Hat Web site. This will ensure that your hardware is compatible with Fedora Core and Enterprise Linux. But even if you check and find

that your CD-ROM drive is supported, the drive might not be detected by Fedora Core or Enterprise Linux although it is recognized by the system BIOS and you can boot from the drive.

In this case, you can specifically tell the system where to find the CD-ROM drive by following this procedure:

1. Boot your system by using the first installation disk.
2. When you see the boot prompt, type **linux hdx=cdrom** and then press Enter.

This command specifically tells the kernel where to look for the CD-ROM drive. x should be one of the following:

- a: First drive on primary IDE controller (master)
- b: Second drive on primary IDE controller (slave)
- c: First drive on secondary IDE controller (master)
- d: Second drive on secondary IDE controller (slave)

Your system should now see the CD-ROM drive, and you can do the installation.

CD-ROM Drive Does Not Mount after Installation

On some systems, the CD-ROM drive will not mount after the installation. If you encounter this problem, try this solution:

1. Boot your system from a backup boot disk. Read how in the upcoming section “Making an Emergency Boot Disk.”
2. At a terminal command line, type the command **depmod -ae** and then press Enter. This command creates a dependency file of the system modules. When the system boots, this file is used by the **modprobe** command to automatically load the proper modules.

Sound Does Not Work after Installation

The Fedora Core and Enterprise Linux installation program does a good job of detecting the hardware on your system and attempting to configure it properly. Most of the time, your sound card will be detected and the proper modules loaded so that it works. Sometimes, though, especially on older systems, the installation program might not be able to detect your sound card. Or your sound card might be detected but not configured properly. Here are some suggestions that you can try to get your sound to work:

- **Try running the automatic configuration program again** — Sometimes running it again after logging in as root can correct the problem. Choose Applications ⇨ System Settings ⇨ Soundcard Detection. If your sound card is detected, you are given the option to test the configuration. If your sound works during the test, that's great: You're finished.
- **Edit the `/etc/modprobe.conf` file** — If the sound doesn't work or if your sound card isn't even detected, you can manually try some settings by editing the `/etc/modprobe.conf` file. The `/etc/modprobe.conf` file from my system is shown in Listing 35-2.

```
[terry@terry terry]$ cat /etc/modprobe.conf
alias parport_lowlevel parport_pc
alias eth0 e100
alias usb-controller usb-uhci
alias sound-slot-0 snd-cs4232
options sound dmabuf=1
alias synth0 opl3
options opl3 io=0x388
options cs4232 isapnp=1
```

Listing 35-2 Sound module and configuration information of the `/etc/modprobe.conf` file.

NOTE A good source of information about configuring sound cards in Linux is the **Advanced Linux Sound Architecture Web site** at www.alsa-project.org.

When I installed Enterprise Linux on one of my older systems, it could not detect my sound card during installation. After I logged in, running the sound card detection tool still did not detect my sound card. I had to manually edit the `/etc/modprobe.conf` file to configure my sound card, as shown in Listing 35-2. The sound configuration information begins with the line `alias sound-slot-0 snd-cs4232`. The lines following it are options for the `cs4232` sound card.

To configure sound on your system, you need to know the type of sound card installed in your system as well as the configuration options for the sound card. The best place to get specific information for your card and configuring it under Enterprise Linux is the ALSA Web site at www.alsa-project.org. You can find information about nearly every sound card there as well as specific instructions to configure the card. With a little effort on your part and some reading at the ALSA site, you can get your sound working.

TIP Another common problem is that sound is working but muted. Run `alsamixer` and make sure that the Main Stereo and PCM variables are not muted or with `volume=0`.

Unable to Unmount a Drive

Whenever you work with a floppy disk or CD-ROM, you have to mount the disk before you can read or write to it. After you finish, you no longer need the drive mounted. In fact, if you want to remove the CD from the drive, you must unmount the disk or the CD-ROM drive won't let you eject the disk. Look at an example. You finish using the CD-ROM drive, and you try to unmount the drive by using the `umount` command, as follows:

```
[root@terry cdrom]# umount /mnt/cdrom
```

The system returns a message telling you that the drive is busy:

```
umount: /mnt/cdrom: device is busy
```

Although this is an easy problem to solve, it is a real source of frustration for a new user. You can see from the example prompt that the current working directory is `cdrom`. If you use the `pwd` command

```
[root@terry cdrom]# pwd
```

you see that you are in the directory that you're trying to unmount.

```
/mnt/cdrom
```

All you need to do is to change to a different directory and then run the `umount` command again. Now you can successfully unmount the drive.

I have also noticed a similar problem that isn't related to the current working directory. Sometimes when trying to unmount a drive, you receive the Drive Busy error message even though you're not in the directory that you're trying to unmount. No matter what you try, you can't get the `umount` command to unmount the drive. Check that you don't have any background jobs running that were started when you were in the `CDROM` directory. This may be keeping you from unmounting the drive.

NOTE After some thought one day, I decided to try logging out of the desktop and then logging back in. I thought that perhaps there was some glitch in the GNOME or KDE desktop that was not allowing the drive to unmount. Sure enough, this method did work because when I logged back in, the drive was no longer mounted. If you experience a similar situation when trying to unmount a drive, give this method a try.

Shell Commands Don't Work

This is a common problem encountered by new users who log in as a regular user and then execute the substitute user command to become the root user. The problem is that after entering the `su` command and becoming root, commands issued return the message `Command Not Found`.

This problem is caused because even though the `su` command lets a regular user become the root user with all of root's permissions, it does not give root an actual login shell. So in this case, the user has root's permissions but does not have root's search path for directories. For example, when I use `su` to become root on my system and then run a command (`chkconfig`, for example), I receive the following:

```
[terry@terry terry]$ su
Password:
[root@terry terry]# chkconfig --list
bash: chkconfig: command not found
[root@terry terry]#
```

To solve this problem, you can do two things:

- Enter the complete path to the command.
- Use the `su` command with the `-` (hyphen) option as shown here:

```
[terry@terry terry]$ su -
Password:
[root@terry root]# chkconfig --list
ntpd                0:off    1:off    2:off    3:off    4:off    5:off    6:off
syslog              0:off    1:off    2:on     3:on     4:on     5:on     6:off
<output clipped>
```

Another instance of shell commands not working is also related to the search path. Users who are familiar with DOS know that DOS looks in the current directory for executable files as well as the search path. Linux, on the other hand, does not search the current directory but only the search path. Thus, an executable file in the same directory as the directory the user is in will not be found.

You need to explicitly tell the shell to look in the current directory by using `./ (executable file name)`. The period and forward slash tells the shell to look in the current directory.

Solving File System Problems

The tips in this section help you solve common problems you might encounter when working with Linux files and file systems.

Cannot Delete a File

If you cannot delete, move, or rename a file, use the `ls -l` command to verify that you have permission to do so. If you receive an error message resembling `rmdir: `dirname': directory not empty` when using `rmdir` to delete a directory, either delete the files in the directory before retrying the `rmdir` command or use `rm -r` to delete the directory and its contents. For example:

```
$ rmdir images
rmdir: `images': directory not empty
$ rm -r images
```

A common question is how to delete a file that has a name beginning with a minus sign (-). Suppose, for example, a file in your home directory is named `-foo`. The command `rm -foo` fails and generates the following error:

```
$ rm -foo
rm: invalid option -- o
```

This problem occurs because most Linux commands interpret a minus sign followed by a letter as a command option, and anything following that as an argument to the command. So, the argument to the `rm` command, `-foo`, looks like invalid syntax, not a filename. The solution is to use two minus signs (- -) between `rm` and `-foo`. Most Linux commands interpret two minus signs standing alone as the end of all options; everything afterward is interpreted as an argument. So, to remove the file named `-foo`, try `rm -- -foo`, as follows:

```
$ rm -- -foo
```

TIP To delete a file that has a name beginning with -, you can also use the following:

```
$ rm -- '-foo'
```

Similarly, to delete a filename that contains spaces, you can use a command similar to the following:

```
$ rm -- '-a file name containing spaces'
```

Commands with Multiword Arguments

If you pass an argument made up of multiple words to a command and get an error message stating, in part, `No such file or directory`, enclose the argument between weak quotes (" ") and try again. Remember that most shells use spaces and tab characters to distinguish between commands, options, and arguments. For example, the following command generates an error message before showing its result:

```
$ grep Kurt Wall /etc/passwd
grep: Wall: no such file or directory
/etc/passwd:kwall:x:500:500:Kurt Wall:/home/kwall:/bin/bash
```

Enclosing Kurt Wall within weak quotes solves the problem:

```
$ grep "Kurt Wall" /etc/passwd
/etc/passwd:kwall:x:500:500:Kurt Wall:/home/kwall:/bin/bash
```

Accessing Windows File Systems

If you get an error message that the Virtual File Allocation Table (VFAT) file system, the Windows file system, is not supported when using the `mount` command to mount a DOS/Windows floppy disk, log in as the root user and execute the following command:

```
# /sbin/modprobe vfat
```

This command loads the necessary modules that Red Hat Linux needs in order to provide support for DOS/Windows file systems.

Working with Floppy Disks

If you get a `device busy` error message when trying to unmount a floppy disk with the `umount` command, make sure that your current directory is not located on the floppy disk (use the `pwd` command). Red Hat Linux cannot unmount a file system or disk of any sort if a process is using it, which includes having a current working directory located anywhere on the file system.

If you cannot use the `cd` command to change your current working directory, execute the following command to see the process IDs (PIDs) of any processes using the mount point:

```
# fuser /file-system-name
```

Next, use the following command:

```
# ps -p pid
```

Replace `pid` with the PIDs from the previous command, to see what process or processes are still using the file system.

If you accidentally remove a floppy disk from a mounted drive without first unmounting it, in many cases you can simply reinsert the floppy disk and execute the appropriate `umount` command to unmount the drive properly.

Cannot Mount a Partition

If you are unable to mount one or more partitions after an upgrade, the problem could be due to the use of partition labels. Red Hat Linux uses partition labels to help prevent problems that occur when moving hard drives around in a system. For example, if `/` is on `/dev/sdb` and a new drive is added to the system with a lower SCSI ID, the drive that was previously `/dev/sdb` becomes `/dev/sdc`. This change causes the mount command to generate errors when it tries mount `/`. Partition labels address this problem because mount tries to mount the partition identified by its labels as `/` instead of trying to mount `/dev/sdb1` and expecting it to be `/`. The idea is great in theory as long as you keep partition labels less than 16 characters.

Avoiding File System Checks at Each System Reboot

If you constantly have to run `fsck` each time you boot your system, the most likely suspect is that your partitions are not being unmounted properly when you last shut down the machine. The most important thing you can do is make sure that you are shutting down the machine properly. Shutdown is done through one of two methods:

- If you are in text mode — run level 3 — you should log in as root and type the following command:

```
# sync;sync;sync;shutdown -r now
```
- If you are using the graphical login — run level 5 — click Actions ⇄ Log Out ⇄ Shutdown. Eventually, you see a line that reads `Power Down`, indicating it is safe to turn off the machine.

In either case, if you are running APM or ACPI, Linux tries to stop the machine using the BIOS.

Solving Networking Problems

There are many possible causes of problems with your network and troubleshooting them can be a tough task. But if you follow a systematic procedure as explained earlier in this chapter you can get through it and solve your problem. Some general guidelines to follow for troubleshooting network problems are presented here:

1. Don't overlook the simple things. For example, if a user calls complaining that he cannot log in, be sure that the user is using the proper login procedure with the correct username and password. Does the user have the correct rights assigned for the service? Check the cable connections and look for link and collision lights.

2. Determine whether the problem is software or hardware related. A hardware problem usually appears when some device stops working. Software problems may give an error message indicating the problem.
3. Does the problem affect one workstation or a group of them? If the problem affects just one user, the problem is most likely at that workstation. If many users are affected, the problem could be at the server.
4. After you have eliminated physical connections as the source of the problem, you can then begin checking workstation and server configuration files.

If you are unable to access hosts on a network, make sure you have at least one name server listed in `/etc/resolv.conf`. If an external system cannot connect to your system, remember the order in which the `hosts.allow` and `hosts.deny` files are read and applied. The program that starts Internet services consults `/etc/hosts.allow` first. If TCP wrappers does not find a matching rule, it applies the first matching rule, if any, in `/etc/hosts.deny`. If neither file contains a match, access is granted.

If you performed an upgrade of Red Hat Linux from an earlier version, it may appear that the upgrade did not preserve the contents of your network service configuration files. This is not the case, however. The configuration files were renamed with the extension `.rpmsave` and can be found in the network service's directory. For example, you might find that the old `/etc/httpd/conf/httpd.conf` file was renamed `/etc/httpd/conf/httpd.conf.rpmsave`. Changes should be copied to the new version of the configuration file. The following list explains the meaning of the saved files.

- Files with an `.rpmorig` extension are configuration files not in an RPM package that was upgraded or installed.
- Files with an `.rpmsave` extension are configuration files that were in an RPM package that was upgraded or installed.
- Files with an `.rpmnew` extension are new versions of files installed by an upgraded package.

Affected network services configuration files include, but are not limited to:

- `/etc/httpd/conf/httpd.conf`
- `/etc/samba/smb.conf`
- `/etc/sendmail.cf`
- `/etc/named.conf`
- `/etc/dhcpd.conf`

To avoid unfortunate accidents and needless panic or confusion, remember to back up your data before upgrading or installing the latest version of Fedora Core or Red Hat Enterprise Linux.

Getting Online with a Modem

If you are having trouble getting your modem to work, first verify that it is supported by checking the Hardware Compatibility List at <http://hardware.redhat.com/>.

CROSS-REFERENCE Refer to Chapter 11 for instructions on using the graphical tool to configure your modem.

Next, see if the modem is being detected by the system and make sure that it does not conflict with other resources. You can check this using the following commands:

```
# cat /proc/ioproports
0000-001f : dma1
0020-003f : pic1
0040-005f : timer
0060-006f : keyboard
0070-007f : rtc
00a0-00bf : pic2
00c0-00df : dma2
00f0-00ff : fpu
0170-0177 : ide1
01f0-01f7 : ide0
0220-022f : soundblaster
02f8-02ff : serial(auto)
0330-0333 : MPU-401 UART
0376-0376 : ide1
03c0-03df : vga+
03f6-03f6 : ide0
03f8-03ff : serial(auto)
d000-d07f : eth0
d800-d807 : ide0
d808-d80f : ide1

# cat /proc/interrupts
CPU0
0:      1296380          XT-PIC  timer
1:       30736          XT-PIC  keyboard
2:         0          XT-PIC  cascade
5:         1          XT-PIC  soundblaster
8:         1          XT-PIC  rtc
0:       73593          XT-PIC  eth0
2:      159669          XT-PIC  PS/2 Mouse
3:         1          XT-PIC  fpu
4:      246863          XT-PIC  ide0
5:      584998          XT-PIC  ide1
NMI:         0
```

One example of a resource conflict is that your modem and some other device share an interrupt. COM1 (`/dev/ttyS0`) and COM3 (`/dev/ttyS3`) try to share the same interrupt unless told otherwise. Eliminate the conflict by setting a jumper for one of the devices that causes it to use another IRQ. Next, use `minicom` to see if you can communicate with the modem. The following Web pages help you configure a PPP connection:

- redhat.com/support/docs/tips/PPP-Client-Tips/PPP-Client-Tips.html
- redhat.com/support/docs/tips/Network-Config-Tips/Network-Config-Tips.html

If `setserial` shows your modem's UART as unknown, as you see in the following example, the kernel has not detected your serial port or the modem attached to it:

```
/dev/ttyS2, UART: unknown, Port: 0x03f8, IRQ: 10
```

This problem usually occurs because your PC's BIOS is set up to expect a plug-and-play (PnP) operating system. To solve the problem, reboot the PC and, as it powers up, press the key that permits you to access the system's BIOS. Typically, this is a function key, such as F2, but the exact key depends on your PC's BIOS. In the setup screen, locate the option for PnP operating system (often labeled Plug & Play O/S) and turn off that option. Then save the BIOS settings and exit. Doing so causes the PC to reboot. This time, when Red Hat Linux boots, the kernel should be able to detect the PC's serial port correctly.

If your serial ports are correctly detected, but the modem does not respond, make sure that `/dev/modem` is linked to the proper device file in `/dev`. An unresponsive modem is especially common with PCI modem cards, which typically do not use COM1 (`/dev/ttyS0`) or COM2 (`/dev/ttyS1`) by default. For example, suppose `dmesg | grep ttyS` shows the following:

```
ttys02 at port 0x6800 (irq = 10) is a 16550A
```

In this case, execute the following command to make sure that `/dev/modem` is linked to the proper device file:

```
# ln -sf /dev/ttyS2 /dev/modem
```

If your modem does not appear to be detected, use the Windows Device Manager to obtain the modem's IRQ and I/O address, and compare those values to the values the `setserial` command reports for that port.

At present, some modems simply do not work with Linux. These are so-called "WinModems," which, in order to function, rely on the Windows

operating system and a special, Windows-specific device driver. WinModems, also called *software modems*, rely on a device driver, rather than hardware, to function. Don't despair, you may still be able to use your WinModem. You can search for your modem at <http://linmodems.org> to see if there is a Linux driver available. If there is, you can download, install, and configure it using the instructions provided at the Linmodem site. If you can't find your modem listed there, or if it is listed as unsupported, your only recourse is to replace the modem with a hardware modem or one that is listed as supported on the Linmodem site.

If you can connect to your ISP but are unable to surf the Web, or if your Firefox complains that it cannot connect to remote hosts, the problem is most likely unconfigured or misconfigured Domain Name Server (DNS) information. You need to specify your ISP's DNS servers in the `/etc/resolv.conf` file. Contact your ISP for this information and edit the file to include those settings. For example:

```
search example.com
nameserver 24.8.89.15
nameserver 24.8.89.16
```

The Boot Process Hangs

A network problem exists if your system boots but then seems to hang when starting `sendmail`, the Sendmail daemon; `httpd`, the Apache Web server daemon; or `smb`, the Samba daemon. The most common cause is that Linux cannot resolve the host name to an IP address. The apparent hang is a pause while the kernel waits for the name resolver to time out; the boot process *will* eventually complete. To solve this problem, wait until you can log in, then log in as root to investigate and solve the problem. If you are attached directly to a network with a functioning DNS server, make sure that the file `/etc/resolv.conf` has the correct values for your system's DNS server(s). Make sure that the values are correct. If you are using Red Hat Linux on a system attached to a network without a DNS server, or if your Red Hat system is destined to be the DNS server, edit the `/etc/hosts` file and insert your system's IP address and name to have the hostname and IP address so that the lookups occur correctly. The format of the `/etc/hosts` file is:

```
127.0.0.1      localhost.localdomain localhost
192.168.0.1    bubba.somedomain.com bubba
```

Replace `bubba.somedomain.com` with the proper name of your system and the IP address with the IP address of your system.

Using Two Ethernet Cards

To use two Ethernet cards in your Red Hat Linux system, first ensure that both cards are supported. Next, if the two cards use different drivers, you need to set up the second network interface and edit the `/etc/modules.conf` file to load the proper driver for the second card. If the two cards use the same driver, you may need to recompile your kernel, but several modules now allow for multiple cards. It may be that you just need to use boot arguments, such as:

```
boot: linux ether=11,0x300,eth0 ether=5,0x340,eth1
```

This option can be made permanent so that you do not have to reenter it every time your system boots. See the GRUB configuration option `append=` in the `grub.conf` man page. The Ethernet HOWTO is an excellent source of information for configuring multiple Ethernet cards in the same system. It can be found at redhat.com/mirrors/LDP/HOWTO/Ethernet-HOWTO.html.

Solving NFS Problems

In addition to performance degradation, you might encounter other problems with NFS that require resolution. This section discusses some of the typical difficulties system administrators have with NFS and how to resolve them. These tips are only a starting point.

NOTE The Linux NFS-HOWTO on the NFS Web page at <http://nfs.sourceforge.net/nfs-howto/> **dedicates an entire section to troubleshooting NFS.**

First up are apparent problems that, in reality, are red herrings. Log messages resembling the following are annoying but harmless:

```
kernel: fh_verify: bubba/users: permission failure, acc=4, error=13
kernel: nfs: server localhost not responding, still trying
kernel: nfs: task 18273 can't get a request slot
kernel: nfs: localhost OK
nfslock: rpc.lockd startup failed
kmem_create: forcing word size alignment - nfs_fh
```

The first message occurs when the NFS `setattr()` RPC call fails because an NFS client is attempting to access a file to which it does not have access. This message is harmless, but many such log entries might indicate a systematic attempt to compromise the system.

The next three messages represent client attempts to contact the NFS server that are timing out. When such timeouts occur, the NFS client reduces the

number of concurrent requests it sends to avoid overloading the server, which results in these messages. Although such messages are usually harmless, if they persist, you might want to investigate possible saturation of the network or the NFS server.

The `rpc.lockd` startup failure message almost always occurs when older NFS startup scripts try to start newer versions of `rpc.lockd` manually; these attempts fail because the kernel NFS server daemon, `knfsd`, starts the locking daemon automatically. To make the failure message go away, edit the startup scripts and remove statements that attempt to start `lockd` manually. The final error message occurs if the kernel detects that an NFS file handle is 16 bits, rather than 32 bits or a multiple thereof.

Would that all error messages were as harmless as these! A log message resembling the following, while not dire, demands timely attention:

```
nfs warning: mount version older than kernel
```

This message means exactly what it says: the version of the `mount` command that you are using is older than the kernel version. So, the solution is to upgrade the `mount` package so `mount` can recognize any additional options or features of the new kernel.

If you transfer very large files via NFS, and NFS eats all of the available CPU cycles, causing the server to respond at a glacial pace, you are probably running an older version of the kernel that has problems with the `fsync()` system call that accumulates disk syncs before flushing the buffers. This issue, primarily a problem with 2.2 kernels and early 2.4 kernels, is reportedly fixed in newer 2.4 kernel releases, so upgrading your kernel might solve the problem.

Similarly, if you execute commands on an NFS exported file system that do not result in large data transfers from server to client (such as an `ls` command) and have no problem, but then nevertheless cause severe response problems with large data transfers (such as a `cp` command), you may be using `rsize=` or `wsiz=` parameters on the client that are larger than the `rsize=` or `wsiz=` parameters on the server. Reduce these values on the client side to see if performance recovers. Also make sure that the firewall for the client and the server permits fragmented packets to pass through. NFS uses packet fragmentation, so firewalls that deny or drop fragmented packets force constant retransmission of data. Reportedly, this is especially a problem on Linux systems still using the 2.2 packet filter, `ipchains`. Either switch to `iptables` or rewrite the `ipchains` filters to accept fragmented packets; performance should improve almost immediately.

If you are unable to see files on a mounted file system, check the following items:

1. Make sure that the file system *is*, in fact, mounted. If the file system is not mounted, mount it. Use one of the following commands to verify that the file system is mounted:

```
# cat /proc/mounts
# mount -f
```

2. If the file system is mounted, make sure that another file system is not mounted on top of it. If you have layered a file system on top of an export, unmount and remount both, making sure to mount them on separate mount points.
3. Make sure the client has read privileges to the specified files by verifying the file system permissions on the server, the client's mount point, and that the mount options in `/etc/fstab` or specified on the command line are correct on the client.

If you cannot mount an exported directory, the most common error message is:

```
mount failed, reason given by server: Permission denied
```

This error message means that the server thinks the client does not have access to the specified export. In this case, do one or more of the following:

- Review the export specifications in `/etc/exports` on the server, making sure that they are correct and that the client is mounting the export the same way it is exported. For example, an NFS client cannot mount an exported directory read/write (the `-o rw` option to the `mount` command) if that directory is exported from the server read-only (using the `-o ro` option to the `mount` command).
- On the server, execute the following command:

```
# exportfs -ar
```

This command makes sure that any changes made to `/etc/exports` since the exports were first exported are updated in the server's exports table and propagated out to NFS clients.

Look at the contents of `/var/lib/nfs/xtab` to review the complete list of all of the export options applied to a given export. If they are incorrect, edit `/etc/exports` accordingly and then rerun the `exportfs -ar` command to update the server's export table with the proper options.

Exploring Miscellaneous Problems

The tips suggested in this section address some problems and challenges you might face when booting your system, installing on a laptop, and using the X Window system with Fedora Core and Red Hat Enterprise Linux.

Solving Boot Problems

If you try to shut down or reboot your Red Hat system using the commands `reboot`, `halt`, `shutdown -r now`, or `shutdown -h now` and the shutdown process starts to execute correctly but then the display blanks and the system hangs, the only way to recover is to power cycle the system and then try some of the following workarounds. The problem is that at the point the system appears to hang, control of the hardware has been handed back from Linux to the firmware — it is up to the firmware (software embedded in key system hardware components) to reboot the system correctly. Fortunately, Linux enables you to select multiple ways to reboot the system to fix, or at least side-step, buggy or broken BIOSes or hardware.

At the GRUB: boot prompt, you can specify:

```
reboot=X,Y
```

X can be one of `hard` or `bios`.

- **hard** — Uses the CPU's reset instruction to restart the system.
- **bios** — Uses a BIOS routine (sometimes called a *BIOS vector*) to restart the system.

Y can be one of `warm` or `cold`.

- **warm** — A *warm boot* is the type of reboot invoked when you press Ctrl+Alt+Delete.
- **cold** — A *cold boot* is the type of boot invoked by power cycling the system.

So if you boot with the following command, Linux reboots by the BIOS vector with a warm reboot:

```
boot: linux reboot=bios,warm
```

The goal is to find the right combination of X and Y that triggers the bugs in the system BIOS. Once you have found this magic sequence, use GRUB's `append=` option to pass these parameters to the kernel each time you boot the system by adding it to `/etc/grub.conf` as shown in the following example:

```
append="reboot=bios,warm"
```

If your system installed without incident until it tried to write GRUB information to the master boot record (MBR), at which point the installer complained that it could not write to the MBR, the MBR may be locked by the BIOS. You need to access your system's BIOS and verify that the MBR is not

write-protected. Similarly, disable any virus scan enabled in the BIOS that may interfere with writing to the MBR.

ht://Dig Won't Run

If you start the `rundig` program to create `ht://Dig`'s databases and search indexes, you might see the following error, or one that closely resembles it:

```
# /usr/bin/rundig
/usr/bin/rundig: line 101: 21924 Segmentation fault      /usr/bin/htfuzzy $verb
ose metaphone
/usr/bin/rundig: line 102: 21925 Segmentation fault      /usr/bin/htfuzzy $verb
ose soundex
```

This error occurs if the file `/var/www/html/index.html` does not exist, even if the directory `/var/www/html` contains other content. The workaround is to create an index file. At a bare minimum, you can create an empty `index.html` using the `touch` command:

```
# touch /var/www/html/index.html
```

The `rundig` command will then execute properly.

Starting cyrus-imapd

If you see the following message when you start `cyrus-imapd`, neither the problem nor the solution actually has *anything* to do with the mail server, despite what the error message says:

```
# service cyrus-imapd start
Starting cyrus-imapd: preparing databases... error!           [FAILED]
```

The problem occurs because `cyrus-imapd` converts a binary file, `/var/lib/imap/mailboxes.db`, to a flat file when it starts. The log file that tells you what happened is `/var/lib/imap/rpm/cvt_cyrusdb_all.log`:

```
you are using /var/lib/imap/sieve as your sieve directory.
fatal error: can't open old database
Converting from /var/lib/imap/mailboxes.db (berkeley) to
/var/lib/imap/mailboxes.db.flat (flat)
ERROR: unable to convert /var/lib/imap/mailboxes.db from berkeley to
flat
fatal error: can't open old database
```

The issue, it turns out, is that the database conversion uses the `file` utility to determine the file type of the `mailboxes.db` file. Unfortunately, the `file` utility incorrectly classifies `mailboxes.db` as an Apple QuickTime movie!

```
# file /var/lib/imap/mailboxes.db
/var/lib/imap/mailboxes.db: Apple QuickTime movie (modified)
```

file misidentifies mailboxes.db because file's own database file, /usr/share/file/magic.mgc, has an invalid entry. The good news is that you can easily fix this problem by editing /usr/share/file/magic. The procedure is simple:

1. Edit /usr/share/file/magic, moving the line (near line 6461) that reads:

```
0 string          \241\002\213\015skiplist\ file\0\0\0      Cyrus
skiplist DB
```

above the line that reads (near line 819):

```
# Apple Quicktime and ISO types
```

2. Execute the command `file -C` to update the binary `magic.mgc` file that the file command uses:

```
# cd /usr/share/file
# file -C
# file /var/lib/imap/mailboxes.db
/var/lib/imap/mailboxes.db: Cyrus skiplist DB
```

After you have made this change, the `cyrus-imapd` service should start. As a convenience, you can use the patch file `magic.patch` and the following patch command to update the magic file without having to edit it manually:

```
# patch -d /usr/share/file < magic.patch
patching file magic
```

Whether you use the patch file or edit /usr/share/file/magic manually, you must execute the `file -C` command (and do so in /usr/share/file) to update the `magic.mgc` file or the change won't take effect. For more information about this bug, see https://bugzilla.redhat.com/bugzilla/show_bug.cgi?id=148808.

Solving Laptop Video Problems

Laptop installations are typically the most difficult type of installation to perform because the companies that build laptop computers often use proprietary hardware or modify standard PC components to shoehorn desktop PC functionality into the confines of a laptop or to meet weight, power, or functionality requirements. Aggravating such practical concerns, to protect trade secrets, these engineering decisions are rarely publicly documented. As a result, you often have to use a trial-and-error method and rely on the experiences of others.

NOTE A few laptop manufacturers actively support Linux. IBM, for example, preinstalls Linux on a select group of laptop computers, and certain dedicated Linux hardware vendors do the same. The lack of support *is* frustrating, but until manufacturers can be bothered to develop Linux drivers for their products at the same time as they develop Windows drivers, installing Linux on a laptop computer will continue to be a challenge. A good source of information about running Linux on laptops can be found at linux-laptop.net.

While attempting a graphical Red Hat Linux installation from either the CD-ROM or a floppy disk, you may see the laptop screen go blank and you cannot continue. In this case, attempting a text-based installation might work — trying to force a graphical installation rarely works if the installer cannot use your video hardware. With *some* laptops, the graphical installer *might* work if you type the following parameter at the `boot :` prompt:

```
boot: linux vga=2
```

Similarly, you can try all of the possible VGA modes using the following parameter:

```
boot: linux vga=ask
```

Before giving up completely, have a look at the authoritative reference for Linux and laptops, the Linux Laptop Web site at linux-on-laptops.com.

The Signal 7 and Signal 11 Problems

Perhaps the most confusing problem people run into when installing Red Hat Enterprise Linux is an error message resembling `fatal signal 11` or `fatal signal 7`. Signal 11s and signal 7s are errors indicating a hardware problem in memory or on the system's data bus. Red Hat Linux does not cause such errors. Rather, it brings such problems to light because the Linux kernel typically pushes hardware to the fullest extent of its capabilities, much more so than DOS or Windows, often revealing substandard hardware.

How should you proceed? The first thing to do is perform memory testing using `memtest86`, which is included on the Red Hat installation boot disk. Instructions for using `memtest86` on the boot disk are included in the release notes. If `memtest86` indicates hardware trouble, take your computer to a repair shop and ask them to test the RAM and, possibly, the CPU cache, on a hardware tester.

Meanwhile, check to see whether you have the latest installation image from Red Hat. If the latest image still fails, the problem may be hardware-related. Common suspects include bad RAM chips or defective CPU cache memory. Try turning off the CPU cache in the BIOS and see if the problem goes away.

Likewise, try swapping memory around in the system's memory slots to see whether the error is slot or memory related. If that does not solve the problem, the Signal 11 Web site, <http://www.bitwizard.nl/sig11>, may be able to help you.

Using Screensavers and Power Management

To disable screen blanking, turn off your screen saver. In text mode, the kernel turns on the screen blanker after 15 minutes, but you can disable this using the following command:

```
# setterm -powersave off -blank 0
```

If you hear disk drives speed up or other sounds, this is most likely APM (Advanced Power Management) starting up the system after idle time. You can disable APM from starting at boot time by logging in as root and typing `system-config-services`. Deselect APM, exit `system-config-services`, and reboot the machine. APM is one of the few Red Hat Linux services that requires a system restart to make it take effect; APM is a low-level kernel function, so a full reset is needed.

Starting the X Window System

What should you do if you run `startx` and get a black screen? To get out of the black screen mode, try pressing `Ctrl+Alt+Backspace`. This keystroke combination causes the X server to exit if possible. If it does not work, reboot the system and reconfigure the X Window system using `redhat-config-xfree86` after making sure that all your video hardware is compatible. You may want or need to obtain the latest version of XFree86 from redhat.com/support/errata. Upgrading X is fairly simple, but an upgrade HOWTO is available at the Red Hat Web site at redhat.com/support/docs/howto/XFree86-upgrade/XFree86-upgrade.html.

If you get an error message resembling `errno 111` when you run `startx`, an X client (any X program except the X server itself running on your XFree86 X system, such as terminal window or even the window manager) tried to connect to the X server but failed to do so for some reason. Unfortunately, you ordinarily see only the last few lines of the error message. To see the complete message, execute the following command:

```
$ X -probeonly >& startx.out
```

This command creates a file named `startx.out` that contains the complete error message. Review the text of the error message carefully for clues concerning the real problem that X is having.

Making an Emergency Boot Disk

Sometimes you might have a problem with your system that prevents it from booting properly. Maybe you made some configuration changes and then shut down your system. When you tried to boot up again, the system wouldn't start because you changed the boot loader configuration files. This is just the time when you could use an emergency rescue disk. In this section, you learn how to make one.

Making a boot disk is very easy. Follow this procedure:

1. From a terminal command line, find your kernel version by entering the command:

```
[root@main root]#uname -r
```

which gives the following output. (Your output might be different.)

```
2.6.9-5.EL
```

2. Type the command **mkbootdisk 2.6.9-5.EL**.
3. Insert a floppy disk into the disk drive when prompted and then press Enter. When the command prompt returns, the boot disk has been created.

Summary

This chapter offered numerous tips and techniques for overcoming commonly encountered problems installing, configuring, and using Fedora Core and Red Hat Enterprise Linux. You first read about how to solve installation problems, such as not being able to mount a CD-ROM after the postinstallation reboot. Next, you learned how to work around problems accessing files and using Windows file systems. After you explored ways to resolve difficulties getting online using a modem, you read about disabling power management and working through problems starting the X Window system.

Bash Shell Scripting

IN THIS CHAPTER

- Using Wildcards and Special Characters
- Using Variables
- Using Bash Operators
- Understanding Flow Control
- Using Shell Functions
- Processing Input and Output
- Working with Command Line Arguments
- Using Processes and Job Control

System administration is frequently performed on a repetitive or ongoing basis. As a result, such duties are prime candidates for automation because repeatedly typing the same commands is tedious, error prone, and time-consuming. This appendix teaches you to use Bash's programming language to automate standard system administration tasks. After you read an overview of the fundamentals of Bash programming, you will learn how to use some very useful shell utility programs that ease the task of shell scripting. You will also receive some guidance in selecting an alternative scripting language if you do not like Bash or find its abilities insufficient for your needs. The discussion assumes that you are comfortable using Bash as an end user and concentrates on Bash features from a shell programmer's perspective.

As a programming language, Bash has all the features one would expect: wildcards, variables, operators, functions, and input and output capabilities. Although Bash's programming support is not as complete, fully featured, or powerful as traditional programming languages such as C and C++ or advanced scripting languages such as Perl and Python, Bash is nevertheless surprisingly capable and well suited to the task.

Using Wildcards and Special Characters

Wildcards are single characters that stand for or substitute for one or more values. Bash uses the familiar `*` and `?` as wildcards. `*` stands for one or more characters and `?` represents any single character. For example, the command `ls d*` executed in `/bin` on a Red Hat system should result in output resembling the following:

```
$ ls d*
date dd df dmesg dnsdomainname doexec domainname dumpkeys
```

The command `ls d?`, however, shows only the commands beginning with `d` followed by any single alphanumeric value:

```
$ ls d?
dd df
```

The command `ls ??` shows all files with names consisting of any two alphanumeric characters:

```
$ ls ??
cp dd df ed ex ln ls mt mv ps rm sh su vi
```

The `*` and `?` characters are useful, but somewhat blunt tools. Bash also supports set operators or set wildcards. Set operators allow you to define a range or set of characters to use as wildcards. The notation for defining a set is `[set]`, where `[` and `]` delimit the range and set lists the alphanumeric characters making up the range. The range can be inclusive, disjoint, discrete, or a combination of all three. An inclusive set includes all the characters in the set and is defined using a hyphen, for example, `[b-f]`. A disjoint set is at least two inclusive ranges separated by a comma (`,`), such as `[1-4, 7-0]`. A discrete set refers to a simple list of characters, such as `[13579]`. Table A-1 lists some examples of the kinds of sets you can create using the `[]` set operator.

For the second set in Table A-1, possible matches include 17, 38, and 49, but 57 and 94 do not match. The next few commands show a few examples of using this set notation in the `/bin` directory on a Fedora Core or RHEL system.

```
$ ls [b-c]*
basename bash bsh cat chgrp chmod chown cp cpio csh cut
```

The resulting set consists of any filename beginning with `b` or `c`.

```
$ ls [b-d,f]?
cp dd df
```

Table A-1 Examples of Sets

SET	EXPLANATION
[b-f]	Denotes any characters in the set b, c, d, e, f
[1-4,7-9]	Denotes any two numbers, the first of which must be 1, 2, 3, or 4 and the second of which must be 7, 8, or 9
[aeiou]	Denotes any character that is a vowel
[b,aeiou]	Denotes any character that is either b or a vowel
[aeiou][a-z]	Denotes any vowel followed by any lowercase character between and including a and z

The resulting display consists of any filename beginning with b, c, d, or f followed by any single alphanumeric character.

```
$ ls [a-z][a-z][a-z]
ash  awk  bsh  cat  csh  cut  env  ksh  pwd  red  rpm  rvi  sed  tar
zsh
```

The resulting display consists of files whose name is made up of any combination of three lowercase characters between a and z, inclusive.

Bash also has a way to reverse the meaning of the set notation. That is, the notation `[!set]` refers to all characters not in the set defined by set. So, for example, an easy way to look for all filenames not beginning with a consonant is to use `[!aeiou]*`, that is:

```
$ ls [!aeiou]*
basename  dd          gtar        mail        ping        setserial  tracepath6
bash      df          gunzip      mkdir       ping6       sfxload    traceroute
bsh       dmesg      gzip        mknod       ps          sh          traceroute6
cat       dnsdomainname  hostname    mktemp      pwd         sleep      true
chgrp     doexec     kbd_mode   more        red         sort       vi
chmod     domainname  kill       mount       rm          stty       view
...
```

Bash's wildcard and set operators are a subset of its special characters, also called metacharacters. *Metacharacters* are characters to which Bash assigns a special meaning or interpretation. Table A-2 lists all of Bash's special characters.

You should already be familiar with redirecting input and output and using pipes, but you will see examples of all three operations later in the chapter. Input and output redirection should be familiar to you. Commands in a block, that is, delimited by `{` and `}` elicit different behavior from Bash than commands executed in a subshell, that is, commands delimited by `(` and `)`. (You will read about subshells in the section "Shell Functions" later in this chapter.)

Table A-2 Bash Special Characters

CHARACTER	DESCRIPTION
'	Strong quote
"	Weak quote
#	Comment
\$	Variable expression
&	Execute command in background
(Start subshell
)	End subshell
*	Multicharacter wildcard
;	Command separator
?	Single-character wildcard
\	Interpret the next character literally
`	Command substitution
{	Start command block
	Pipe
}	End command block
~	Home directory
<	Redirect input
<<	Here-document
>	Redirect output
>>	Append output

The commands between two ``` characters cause command substitution (that is, their output or result replaces the expression itself). Consider the command `ls `which tar``. Before `ls` executes, ``which tar`` will be replaced by its result (`/bin/tar`, in this case). So, for example, the two commands:

```
$ which tar
/bin/tar
$ ls -l /bin/tar
-rwxr-xr-x  2 root  root      161380 Oct 11  2004 /bin/tar
```

produce the same result as the single command:

```
$ ls -l `which tar`
-rwxr-xr-x    2 root      root          161380 Oct 11  2004 /bin/tar
```

Command substitution is commonly used in shell scripts, and you will see it used throughout this chapter. Another commonly used metacharacter is `$`, used to obtain the value of a variable, as in the command `echo $PATH`.

A more readable syntax for command substitution is `$(...)`. This syntax works just like back quotes, with the interesting exception that `$(...)`, unlike ``...``, can be nested. So, for example, the following commands are exactly equivalent:

```
$ ls -l `which tar`
-rwxr-xr-x    2 root      root          150908 Mar  6 11:34 /bin/tar

$ ls -l $(which tar)
-rwxr-xr-x    2 root      root          150908 Mar  6 11:34 /bin/tar
```

Why are `'` and `"` called strong quote and weak quote characters, respectively? In short, because `'` is stronger than `"`. That is, `'` forces Bash to interpret all special characters literally, while `"` protects only some of Bash's special characters from interpretation as special characters. To illustrate, compare the effect of `'` and `"` on the output of the command `echo $PATH`:

```
$ echo $PATH
/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin:/home/kwall/bin
$ echo "$PATH"
/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin:/home/kwall/bin
$ echo '$PATH'
$PATH
```

The output you see might be different. Notice how using `'` around the `$PATH` variable caused it to be echoed literally, rather than producing the path listing displayed by the other two commands. You will see the implication of this difference throughout the scripts used in this chapter.

Using Variables

Like any programming language, Bash has variables, named entities that store values. Bash supports both string or character variables and numeric (integer) variables. By convention, variables are named using uppercase characters, but this is only a convention, and one not universally followed. For purposes of discussion, this section examines user-defined variables, predefined variables, and positional variables.

NOTE If you read the Bash manual page, it might seem that the discussion of variables contradicts the manual page, which treats variables as “parameters that have names.” The manual page’s approach is potentially confusing, especially for nonprogrammers. To prevent confusion, this chapter uses different terminology than the author of the manual page.

A *user-defined variable* is just that, a variable created by the shell programmer or end user, rather than one of Bash’s many intrinsic (or predefined) variables. To assign a value to a variable, use the following syntax:

```
varname=value
```

To obtain a variable’s value, you can use one of the following two formats:

```
$varname  
${varname}
```

Of the two formats, `$varname` is more commonly used, but `${varname}` is more general. You must use the second form to distinguish variable names from trailing letters, digits, or underscore characters. Consider a variable named `$STRVAR` with the value `See Rock City!` Suppose that you want to display `$STRVAR`’s value followed by an underscore character. The following command fails because Bash attempts to print the value of the variable `$STRVAR`, not `$STRVAR`.

```
$ echo $STRVAR_
```

In this situation, you must use the more general form, illustrated in the following:

```
$ echo ${STRVAR}_
```

Listing A-1 shows the code for `refvar.sh`, a script illustrating this distinction.

```
#!/bin/bash  
# refvar.sh - Referencing a variable  
  
STRVAR='See Rock City!'  
echo '${STRVAR}_ yields: ' ${STRVAR}_  
echo '$STRVAR_ yields : ' $STRVAR_
```

Listing A-1 Referencing variables.

The output from the script, shown in the following example, makes it clear that `${STRVAR}_prints nothing`, but `${STRVAR}` produces the expected output:

```
$ ./refvar.sh
${STRVAR}_ yields:  See Rock City!_
$STRVAR_ yields   :
```

Many of Bash's predefined or built-in variables are environment variables, such as `$BASH_VERSION` or `$PWD`. The complete list is available from the Bash man page. Table A-3 lists commonly used Bash predefined variables.

Listing A-2 shows the code for `predef.sh` that displays the names and values of a select subset of Bash's predefined variables.

```
#!/bin/bash
# predef.sh - Show the values of a select predefined
#             Bash variables

echo "          PPID = $PPID"
echo "BASH_VERSION = $BASH_VERSION"
echo "          RANDOM = $RANDOM"
echo "  SHELLOPTS = $SHELLOPTS"
echo "          IFS = $IFS"
echo "    HISTSIZE = $HISTSIZE"
```

Listing A-2 Referencing predefined Bash variables.

The output from this script should resemble the following:

```
$ ./predef.sh
          PPID = 9205
BASH_VERSION = 3.00.13(1) -release
          RANDOM = 29852
  SHELLOPTS = braceexpand:hashall:interactive-comments:posix
          IFS =

    HISTSIZE = 1000
```

The value of `$PPID` and `$RANDOM` will surely vary on your system.

Bash has another group of predefined variables, positional parameters. Positional parameters are variables that contain the arguments, if any, passed to a script on its command line. The parameter `$0` contains the name of the script; the actual parameters begin at `$1`. If there are more than nine such parameters, you must use the `${}` syntax, such as `${10}`, to obtain their values. Positional parameters passed to a script are read-only, that is, you cannot change their values.

Table A-3 Bash 2.X Predefined Variables

VARIABLE NAME	DESCRIPTION
\$PPID	Stores the process ID of the Bash's parent process.
\$PWD	Contains the current working directory (as set by the <code>cd</code> command).
\$OLDPWD	Stores the most previous working directory (as set by the next most recent <code>cd</code> command).
\$UID	Contains the user ID of the current user as set when the shell started.
\$BASH_VERSION	Stores the version string of the current Bash instance.
\$RANDOM	Returns a random number between 0 and 32,767 each time it is referenced.
\$OPTARG	Contains the last command line argument read using the <code>getopts</code> built-in command.
\$OPTIND	Points to the array index of the next argument to be processed using the <code>getopts</code> built-in command.
\$HOSTNAME	Stores the name of the current host system.
\$SHELLOPTS	Contains a colon-separated string of the currently enabled shell options (as set using the <code>set -o</code> shell built-in command).
\$IFS	Stores the value of the internal field separator (IFS), used to define word boundaries. The default value is a space, a tab, or a newline.
\$HOME	The home directory of the current user.
\$PS1	Defines the primary command prompt. The default value is <code>"\s-\v\\$ "</code> .
\$PS2	Defines the secondary command prompt, used to indicate that additional input is required to complete a command.
\$HISTSIZE	The number of commands retained in Bash's command history. The default value is 500.
\$HISTFILE	The name of the file in which Bash stores the command history. The default value is <code>\$HOME/.bash_history</code> .

In addition to the numerically named positional parameters, Bash recognizes three special ones, `$#`, `$@`, and `$*`. `$#` is the number of positional parameters passed to the script, not counting `$0`. `$*` contains a list of all the positional parameters passed to a script, except `$0`, formatted as a string. Each parameter is separated from the others by the first character of `$IFS`, the internal field separator. `$@`, finally, is a list of all of the positional parameters stored as separate strings delimited by weak quotes.

Using Bash Operators

In a programming context, operators are tokens that perform some sort of operation or manipulation. For example, many programming languages use the plus sign, `+`, to indicate addition and the asterisk, `*`, to indicate multiplication. Bash operators behave similarly. They fall into several categories, including comparison operators, file test operators, substitution operators, and pattern matching operators. This section looks at each class of operators and shows examples of how to use them.

Comparison Operators

Comparison operators, the simplest to use and understand, compare one variable or value to another, returning 0 if the comparison is true or 1 if the comparison is false. There are two sets of comparison operators, one for string or character data and one for numeric (integer) data. Tables A-4 and A-5 list Bash's string and numeric operators.

When comparing strings, keep in mind that A–Z come before a–z in the ASCII character set, so A is “less than” a and foo is “greater than” bar.

TIP If you are not familiar with the ASCII sort order, Fedora Core and RHEL systems include a manual page (`man ascii`) that defines the ASCII character set.

Listings A-3 and A-4, `strcmp.sh` and `numcmp.sh`, respectively, illustrate the use of the string and comparison operators.

Table A-4 Bash String Comparison Operators

OPERATOR	EXPRESSION	TRUE IF . . .
<code>=</code>	<code>str1 = str2</code>	<code>str1</code> matches <code>str2</code>
<code>==</code>	<code>str1 == str2</code>	<code>str1</code> matches <code>str2</code>
<code>!=</code>	<code>str1 != str2</code>	<code>str1</code> does not match <code>str2</code>
<code><</code>	<code>str1 < str2</code>	<code>str1</code> is less than <code>str2</code>
<code>></code>	<code>str1 > str2</code>	<code>str1</code> is greater than <code>str2</code>
<code>-n</code>	<code>-n str</code>	<code>str</code> 's length is nonzero
<code>-z</code>	<code>-z str</code>	<code>str</code> 's length is zero

Table A-5 Bash Numeric Comparison Operators

OPERATOR	EXPRESSION	TRUE IF . . .
-eq	val1 -eq val2	val1 equals val2
-ne	val1 -ne val2	val1 is not equal val2
-ge	val1 -ge val2	val1 is greater than or equal to val2
-gt	val1 -gt val2	val1 is greater than val2
-le	val1 -le val2	val1 is less than or equal to val2
-lt	val1 -lt val2	val1 is less than val2

```
#!/bin/bash
# strcmp.sh - Using Bash string comparison operators

C1="b"
C2="F"
S1="STRING"
S2="string"
S3="some darn string"

if [[ $C1 > $C2 ]]; then
    echo "$C1 is greater than $C2"
else
    echo "$C1 is less than $C2"
fi

if [[ $S1 > $S2 ]]; then
    echo "$S1 is greater than $S2"
else
    echo "$S1 is less then $S2"
fi

if [[ -n $S3 ]]; then
    echo "S3 is $S3"
else
    echo "S3 is empty"
fi

if [[ -z $S4 ]]; then
    echo "S4 is empty"
else
    echo "S4 is $S4"
fi
```

Listing A-3 Using Bash string comparison operators.

`strcmp.sh`'s output looks like this:

```
$ ./strcmp.sh
b is greater than F
STRING is less than string
S3 is some darn string
S4 is empty
```

A few notes are in order. First, do not worry about understanding the syntax of the `if-then-else` construct right now. The section titled “Flow Control” later in the chapter discusses it, so just accept that it works for the time being. Secondly, the `[` operator must be terminated with a `]`. Finally, as noted earlier, `F` precedes `b` in the ASCII character set, so `b` is greater than `F` lexicographically. Similarly, `STRING` is less than `string`. The last two tests use the `-z` and `-n` operators to test for empty and nonempty strings.

Listing A-4 presents `numcmp.sh`, a script that illustrates how to use some of Bash's numeric comparison operators.

```
#!/bin/bash
# numcmp.sh - Using Bash numeric comparison operators

W=7
X=10
Y=-5
Z=10

if [ $W -gt $X ]; then
    echo "W is greater than X ($W > $X)"
else
    echo "W is less than X ($W < $X)"
fi

if [ $X -lt $Y ]; then
    echo "X is less than Y ($X < $Y)"
else
    echo "X is greater than Y ($X > $Y)"
fi

if [ $X -eq $Z ]; then
    echo "X equals Z ($X = $Z)"
fi

if [ $Y -le $Z ]; then
    echo "Y is less than or equal to Z ($Y <= $Z)"
else
    echo "Y is greater than or equal to Z ($Y >= $Z)"
fi
```

Listing A-4 Using Bash numeric comparison operators.

numcmp.sh's output should be

```
$ ./numcmp.sh
W is less than X (7 < 10)
X is greater than Y (10 > -5)
X equals Z (10 = 10)
Y is less than or equal to Z (-5 <= 10)
```

The next section extends your ability to work with numeric values in shell scripts, showing you how to use arithmetic operators to perform basic arithmetic operations in Bash shell scripts.

Arithmetic Operators

If Bash allows you to compare numeric values and variables, it follows that Bash allows you to perform arithmetic on numeric values and variables. In addition to the comparison operators discussed in the previous section, Bash uses the operators listed in Table A-6 to perform arithmetic operations in shell scripts.

Table A-6 Bash Arithmetic Operators

OPERATOR	EXPRESSION	DESCRIPTION
++	var++	Increments <code>var</code> after obtaining its value
++	++var	Increments <code>var</code> before obtaining its value
--	var--	Decrements <code>var</code> after obtaining its value
--	--var	Decrements <code>var</code> before obtaining its value
+	var1 + var2	Adds <code>var1</code> and <code>var2</code>
-	var1 - var2	Subtracts <code>var2</code> from <code>var1</code>
*	var1 * var2	Multiplies <code>var1</code> by <code>var2</code>
/	var1 / var2	Divides <code>var1</code> by <code>var2</code>
%	var1 % var2	Calculates the remainder of dividing <code>var1</code> by <code>var2</code>
=	var = op	Assigns result of an arithmetic operation <code>op</code> to <code>var</code>

The complete list of Bash's arithmetic operators also includes bitwise and logical functions, a conditional evaluation operator, C-style assignment operators, and a comma operator. Refer to Bash's excellent manual page for complete details.

File Test Operators

Many common administrative tasks involve working with files and directories. Bash makes it easy to perform these tasks because it has a rich set of operators that perform a variety of tests. For example, you can, determine whether a file exists at all or if it exists but is empty, if a file is a directory, and what the file's permissions are. Table A-7 lists commonly used file test operators.

Table A-7 Bash File Test Operators

OPERATOR	DESCRIPTION
<code>-d file</code>	<code>file</code> exists and is a directory.
<code>-e file</code>	<code>file</code> exists.
<code>-f file</code>	<code>file</code> exists and is a regular file (not a directory or special file).
<code>-g file</code>	<code>file</code> exists and is set-GID (set group ID).
<code>-r file</code>	You have read permission on <code>file</code> .
<code>-s file</code>	<code>file</code> exists and is not empty.
<code>-u file</code>	<code>file</code> exists and is set-UID (set user ID).
<code>-w file</code>	You have write permission on <code>file</code> .
<code>-x file</code>	You have execute permission on <code>file</code> or, if <code>file</code> is a directory, you have search permission on it.
<code>-O file</code>	You own <code>file</code> .
<code>-G file</code>	<code>file</code> 's group ownership matches one of your group memberships.
<code>file1 -nt file2</code>	<code>file1</code> is newer than <code>file2</code> .
<code>file1 -ot file2</code>	<code>file1</code> is older than <code>file2</code> .

The script in Listing A-5 illustrates how to use the file test operators. It applies several file test operators to each file in the top level of a directory name provided as an argument.

```
#!/bin/bash
# filetest.sh - Illustrate file test operators
# Arguments: Name of directory to scan

# Have to provide the argument
if [[ $# != 1 ]]; then
    echo "usage: filetest.sh DIR"
    exit 1
fi

# Only accept directory arguments
if [ ! -d $1 ]; then
    echo "$1 is not a directory"
    exit 1
fi

# Process each file in the directory
for FILE in $1/*
do
    # Ignore directories and special files
    if [ -f $FILE ]; then
        echo $FILE
        if [ -r $FILE ]; then
            echo -e "\tReadable by $USER"
        fi
        if [ -w $FILE ]; then
            echo -e "\tWritable by $USER"
        fi
        if [ -x $FILE ]; then
            echo -e "\tExecutable by $USER"
        fi
        if [ -O $FILE ]; then
            echo -e "\tOwned by $USER"
        fi
    fi
done
```

Listing A-5 File test operators.

The first `if` statement ensures that the script was called with one argument, using the `$#` positional parameter discussed earlier in the chapter. If not, it displays a usage message and exits; otherwise, execution continues with the next `if` statement. The second `if` statement makes sure that the argument passed to `filetest.sh` (the `$1` parameter) is a directory name. If it is not, the script exits after displaying an error message. Otherwise, processing continues with the `for` loop.

The `for` loop processes each file in the directory. For each file, the script first tests whether it is a regular file using the `-f` operator. If it is not, it skips to the

next file. Otherwise, `filetest.sh` tests whether the user executing the script has read, write, and execute permission on the file (using the `-r`, `-w`, and `-x` operators, respectively), whether the user owns the file (using the `-O` operator), displays the results of each test on `stdout`, and then repeats these tests for the next file. After evaluating the last file, the script exits.

The output of this script will vary, of course, but it should resemble the following:

```
$ ./filetest.sh $HOME
./599946_chxa.txt
    Readable by kwall
    Writable by kwall
    Owned by kwall
./filetest.sh
    Readable by kwall
    Writable by kwall
    /Executable by kwall
    Owned by kwall
./numcmp.sh
    Readable by kwall
    Writable by kwall
    /Executable by kwall
    Owned by kwall
./predef.sh
    Readable by kwall
    Writable by kwall
    /Executable by kwall
    Owned by kwall
./refvar.sh
    Readable by kwall
    Writable by kwall
    /Executable by kwall
    Owned by kwall
./strcmp.sh
    Readable by kwall
    Writable by kwall
    /Executable by kwall
    Owned by kwall
```

Understanding Flow Control

Any programming language worth the name must support flow control — the ability to control the order in which code executes. Flow control allows code to loop, branch, or execute conditionally. To *loop* means to execute the same block of code multiple times. To *branch* means to execute a block of code specific to the value of a condition. *Conditional execution* means to execute or not execute

code, depending on the value of some variable. Bash supports flow control, and this section explains the syntax and use of all its flow control structures, listed in Table A-8.

Table A-8 Bash Flow Control Statements

STATEMENT	TYPE	DESCRIPTION
<code>if</code>	Conditional	Executes code if a condition is true
<code>for</code>	Looping	Executes code a fixed number of times
<code>while</code>	Looping	Executes code as long as a condition is true
<code>until</code>	Looping	Executes code as long as a condition is false
<code>case</code>	Branching	Executes code specific to the value of a variable
<code>select</code>	Branching	Executes code based on a selection made by a user

Conditional Execution Using `if` Statements

The syntax of Bash’s `if` statement is shown in the following:

```
if condition then
    statements
[elif condition then
    statements]
[else
    statements]
fi
```

`condition` is the control variable that determines which statements execute. `elif` is Bash’s equivalent to `else if` in other programming languages. `fi` signals the end of the `if` structure. Be careful when creating `condition` because `if` checks the exit status of the last statement in `condition`. That is, if `condition` consists of multiple tests, only the last one (unless grouping using `()` is applied) affects the control flow. Most Linux programs return 0 if they complete successfully or normally and nonzero if an error occurs, so this peculiarity of Bash’s `if` statement usually does not pose a problem.

Bash’s `if` statement works just like the `if` structures in most other programming languages. If the exit status is 0 (true), then statements will be executed. If it is nonzero (false), the statements following an `elif` or `else` clause, if present, will be executed, after which control jumps to the first line of code following `fi`. The `else` clause executes only if all other `conditions` in `elif` clauses are false. The `else` and `elif` clauses are optional, and you can use multiple `elif` clauses if you wish.

CAUTION While most Linux programs return 0 if they complete successfully and nonzero otherwise, this is not universally true. The `diff` program, for example, returns 0 if it does not find differences, 1 if it does, or 2 if a problem occurred. Nonstandard exit values such as these can alter the way a shell script works, so double-check return codes in a program's or command's documentation.

Evaluating exit codes, and the order in which you evaluate them, is important because Bash allows you to use the `&&` and `||` operators to combine exit codes in condition. `&&` is known as a *logical AND*, which means that a condition is true if and only if the expressions on both sides of the `&&` operator are true. So, given the condition `expression1 && expression2`, the condition is true if and only if `expression1` and `expression2` are true. If `expression1` is false (nonzero), Bash will not evaluate `expression2`. The `||`, called a *logical OR*, requires only that either one or the other of the expressions on either side of the operator be true. Thus, the condition `expression1 || expression2` is true if either `expression1` or `expression2` is true. The condition follows the general form `expression1 || expression2`.

How do these operators work? Suppose that you have to change the current working directory and copy a file before executing a certain code block. One way to accomplish this is to use nested `if` statements, as in the following snippet:

```
if cd $DATADIR then
    if cp datafile datafile.bak then
        #
        # Code here
        #
    fi
fi
```

Using the `&&` operator, you can write this much more concisely, as the following code snippet illustrates:

```
if cd $DATADIR && cp datafile datafile.bak then
    #
    # Code here
    #
fi
```

Both code snippets accomplish the same thing, but the second example is shorter and easier to read and understand because you avoid getting bogged down sorting out the nested `ifs`. If, for some reason, the `cd` command fails, Bash will not attempt the copy operation.

Several shell scripts in this chapter have used the `if` structure, so refer to the previous examples to see various ways to use `if`.

To make sense of conditional execution, you will find it helpful to understand how Bash uses the `test` and `[` commands and the `[[` compound command. Yes, `[` and `[[` are commands, not operators. `test` and `[`, which are equivalent commands, evaluate conditional expressions and return true or false (zero or nonzero, respectively), depending on how the conditional expression evaluates.

Determinate Loops Using the `for` Statement

The `for` structure allows you to execute a section of code a fixed number of times, so the loops it creates are called determinate. The `for` loop is a frequently used loop tool because it operates neatly on lists of data, such as Bash's positional parameters (`$1–$9`) or lists of files in a directory. The general format of the `for` statement is:

```
for value in list
do
    command
[... ]
done
```

`list` is a white-space-separated list of values, such as filenames. `value` is a single list item; for each iteration through the loop, Bash sets `value` to the next element in `list`. `command` can be any valid Bash expression, and, naturally, you can execute multiple commands. `command` typically uses or manipulates `value`, but it does not have to.

TIP The `for` statement is a Bash built-in command, so you can type `help for` at the command prompt to get a quick syntax summary.

Suppose that you want to copy a number of files in the same directory, adding the extension `.bak` to the copies. DOS users accustomed to the way DOS copies multiple files are dismayed when they discover that the same approach does not work in Linux. To illustrate, the following DOS command uses the `COPY` command to copy every file with a `.dat` extension, replacing `.dat` with `.bak`:

```
C:\> copy *.dat *.bak
data1.dat
data2.dat
data3.dat
        3 file(s) copied
```

The same method does not work for Linux's `cp` command because if you specify multiple files to copy, `cp` expects the final argument to be a directory. One way around this inconvenience is to use Bash's `for` loop. The following example illustrates using `for` at the command line to emulate the DOS `COPY` command's behavior:

```

$ for FILE in *.dat
> do
>     cp -v $FILE ${FILE%.dat}.bak
> done
`data1.dat' -> `data1.bak'
`data2.dat' -> `data2.bak'
`data3.dat' -> `data3.bak'

```

The `cp` command's `-v` (verbose) option causes it to display what it is doing. Note the use of the `%` pattern matching operator to replace `.dat` with `.bak` for each file copied. Notice also how the Bash command prompt changes to the secondary prompt (PS2), `>`, indicating that Bash needs more input to complete the command. The previous listing showed a slightly more complex usage of the `for` loop.

Indeterminate Loops Using `while` and `until` Statements

The `for` loop is ideal when you know in advance, or can determine before entering the loop, how many times a given section of code must execute. If you do not know in advance or cannot determine at runtime how many times to execute a code block, you need to use one of Bash's indeterminate loop structures, `while` and `until`. `while` and `until` constructs cause continued code execution as long as or until a particular condition is met. The key here is that your code must ensure that the condition in question is eventually met, or you will be stuck in an infinite loop.

The general form of a `while` loop is:

```

while condition
do
    command
    [...]
done

```

In a `while` loop, as long as `condition` remains true, execute each command until `condition` becomes false. If `condition` is initially false, execution jumps to the first statement following `done`.

The general form of the `until` loop is:

```

until condition
do
    command
    [...]
done

```

`until`'s structure and meaning is the opposite of `while`'s because an `until` loop executes as long as `condition` is false; a `while` loop executes as long as `condition` is true.

Listing A-6 illustrates the use of Bash's `while` loop. It makes 150 copies of a file named `junk` in the current directory.

```
#!/bin/bash
# while.sh - Illustrate using Bash while loops

declare -i CNT
CNT=1

if [ ! -r junk ]
then
    touch ./junk
fi

while [ $CNT -le 150 ]
do
    cp junk junk.$CNT
    CNT=$((CNT+1))
done
```

Listing A-6 Bash while loops.

`while.sh` illustrates several Bash programming features. Of course, you see the `while` loop in action. It also uses the `-e` file test operator to test for the existence of the file named `junk` and, if it does not exist, uses the `touch` command to create it. The `declare` statement creates an integer variable, `CNT`, to serve as the loop control variable. Each iteration of the loop increments `CNT`, making sure that the loop condition eventually becomes false. Do not forget to delete the files the script creates!

The following code snippet shows how to use an `until` loop to accomplish the same thing as the `while` loop in Listing A-6.

```
until [ $CNT -gt 150 ]
do
    cp junk junk.$CNT
    CNT=$((CNT+1))
done
```

The logic of the condition is different, but the end result is the same. In this case, using a `while` loop is the appropriate way to handle the problem.

Selection Structures Using `case` and `select` Statements

This section discusses the `case` and `select` selection structures, which allow a shell script to execute a specific section of code depending on the value of a

variable or expression or in response to user input. You will learn about the `case` statement first and then about the `select` statement.

The case Statement

The `case` structure, approximately comparable to C's `switch` keyword, is best suited for situations in which a variable or expression can have numerous values and as a replacement for long `if` blocks. The complete syntax for `case` is as follows:

```
case expr in
    pattern )
        commands ;;
    pattern )
        commands ;;
    [...]
esac
```

The space between `pattern` and `)` is required, as is the space between `commands` and `;;`. `expr` can be any valid Bash expression. `expr` is compared to each `pattern` until a match is found, at which point, the commands associated with the first match are executed. `;;`, equivalent to C's `break`, signals the end of the code block and causes control to jump to the first line of code past `esac` (case in reverse), which marks the end of the `case` structure. Why use two semicolons? The first semicolon terminates the last command and the second semicolon terminates the enclosing portion of the case block. If no `pattern` matches, execution resumes at the first line of code after `esac`. Listing A-7 shows the `case` statement at work.

```
#!/bin/bash
# case.sh - Using the case selection structure

clear
echo -n "Type a single letter, number, or punctuation character: "
read -n 1 OPT
echo

case $OPT in
    [:upper:] ) echo "$OPT is an upper case letter" ;;
    [:lower:] ) echo "$OPT is a lower case letter" ;;
    [:digit:] ) echo "$OPT is a digit" ;;
    [:punct:] ) echo "$OPT is punctuation" ;;
    *         ) echo "What is *that*?" ;;
Esac
```

Listing A-7 Using the case statement.

After clearing the screen with the `clear` command, the script prompts for a single character of input and uses the `read` built-in to read a single character from stdin and store it in `$OPT`. The `case` statement uses a special notation, explained shortly, to represent sets of characters. If the character typed at the prompt matches one of the four patterns (`[:upper:]`, `[:lower:]`, `[:digit:]`, or `[:punct:]`), `case.sh` displays the appropriate message. If no match is found, the script silently exits.

In addition to illustrating the case selection structure, `case.sh` introduces two new Bash programming elements, `read` and character classes. You will learn about the `read` built-in command in the section “Processing Input and Output.” later in the chapter. A character class is a symbolic name embedded between `[:` and `:]` that represents a group of characters. In fact, character classes are special cases of the `[]` set notation described in the section titled “Using Wildcards and Special Characters” near the beginning of the chapter. For example, the set `[a-z]` is equivalent to the character class `[:lower:]`. The most common character classes are listed in Table A-9.

Table A-9 Common Bash Character Classes

CLASS	DESCRIPTION
<code>[:alnum:]</code>	All alphabetic and numeric characters
<code>[:alpha:]</code>	All alphabetic characters
<code>[:digit:]</code>	All numerals
<code>[:lower:]</code>	All lowercase letters
<code>[:punct:]</code>	All punctuation symbols
<code>[:upper:]</code>	All uppercase letters

The select Statement

The `select` control structure makes it easy to create menus. Indeed, it seems almost to have been designed specifically for that purpose. Its syntax is

```
select value [in list]
do
    commands
done
```

The `select` statement creates a numbered menu for each value in `list`. Selecting the number corresponding to a value causes `commands` to execute. Listing A-8, `dircnt.sh`, illustrates how `select` works. It counts the number of files in a user-specified directory, using the `select` statement to identify the directory to analyze.

```
#!/bin/bash
# dircnt.sh - Using the select statement

PS3="[Number]: "
IFS=:

clear

select ITEM in $PATH Quit
do
    [ "$ITEM" = "Quit" ] && exit 0
    if [ "$ITEM" ]; then
        CNT=$(ls -Al $ITEM | wc -l)
        echo "$CNT files in $ITEM"
    else
        echo "Invalid selection"
    fi
    echo
done
```

Listing A-8 Creating menus with select.

The script first sets the `$PS3` environment variable, which is the prompt that `select` displays to request user input. Next, it sets the `$IFS` (internal field separator) variable to `:` to enable `select` properly to parse the `PATH` environment variable. After clearing the screen, it enters a loop, presenting a numbered menu of options derived from the `$PATH` variable and the literal string `Quit`. Then it prompts the user to make a choice, as shown in the following (the exact appearance depends on the value of the `PATH` variable):

```
1) /usr/kerberos/bin  4) /bin  7) /sbin
2) /usr/local/bin    5) /usr/X11R6/bin  8) /usr/sbin
3) /usr/bin          6) /home/kwall/bin  9) Quit

[Number]: 1
22 files in /usr/kerberos/bin

[Number]: 2
4 files in /usr/local/bin

[Number]: 3
3627 files in /usr/bin

[Number]: 9
```

If the user selects the numeral corresponding to the `Quit` option (9 in this example), `dircnt.sh` exits. Selecting a valid choice (1–8 in this case) stores

the corresponding directory name in `$ITEM`, which is then used in the statement `CNT=$(ls -Al $ITEM | wc -l)`. This command counts the number of files (not including `.` and `..`) in the directory specified by `$ITEM`. The file count is displayed to `stdout`, and the menu redrawn. If the user makes an invalid selection, `dircnt.sh` displays an error message and then redraws the menu.

Bash has a fully capable range of flow control structures. Although their syntax is different from traditional programming languages such as C, C++ they work nearly the same and enable you to create powerful, flexible shell scripts.

Using Shell Functions

Shell functions offer two important advantages over aliases. First, shell functions accept command line arguments that can be processed within the body of the function. Aliases do not accept arguments in this way. Second, shell functions increase the modularity of your scripts. If you store frequently used shell functions in a separate file, you can use the `source` built-in to read that file into your scripts. Then, if you change a function, you have to change only one file rather than edit each file that uses the script.

TIP Aliases, do not work in shell scripts unless the shell option

`expand_aliases` is set (`shopt expand_aliases`).

Define shell functions using one of the following two forms:

```
function fname
{
    commands
}
```

```
fname()
{
    commands
}
```

`fname` is the name of the function and `commands` are the statements and expressions that define its behavior. Both forms are widely used, and there is no difference between them. You must define functions before they are used, such as at the beginning of a script or in another file that is sourced before the functions are called. The following example defines a function (at the command prompt) named `mydate` that displays the output of the `date` command in a custom format.

```
$ function mydate
> {
>   date '+%A, %B %d, %Y'
> }
$
```

We defined this function at the command prompt for convenience's sake. You could just as well define it in a file and use the `.` or `source` commands to read the definition into your environment.

To use a function, simply invoke it using its name followed by any arguments it needs. Once a function has been defined, you can use it on the command line or in a script. For example, to use the `mydate` shell function just defined on the command line, type its name at the Bash prompt:

```
$ mydate
Sunday, April 24, 2005
```

Alternatively, place the definition in a script and then execute the script. The Red Hat startup scripts use precisely this strategy. The file `/etc/rc.d/init.d/functions` defines at least 15 shell functions used throughout the initialization scripts. Near the top of every init script in `/etc/rc.d/init.d` you will find these two lines:

```
# Source function library.
. /etc/rc.d/init.d/functions
```

The key command is `. /etc/rc.d/init.d/functions`, which sources the contents of the file and inserting the contents into the init script at that point, which enables the init script to use the functions defined in the file. In fact, studying the contents of `/etc/rc.d/init.d/functions` gives you great insight into using Bash functions and into the larger subject of shell programming in general.

Processing Input and Output

Bash sports a surprisingly rich collection of input and output features. Many of them, such as manipulating file descriptors or reading and writing to device files, are esoteric and most useful to low-level system programmers and will not be covered in this section. Rather, it covers I/O redirection, using the `echo` command for output, and using the `read` command for input.

Redirecting I/O

The basic I/O redirectors, `>` and `<`, redirect output from `stdout` and input from `stdin`, respectively. Most often, `>` is used to save the output from a command to

a file. For example, the command `cat /etc/passwd > out` captures the output of the `cat` command and sends it to a file named `out` in the current directory. To capture `stderr` output to a file, put a `2` immediately in front of `>`. For example, if you invoke `tar` with no options or arguments, it displays an error message to `stderr` (usually the display). The following command sends the error message to `/dev/null` (the bit bucket):

```
$ tar 2> /dev/null
```

Of course, you can also redirect `stdout` to `/dev/null`. In fact, it is common practice in shell scripts to redirect both `stdout` and `stderr` to `/dev/null`. Why? In many cases, a command's output is less important than whether it succeeded or failed. Because most programs return an exit code indicating success (usually `0`) or failure (usually any nonzero value), from a programmatic standpoint it is sufficient to use the predefined Bash variable `?` to check a program's exit status. In other cases, hiding output gives a script a more polished, professional appearance and lets you control how your script appears to users.

To redirect `stdout` and `stderr` to the same file, such as `/dev/null`, use a command of the following form:

```
mycommand > /dev/null 2>&1
```

Replace `mycommand` with the command and any arguments the command requires. The notation `2>&1` redirects `stderr` to `stdout`, which was previously redirected to `/dev/null`. A less familiar but functionally identical form and more concise version of this command is:

```
mycommand &> /dev/null
```

The section titled “Redirecting Standard Output and Standard Error” in the Bash man page explains how and why this syntax works.

If you use `>` to redirect `stdout` to an existing file, you will lose the file's original contents. To avoid this circumstance, use Bash's append operator, `>>`, which adds data to the end of a file, thus protecting the file's original contents.

To redirect `stdin`, which is ordinarily the keyboard, use the input redirector, `<`. For example, the command `cat < /etc/passwd > out` redirects both input and output.

So-called here-documents, which use the `<<` operator, are an interesting case of input redirection. The term here-document derives from the fact that Bash reads its input from the current source, “here,” if you will, rather than from `stdin` or a separate file. A here-document stores the input to a command in the script. The syntax, which looks somewhat odd, is:

```

command << label
    commandinput
label

```

label demarcates the beginning and end of the command's input. commandinput is the input itself and command is the program or utility or command to execute. This syntax says that command should read commandinput until it encounters label on a line by itself.

A typical use of here-documents is for scripting programs that lack a convenient scripting interface. The familiar FTP client program is a fine example of this. Listing A-9, ftp.sh, illustrates how to script the FTP client using a here-document.

```

#!/bin/bash
# ftp.sh - Scripting ftp using a here-document

USER=anonymous
PASS=kwall@kurtwerks.com

ftp -i -n << END
open ftp.kurtwerks.com
user $USER $PASS
cd pub
ls
close
bye
END
echo "done"

```

Listing A-9 Scripting ftp using a here-document.

ftp -i -n uses -i to start in noninteractive mode and -n to suppress the automatic login. The script uses END as the label to signal the end of input. Using various ftp commands, the script first opens an FTP session to localhost. Then it sends the login sequence using the \$USER and \$PASS variables defined at the beginning of the script. Once logged in, it changes to the standard public FTP directory, pub, and then executes an ls command to demonstrate that the script worked. Finally, it closes the connection and exits the program. When Bash encounters the second END label, the script exits after printing done on the terminal.

The output shown in the following illustrates ftp.sh in action:

```

$ ./ftp.sh
-rw-r--r--  1 ftp      ftp      561 Nov 26 18:48 amd8111-makefile.patch
-rw-r--r--  1 ftp      ftp      61976 Aug 31  2003 axis_ev1.avi.gz

```

```

drwxr-xr-x  2 ftp      ftp      4096 Feb  7  2004 books
drwxr-xr-x  8 ftp      ftp      4096 Mar 19 18:48 dict
drwxr-xr-x  2 ftp      ftp      4096 Nov 26 18:31 docbook
-rw-r--r--  1 ftp      ftp      5473754 Apr 29  2003 mailman-2.1.1.tgz
-rw-r--r--  1 ftp      ftp        389 Sep  7  2004 md5sums
-rw-r--r--  1 ftp      ftp     1646447 Mar  9  2003 parted-bin-1.6.5.tar.gz
-rw-r--r--  1 ftp      ftp     1969624 Jun 13  2004 postfix-2.1.1.tar.gz
drwxr-x---  2 root     root      4096 May 10  2003 private
drwxr-xr-x  3 ftp      ftp      4096 Jun 27  2004 slackages
-rw-r--r--  1 ftp      ftp      1576 Dec 30  2003 toshutils-2.0.1-patches.t
ar.gz
-rw-r--r--  1 ftp      ftp     154419 Dec 30  2003 toshutils-2.0.1.tar.gz
-rw-r--r--  1 ftp      ftp        682 Jun 15  2003 webalizer.h.patch
done

```

Another typical use for here-documents is to include documentation, such as online help, in scripts.

String I/O

Many of the scripts that you have seen in this chapter use the `echo` command to display information. It supports a number of useful options. You have already seen `-e`, which enables the interpretation of escaped characters, such as `\n` (newline) and `\t` (tab). `echo`'s `-n` option omits appending the final newline to its output. Table A-10 lists escape sequences that `echo` interprets when called with the `-e` option.

Table A-10 `echo` Escape Sequences

SEQUENCE	DESCRIPTION
<code>\a</code>	Alert
<code>\b</code>	Backspace
<code>\c</code>	Omits the final newline appended to output
<code>\f</code>	Formfeed
<code>\r</code>	Return
<code>\v</code>	Vertical tab
<code>\n</code>	Newline
<code>\\</code>	A single <code>\</code>

To work with string input, use the `read` command. Its syntax is:

```
read var1 var2 ...
```

Although ideal for getting input in an interactive mode from a user, `read` also works with files. In interactive mode, `read` is simple to use, as the following code fragment illustrates:

```
echo -n 'Name: '
read NAME
echo "Name is $NAME"
```

Using `read` to process text files is somewhat more complicated. The easiest way is to create a script and then redirect its input from the file you want to process. Listing A-10 processes the contents of `/etc/passwd`.

```
#!/bin/bash
# read.sh - Using read to process a text file

IFS=:
while read name pass uid gid gecos home shell
do
    echo "*****"
    echo "name   : $name"
    echo "pass   : $pass"
    echo "uid    : $uid"
    echo "gid    : $gid"
    echo "gecos  : $gecos"
    echo "home   : $home"
    echo "shell  : $shell"
done
```

Listing A-10 Using `read` to process a text file.

Setting `$IFS` enables the script to parse the `/etc/passwd` entries properly. The `while read` statement reads each line of input, assigns the value of each field to the appropriate variable, and then uses the `echo` command to display the password information on `stdout`. You can execute this script with (at least) one of the following two commands:

```
$ ./read.sh < /etc/passwd
```

or

```
$ cat /etc/passwd | ./read.sh
```

The output should be identical regardless of which command you use. Table A-11 lists useful options the `read` command accepts.

Table A-11 Useful read Options

OPTION	DESCRIPTION
<code>-n num</code>	Reads <code>num</code> characters from <code>stdin</code>
<code>-p prompt</code>	Displays <code>prompt</code> followed by a space to solicit user input
<code>-s</code>	Disables echoing input read from <code>stdin</code>
<code>-t secs</code>	Cancels the read operation after <code>secs</code> seconds elapse without reading an input line

Using `read`'s `-p` option allows user input code to be more concise. For example, the statement `read -p "User Name: " NAME` is identical to the following fragment:

```
echo -n "User Name: "  
read NAME
```

In Listing A-7, `case.sh` uses `read -n 1` to read a single character from standard input. The `-s` (presumably, a mnemonic for silent) is ideal for reading a user password or other sensitive information. The following commands define a shell function, `getpass`, that uses all four of the options listed in Table A-7 to read a user's password, and then invokes the function.

```
$ getpass()  
> {  
> read -s -p "Password: " -n 8 -t 5 PWORD  
> echo -e "\nYou entered: $PWORD"  
> }  
$ getpass  
Password:  
You entered: sekrit
```

`-t 5` causes the `read` command to time out if the user does not press Enter in five seconds (the `echo` command still executes, though). `-n 8` causes `read` to read no more than eight characters from `stdin`. As you can see in the output, `-s` suppressed displaying what the user typed.

Working with Command Line Arguments

Most nontrivial shell scripts need to handle arguments passed to them on the command line. To maintain consistency with Linux's command line environment, options should be preceded with a `-` (for example, `foo -v`). Bash's `getopts` built-in command makes meeting both of these requirements simple. `getopts`' basic syntax is quite simple:

```
getopts optionstring varname
```

`optionstring` consists of a string of letters and colons. Letters define the valid option characters; if an option requires an argument, it must be followed by a colon (:). `varname` stores the option character being evaluated. `getopts` processes each character in `optionstring`, picking options off one at a time and assigning them to `varname`. An option character followed by a colon (:) indicates that the option requires an argument, which `getopts` assigns to `OPTARG`, one of Bash's predefined variables. As long as options remain to be processed, `getopts` returns 0, but after it has processed all options and arguments, it returns 1, making `getopts` suitable for use in a `while` loop.

Listing A-11, `getopts.sh`, illustrates the use of `getopts` and simplifies the subsequent explanation of how it works.

```
#!/bin/bash
# getopts.sh - Using Bash's getopts command to
#               process command line arguments

while getopts "xy:z:" OPT;
do
    case $OPT in
        x ) XOPT='You used -x' ;;
        y ) YOPT="You used -y with $OPTARG" ;;
        z ) ZOPT="You used -z with $OPTARG" ;;
        ? ) echo 'USAGE: getopts.sh [-x] [-y arg] [-z arg]'
            exit 1 ;;
    esac
done

echo ${XOPT:-'did not use -x'}
echo ${YOPT:-'did not use -y'}
echo ${ZOPT:-'did not use -z'}

exit 0
```

Listing A-11 Using Bash's `getopts` command.

As you can see in Listing A-11, the valid options are `x`, `y`, and `z`. Because `y` and `z` are each followed by a :, they require arguments. The `case` statement handles each of the valid options and also includes a default case, indicated by `?`, that prints a usage message and exits the script if it is invoked with invalid options (options not listed in `optionstring`). The arguments passed with the `y` and `z` options are stored in `$OPTARG`. The `echo` statements at the end of the script report the option used and, if applicable, the value of its corresponding argument.

This ends your whirlwind survey of the fundamentals of Bash programming. The next section presents sample scripts for monitoring and managing processes and creating backups, using many of the concepts and features discussed in this section.

Using Processes and Job Control

This section presents a few Bash shell scripts that automate process monitoring and management. They use both Bash built-in commands and standard Linux commands and utilities available on any Fedora Core or RHEL system. As is often the case in Linux, there are many ways to accomplish most administrative tasks, so consider the scripts in this section starting points rather than the only valid ways to perform a certain task.

Listing A-12, `kujob.sh`, kills all processes owned by the user whose login name is passed as an argument to the script.

```
#!/bin/bash
# kujob.sh - Kill all processes owned by a user
# Arguments - User's login name

# Display a short usage message
function usage
{
    ERR=$1
    case $ERR in
        "root" ) echo "Only root can run this script" ;;
        "syntax" ) echo "USAGE: kujob.sh -u USERNAME" ;;
        "arg" ) echo "-u requires user name" ;;
        "user" ) echo "No such user" ;;
        "*" ) echo "How did you get here?" ;;
    esac
    exit 1
}

# Must be root
[ $UID -eq 0 ] || usage root

# Were we called properly?
getopts ":u:" OPT || usage syntax
[ "$OPT" = ":" ] && usage arg

# Does the user exist?
USERNAME=$OPTARG
id $USERNAME &> /dev/null || usage user
```

Listing A-12 Killing a selected user's jobs.

```
# Kill the user's processes
for SIG in TERM INT HUP QUIT KILL
do
    echo "Trying SIG$SIG"
    pkill -$SIG -u $UNAME &> /dev/null
    sleep 1
    pgrep -u $UNAME &> /dev/null || exit 0
done
```

Listing A-12 *(continued)*

The usage function is a crude but effective error handler. It always terminates the script after displaying the error message associated with the argument it is passed. It is convenient because it centralizes all of the script's error handling; it is crude because it makes no attempt to recover from errors.

The first block of code uses Bash's built-in \$UID variable to see if root, whose UID is 0, is invoking the script. If the test returns false, meaning the user running the script is not root, kujob.sh calls usage with an argument of root. Otherwise, execution continues.

The second block of code uses getopts to process its command line arguments. The only valid option, as you can see in the option string, is -u. The first : (in front of u) turns off Bash's default error messages, allowing the script to provide its own error messages. The trailing :, as you recall, means that -u requires an argument. If an invalid option was used, kujob.sh calls usage with an argument of syntax. The following line makes sure that the required login name argument was provided. getopts stores a : in the \$OPT variable if a required argument was omitted, so if the test returns true, it calls usage with the arg argument.

Once kujob.sh determines it was invoked properly, the next code block performs a sanity check: making sure the named user actually exists. The rest of the script relies on this information. After copying the username passed to the UNAME variable, the grep command searches for that name in the /etc/passwd file. If it is present, grep returns 0 (true) and execution continues with the for statement. If the name is not present, grep returns 1 (false) and kujob.sh calls usage with an argument of user. Notice how both stderr and stdout from the grep command are redirected to /dev/null, resulting in a more polished runtime appearance for the script.

Finally, kujob.sh attempts to kill all of the user's processes. The signals it uses are listed in order of increasing severity and prejudice (see the signal(7) man page for more information). If one signal fails to kill a process, the next strongest one is tried until all processes associated with the user have been killed. The sleep 1 statement gives the kernel time to complete the housekeeping associated with process termination before the script continues. After all the

user's processes have been terminated, the script exits (`pgrep` returns 1, or false, if it finds no processes that match the specified search criteria).

CAUTION In addition to terminating the user's processes, `kujob.sh` kills the user's login shell, too. Be careful when using this script.

The `kill` and `pgrep` commands are part of the `procp`s package, a suite of process management and monitoring tools. `kill` and `pgrep`, in particular, provide a simpler interface to the `ps` command and its bewildering array of command line options. In short, the `kill` statement sends the signal specified by `SIG` to all processes owned by the user in `$UNAME`, specified by the `-u $UNAME` construct. `pgrep`, using the same `-u` option, lists the process IDs (PIDs) of all processes that match (you do not see them, however, because `pgrep`'s `stdout` and `stderr` are redirected to `/dev/null`). See the `kill` and `pgrep` man pages for further information.

The script in Listing A-13, `dskusg.sh`, prints a report of disk usage for each mounted file system, sorting the output from the largest to the smallest directory.

```
#!/bin/bash
# dskusg.sh - Print disk usage on each file system for top level
#             directories in descending order

function usage
{
    echo "You must be root to get an accurate report"
    exit 1
}

[ $UID -eq 0 ] || usage
for FS in $(grep ^\ /etc/mtab | cut -f2 -d" ")
do
    echo -e "$FS\n-----"
    du -x --max-depth=1 --exclude=/dev* $FS | sort -rg
    echo
done
```

Listing A-13 Displaying disk usage sorted by top-level directory size.

The first statement makes sure the user executing the script is the root user, printing an error message and exiting if the user is not root or root equivalent. Normal users do not have access to many directory trees on a Red Hat Linux system (such as `/root` and parts of `/var/log`, so this statement makes sure the report is accurate and, in the process, eliminates the necessity to redirect

stderr to /dev/null. The script selects mounted file systems as listed in /etc/mtab, which contains the list of currently mounted file systems. It uses /etc/mtab rather than mount's output or the contents of /proc/mounts because /etc/mtab is easier to parse. For each file system, the script prints the file system name, followed by the disk usage (in blocks) of the file system itself, followed by the top-level directories on each file system in descending order of disk usage, accomplished using the --maxdepth=1 argument. Using the -x option limits du to analyzing the given file system. This prevents the script from analyzing /proc, for example. --exclude=/dev* keeps the /dev special files out of the analysis. By piping the output to sort, the script can sort its output, using -g to sort based on numeric values and -r to reverse the sorting order. The output from this script might resemble the following:

```
# ./diskusg.sh
/
-----
9253464 /
7258588 /usr
1056880 /root
463760 /var
356360 /lib
84420 /etc
19364 /sbin
8128 /opt
5684 /bin
148 /tmp
80 /tftpboot
8 /boot
4 /srv
4 /space
4 /selinux
4 /mnt
4 /misc
4 /media
4 /initrd
4 /data
4 /.automount
0 /sys
0 /proc
0 /home
0 /dev

/boot
-----
20172 /boot
280 /boot/grub
268 /boot/boot
```

```
/home
-----
2415744 /home
1507560 /home/kwall
908084  /home/kwallold
52      /home/bubba
48      /home/gnuser

/space
-----
8983104 /space
2899376 /space/slackware
2517044 /space/redhat
1026864 /space/gcc
609984  /space/linux
300308  /space/my_documents
292036  /space/kwall
223428  /space/freebsd
213788  /space/downloads
70952   /space/browsers
29784   /space/textmaker
15600   /space/packages
4428    /space/pilot
0       /space/debian

/data
-----
2192    /data
132     /data/zion
16      /data/rhlnsa3
16      /data/lost+found
12      /data/tests
```

The script might take a while to run, especially on large disk drives and file systems with deeply nested subdirectories.

Listing A-14, `vmstat.sh`, runs the `vmstat` command, redirecting its output to a file for later analysis using a spreadsheet or other analysis tool. `vmstat.sh` is intended to run as a cron job.

```
#!/bin/bash
# vmstat.sh - Collect vmstat reports for later analysis

DELAY=${1:-60}
LOG=/var/log/vmstat.log.$$
[ $UID -eq 0 ] || LOG=$HOME/vmstat.log.$$

vmstat -n $DELAY >> $LOG
```

Listing A-14 Collecting `vmstat` statistics for offline analysis.

The first statement sets the default log file to which `vmstat`'s output will be saved. The next statement changes this default if the invoking user is not root, because normal users do not have write permission in the `/var/log` directory tree. The `$$` operator returns the script's PID and is used in this case to distinguish one log file from another. The next statement uses one of Bash's substitution operators to set a default value for number of seconds between `vmstat`'s updates. The `:` operator sets `$DELAY` to the value of the positional parameter 1 (referenced using `$1`) or assigns a default value of 60 if 1 is null or not set. Note that the script has to use the `:` operator rather than `:=`, because the positional parameters are read only at runtime and the "natural" substitution operator to use in this case, `:=`, would cause an error because it attempts to set the 1's value. Finally, the script calls `vmstat`, using `-n` to print the header only once, with a delay as specified, and redirecting the output to the file specified by `$LOG`.

`vmstat.sh`'s output should resemble the following after running a few minutes:

```
procs -----memory----- ---swap-- -----io----- --system-- -----cpu-----
 r  b   swpd   free   buff  cache   si   so    bi   bo    in    cs us sy id wa
 2  0   5072   4828    32 223856    0    0    19   13   191   163 84  3 13  0
 3  0   5072   4504    32 225764    0    0   533    2 1219   802 89 11  0  0
 3  0   5072   4944    32 227312    0    0   527    0 1231   806 90 10  0  0
 3  0   5072   4432    32 223792    0    0   572    0 1297  2736 81 19  0  0
 3  0   5072   4640    32 225036    0    0   482    0 1314  1117 89 11  0  0
 1  1   5072   4616    32 226196    0    0   600   15 1277   936 88 12  0  0
```

In this case, the output was stored in the file `/home/kwall/vmstat.log.17890`. Of course, on your system, the file would have a different PID appended to the end of the log file's name.

Summary

This appendix discussed shell programs and scripts as tools to ease the burden of system administration by handling certain tasks programmatically. First, you received a thorough introduction to the basics of Bash shell programming, including wildcards, shell variables, operators, Bash's flow control structures, shell functions, input and output processes, and command line arguments. The chapter presented a number of scripts that showed many of these shell-programming features in action. You reviewed scripts for working with running processes, killing processes, reporting disk usage, and logging `vmstat` reports.

SYMBOLS AND NUMERICS

- * (asterisk) as Bash wildcard, 906–908
- \$ (dollar sign)
 - in Bash positional parameters, 911, 912
 - in Bash variable format, 910
 - \$mynetworks configuration variable, 484
 - \$PGDATA environment variable, 366–367, 376
 - \$PGDATABASE environment variable, 376
 - \$PGUSER environment variable, 376
 - \$relay_domains configuration variable, 484
- . (dot) before shell configuration script names, 165
- ! (exclamation mark)
 - in /etc/group file, 720, 721
 - in /etc/shadow file, 711
 - to reverse Bash set notation, 907
 - as Sudo shell access command, 737
- # (hash character) for comments, 139
- (minus sign), deleting files starting with, 889
- ? (question mark) as Bash wildcard, 906–908
- / (root directory or partition)
 - creating with Disk Druid, 47
 - moving data to another partition, 142

- overview, 143–144
- root directory versus, 144
- ;(semicolon) terminating MySQL commands, 357
- [] (square brackets) as Bash set operators, 906–907
- ' (strong quote) in Bash, 908–909
- " (weak quote) in Bash, 909

A

- Accelerated Graphics Port (AGP), 31
- access control. *See also* SELinux (Security-Enhanced Linux)
- ACLs for root's power, 731
- Apache features, 514
- BIND server, 451, 455
- NFS server, 281–283
- PostgreSQL privileges, 368–372
- xinetd for, 437
- access.conf file, 520
- ACPI (Advanced Configuration and Power Interface), 643
- Actions menu (GNOME with RHEL), 109–110
- Add Partition dialog box, 49–50
- Address Resolution Protocol (ARP), 232
- Administration Tools Package Details dialog box, 72, 73

- administrative user, 731. *See also* Sudo (superuser do)
- Advanced Configuration and Power Interface (ACPI), 643
- Advanced Linux Sound Architecture (ALSA) Web site, 886
- Advanced Maryland Network Disk Archiver. *See* AMANDA
- Advanced Power Management (APM), 176, 643, 903
- AGP (Accelerated Graphics Port), 31
- Alert icon (Red Hat Network), 598
- aliases. *See also* device aliases
 - shell functions versus, 928
 - in shell scripts, 928
 - for Sudo, 734–735
- aliases files
 - for Mailman mailing lists, 558
 - for Postfix mail server, 169, 481
 - for Sendmail program, 169, 476–478
 - updating the binary aliases database, 481, 558
- ALSA (Advanced Linux Sound Architecture) Web site, 886
- AMANDA (Advanced Maryland Network Disk Archiver)
 - advanced features of, 783
 - .amandahosts file, 803
 - client configuration, 802
 - commands, 795–796
 - compression types, 801
 - configuring amanda.conf file, 797–802
 - disklist file, 802
 - dumping strategies, 801–802
 - dumptype block, 799–802
 - dumptype options, 799–801
 - holdingdisk blocks, 798
 - installing, 796
 - interface block, 798–799
 - options, 797
 - packages, 795
 - performing backups with, 803
 - tapetype block, 798
- amdump command, 803
- anaconda-ks.cfg file, 73, 79
- Apache Web server. *See also* /etc/httpd/conf/httpd.conf file; Web services
 - access controls, 514
 - Apache 2 changes, 516–517
 - challenges for configuring, 519
 - checking if installed, 519
 - content negotiation, 515
 - CPU and performance of, 593
 - default or primary server, defined, 523
 - default server configuration directives, 524–527
 - default server configuration
 - discussion, 530–536
 - default server configuration listing, 527–530
 - described, 436
 - development of, 511–512
 - DSOs or modules, 514
 - enabling CGI, 543–544
 - enabling PHP, 545–546
 - features, 512–515
 - forcing reread of configuration file, 540
 - further information, 512, 517, 520
 - global behavior configuration, 521–524
 - HTTP Configuration Tool, 539
 - logging support, 515
 - MIME subsystem, 520
 - multiple directory index files for, 514
 - multiprocessing modules (MPMs), 516–517
 - origin of name, 513
 - as part of LAMP, 353
 - performance advantages of, 512–513
 - popularity of, 511, 512
 - RAM and performance of, 593
 - required for SquirrelMail, 563
 - SSI implementation, 540–542
 - SSL for secure server, 546–554
 - stability of, 513–514
 - starting, 539
 - startup process, 520–521
 - stopping, 539
 - third-party modules supporting MySQL, 353–354

- virtual host or virtual server, 523, 537–539
 - Web services optimization, 590–593
 - APM (Advanced Power Management), 176, 643, 903
 - applets, 102–103, 121–122
 - application software. *See also* packages; RPM (Red Hat Package Manager); *specific applications*
 - checking versions, 764–767
 - directories for, 6
 - GNOME menus for, 103–104, 107–108
 - groups of users for, 6
 - installing from SRPMs, 771–778
 - KDE menu for, 122–124
 - obtaining newer software, 767–771
 - performance issues, 417
 - recompiling X applications, 417
 - “skeleton” configurations for, 6–7
 - starting at system boot, 137
 - starting in GNOME, 109
 - switching between programs, 102, 121
 - system administrator duties for, 6–7
 - Applications menu
 - GNOME, 103–104, 107–108
 - KDE, 122–124
 - /arch/defconfig file, 631
 - /arch/defconfig.mumble file, 631
 - arithmetic operators (Bash), 916–917
 - ARP (Address Resolution Protocol), 232
 - ASCII sort order, 913
 - asterisk (*) as Bash wildcard, 906–908
 - AT (ISA) bus, 31
 - at command, 702–704
 - “At the Forge—Syndication with RSS” (Lerner), 573
 - atd daemon, 702–704
 - atq command, 703–704
 - atrm command, 704
 - ATrpms (Axel Thimm’s RPMs) Web site, 768–769
 - authconfig applications, 872–873
 - authentication. *See also* Kerberos
 - authentication service
 - Kickstart Configurator settings, 84–85
 - log for privilege messages, 172–173
 - NIS for sharing information, 310
 - OpenLDAP for, 860–864
 - PostgreSQL trust authentication, 368–372
 - VNC, 388–389
 - Authentication Configuration tool, 328–329
 - autofs automount service (NFS), 301–304
 - automating scripts
 - benefits of, 674
 - one-shot jobs with at, 702–704
 - scheduled jobs with cron, 704–705
 - Axel Thimm’s RPMs (ATrpms) Web site, 768–769
- B**
- backgrounds, X performance and, 417
 - Backgrounds and Emblems dialog box (Nautilus), 117
 - backing up. *See also* restoring
 - advanced tools for, 795–803
 - AMANDA for, 795–803
 - amdump command for, 803
 - cdrecord package for, 787–789
 - choosing media for, 781
 - command line tools for, 784–795
 - creating a backup plan, 779–781
 - in disaster recovery plan, 26
 - dump package for, 789–790
 - full backups, 782, 783
 - incremental backups, 782–783
 - methods for, 781–783
 - mirrored disks for, 783
 - mt-st package for, 784–787
 - need for, 7–8
 - strategy needed for, 8
 - system administrator duties for, 7–9
 - tape rotation, 783–784
 - Bash shell
 - arithmetic operators, 916–917
 - bashrc configuration script, 165
 - case structure, 920, 924–926
 - character classes, 926
 - command line arguments, 934–936
 - comparison operators, 913–916
 - configuration files read on startup, 167
 - determinate loops, 922–923

Bash shell (*continued*)

- dsksug.sh script, 938–940
- echo command with, 932
- features for programming, 905
- file test operators, 917–919
- flow control overview, 919–920
- for statement, 920, 922–923
- getopts command, 934–935
- if statements, 920–922
- indeterminate loops, 923–924
- kujob.sh script, 936–938
- metacharacters, 906–909
- positional parameters, 911, 912
- processes and job control, 936–941
- profile configuration script, 165–167
- read command, 932–934
- redirecting I/O, 929–932
- return values and, 921
- reversing set notation, 907
- select control statement, 920, 924–925, 926–928
- set operator, 906–907
- shell functions, 928–929
- string I/O, 932–934
- until statement, 920, 923–924
- variables, 909–912
- vmstat.sh script, 940–941
- while statement, 920, 923–924
- wildcards and special characters, 906–909

bashrc configuration script, 165

basic input/output system (BIOS), 128

bc arithmetic-processing language

- building the software package, 775–776
- configuring the build environment, 772
- configuring the source code, 773–775
- installing the software package, 777–778

steps for upgrading and installing, 771

testing the build, 776

unpacking the source code, 772–773

biffs (mail notifiers), 470

bin directory, 143

BIND (Berkeley Internet Name Daemon).

- See also* DNS (Domain Name System)
- caching server, defined, 448
- caching server configuration, 461

- checking your configuration, 464–466
- client performance tuning, 583–585
- debugging, 465
- dig program, 465–466
- /etc/logrotate.d/named file, 586–587
- /etc/named.conf file, 449–458, 461–464, 583–586
- /etc/rc.d/init.d/named script, 449
- /etc/resolv.conf file, 446, 461
- finding IP address for specific domain name, 464
- host program, 464
- installing, 446–447
- logging, 451, 586–587
- master server, 448
- obtaining latest release, 446
- performance tuning, 585–587
- primary master server configuration, 462–464
- reverse lookups, 465–466
- reverse zone file, 449, 460
- root name server, 447
- rotating the log file, 586–587
- RPM version, 446–447
- secondary master server configuration, 462
- server performance tuning, 585–587
- slave server, 448
- /var/named/named.ca file, 449
- /var/named/named.local file, 449
- zone files, 449, 458–460

BIOS (basic input/output system), 128

bombing (email), 488

bookmarks (Nautilus), 113–114

boot directory, 143

boot disk

- emergency boot disk, 904
- floppy disk capacity for, 32
- for kernel upgrade, 618–619
- for kickstart installation, 93–94
- need for, 35, 51
- for standard installation, 35–36

boot loader. *See* GRUB (Grand Unified Boot Loader)

boot partition, 53, 54, 77

- boot process
 - BIOS and, 128
 - configuration file for, 179–180
 - first hard drive to boot, 128
 - GRUB and, 128–132
 - kernel and, 132–133
 - `rc.sysinit` program and, 133
 - runlevel 5 script for, 133–135
 - `/sbin/init` program and, 133–135
 - `/boot/grub/grub.conf` file, 139–140, 170–171, 670
 - booting. *See also* rebooting
 - boot process hangs, 895
 - boot process overview, 128–135
 - bootable CD-ROM and, 35
 - disabling NIC on boot, 55
 - Firstboot, running after standard installation, 65–70
 - from floppy disk drive, 32
 - GRUB installation, 51–54
 - to MS-DOS for CD-ROM access, 35
 - need for understanding, 127
 - profiles for multiple configuration sets and, 262
 - starting Dovecot at boot time, 485
 - starting mailman service at boot time, 558
 - starting NIS servers at boot time, 322–323
 - starting programs at boot time, 137
 - starting vsftpd at boot time, 493, 501
 - system across the network, 83
 - troubleshooting problems, 899–900
 - updating system time and, 698
 - verifying Postfix startup at boot time, 481
 - verifying Sendmail startup at boot time, 473
 - BOOTP protocol, 83
 - bootstrapping a database server, 352
 - broadcast addresses, 235
 - `bunzip2` command, 628
 - bus topology, 16–17
 - buses
 - defined, 30
 - kernel configuration options, 643–644
 - support for, 31
 - types of, 30–31
- ## C
- cabling for networks, 15, 16
 - caching proxy server, 393, 406–407. *See also* Squid Web proxy
 - calendar, displaying in GNOME, 101
 - capability bounding set, 731
 - CAs (certificate authorities), 547, 548, 549, 554
 - case structure (Bash), 920, 924–926
 - CD-R and CD-RW drives
 - cdrecord package for backups, 787–789
 - icon in GNOME, 100
 - kernel configuration options, 646–647
 - other backup methods versus, 781
 - support for, 33
 - cdrecord package, 787–789
 - CD-ROM drives
 - boot capability, 35
 - booting to MS-DOS for access, 35
 - creating a bootable CD-ROM, 94
 - file systems support, 661
 - kernel configuration options, 646–647, 648
 - mounting, troubleshooting, 885
 - not detected during installation, 884–885
 - overview, 33
 - Center for Information Technology Integration (CITI), 269
 - central processing units (CPUs). *See* processors
 - certificate authorities (CAs), 547, 548, 549, 554
 - certificate signing request (CSR), 550
 - certificates. *See* SSL (Secure Sockets Layer)
 - CGI (Common Gateway Interface), 543–544
 - `chage` command, 708, 714–715
 - character classes (Bash), 926
 - chat protocols, real-time, 435–436
 - `chattr` command, 691, 692–693
 - `chgrp` command, 691, 692

- chkconfig command
 - listing information for services, 200
 - listing xinetd-based services, 200–201
 - starting Dovecot at boot time, 485
 - starting mailman at boot time, 558
 - starting vsftpd at boot time, 493
 - starting ypbind, 331
 - turning services on or off, 202
 - verifying Postfix startup at boot time, 481
 - verifying Sendmail startup at boot time, 473
- chmod command, 691–692
- chown command, 691, 692
- chpasswd utility, 715, 716–717
- chsh command, 708, 714
- CIDR (Classless InterDomain Routing), 244–245, 275
- CITI (Center for Information Technology Integration), 269
- class A, B, and C networks. *See* network classes
- clients, defined, 18
- client-server architecture
 - Kerberos and, 864
 - of LDAP, 852
 - mixed networks and, 19–20
 - of NFS, 266–267
 - of NIS, 310
 - overview, 18
- clock. *See* date and time, system; time server
- clock file, 177
- clustering, 24–25
- coaxial cable, 16
- color depth
 - changing, 207, 214
 - Kickstart Configurator setting, 87
 - setting with Firstboot, 68–69
 - X performance and, 417
- command line
 - advantages of configuring from, 673–675
 - for automating scripts, 674, 702–705
 - backup tools, 784–795
 - Bash arguments, 934–936
 - directories for commands and utilities, 675
 - file system maintenance from, 683–696
 - graphical tools versus, 673–674
 - GRUB interface, 131–132
 - multiword arguments for commands, 889–890
 - non-Red Hat-derived systems and, 674
 - process management from, 675–682
 - RAID tools, 162
 - RPM options, 746–747
 - timekeeping functions, 696–702
 - user account commands, 708
- comments in configuration files, 139
- Common Gateway Interface (CGI), 543–544
- Common Unix Printing System (CUPS), 220
- comparison operators (Bash), 913–916
- compat-openldap package, 856
- Computer icon (GNOME), 99–100
- conditional execution (Bash)
 - case structure, 920, 924–926
 - defined, 919–920
 - if statements, 920–922
 - select control statement, 920, 924–925, 926–928
- configuration files. *See also specific files and directories*
 - for AMANDA, 797–802
 - for Apache, 436, 520
 - for BIND client, 583–585
 - for BIND server, 449–460, 585–586
 - comments in, 139
 - directory for configuration scripts, 164
 - for DNS, 437, 449–458
 - in /etc/sysconfig directory, 175–188
 - for ftpd, 436
 - for GRUB, 139–140, 170–171
 - for ht://Dig, 575–576
 - init scripts, 196–202
 - for Kerberos, 867–869, 870
 - for kernel, 630–632
 - for LDAP, 582–583, 857–859
 - managing init scripts, 196–202

- network configuration files,
 - 188–196, 892
- for NFS server, 273, 274–280
- for NIS clients, 326–330
- for NIS servers, 315, 316–318
- for ntpd, 397–398
- for Postfix, 480–482
- for RPCSEC_GSS, 279–280, 290
- for Samba, 341–344
- security issues, 163–164
- for SELinux, 842–843
- for Sendmail, 474–475
- shell configuration scripts, 164–167
- for sound modules, 886
- for Squid, 409–411
- for SquirrelMail, 565
- for sshd, 436
- for Sudo, 733–737
- for system environmental settings,
 - 168–175
- for vsftpd, 495
- for X server, 210–215
- configuration variables (Postfix), 484
- configure command, 773–775
- configuring the kernel. *See* customizing the kernel; kernel configuration options
- content negotiation, 515
- Coordinated Universal Time (UTC),
 - 59–60, 74, 698, 701
- CPUs (central processing units).
 - See* processors
- crackers, 12
- cramfs virtual file system, 149
- createdb utility (PostgreSQL),
 - 372–373, 376
- createuser utility (PostgreSQL),
 - 373–374, 376
- cron daemon
 - configuration file for settings, 178
 - crontab files for, 171–172, 704–705
 - log for cron messages, 173
 - running scheduled jobs with, 704–705
- crontab command, 704–705
- csh.cshrc configuration script, 165
- CSR (certificate signing request), 550

- CUPS (Common Unix Printing System),
 - 220
- customizing the kernel. *See also* kernel configuration options
 - compiling the kernel, 666–667
 - configuration targets, 629–630
 - defined, 618
 - features for, 615
 - Intel IA-64 architecture configuration files, 632
 - loadable kernel module (LKM), 633
 - location for building the kernel, 630
 - make allnoconfig command, 630
 - make config command, 629
 - make defconfig command, 631
 - make gconfig command, 629
 - make menuconfig command,
 - 629, 634–637
 - make oldconfig command,
 - 629, 630, 632
 - make xconfig command,
 - 629, 633–634, 665–666
 - multiple-platform systems and, 632
 - selecting a kernel configuration file,
 - 630–632
- cyrus-imapd startup error,
 - troubleshooting, 900–901

D

- daemons, 224. *See also* specific daemons
- database servers. *See also* MySQL; PostgreSQL
 - basic bootstrapping procedure, 352
 - overview, 351–352
 - performance tuning, 424–425
- date and time, system. *See also* time server
 - clock configuration file and, 177
 - displaying with date, 697, 698–702
 - Kerberos timestamps and, 864
 - not updating after system boot, 698
 - setting hardware clock, 697–698
 - setting in GNOME, 101
 - setting with date, 698, 701–702
 - setting with Firstboot, 67
 - syncing with time server, 697, 702
 - systems important for, 696

- date command
 - described, 697
 - output format tokens for, 699–700
 - syntax for displaying, 698–700
 - syntax for setting, 698, 701–702
- Date/Time Properties tool, 401–403
- debugging
 - displaying information with RPM, 747
 - DNS servers, 465
- deleting or removing
 - applets from GNOME panel, 103
 - applets from KDE panel, 121
 - bookmarks (Nautilus), 114
 - files, troubleshooting, 889
 - group accounts, 723, 729–730
 - initrd directory, avoiding, 144
 - jobs from atd queue, 704
 - MySQL anonymous accounts, 358
 - NFS shares, 295
 - NICs, 259
 - PostgreSQL database, 375, 376–377
 - PostgreSQL user, 377
 - printers, 227
 - RPM database entries, 747
 - RPMs, 758
 - user account passwords, 713
 - user accounts, 713, 728
 - users from group accounts, 722
- Desktop menu (GNOME), 106–107
- determinate loops (Bash), 922–923
- dev directory, 45, 143
- devfs (Device File System), 154
- device aliases
 - defined, 260
 - DHCP and NIC aliases, 238
 - for multiple IP addresses with one NIC, 260
- device drivers
 - defined, 30
 - print driver selection, 224–225
 - support for, 30
 - for X server, 417
- Device File System (devfs), 154
- device files
 - defined, 45
 - devfs virtual file system for, 154
 - /dev/pts virtual file system for, 154
 - hard disk and floppy drive names, 45
- device is busy message, 887, 890
- /dev/pts virtual file system, 154
- df command, 695–696
- DHCP (Dynamic Host Configuration Protocol)
 - advantages of, 247
 - aliases for NICs and, 238
 - client configuration, 250
 - dhcpcd.conf configuration file, 249–250
 - disabling when installing Red Hat, 55
 - obtaining information automatically via, 56, 83
 - server setup, 248–250
- dhcpcd program, installing, 248–249
- dig DNS server tool, 465–466
- directories, LDAP. *See* LDAP (Lightweight Directory Access Protocol)
- directories or folders. *See also* file systems; home directories; *specific directories*
 - for application software, 6
 - Bash file test operators, 917–919
 - for commands and command-line utilities, 675
 - commands for working with, 691–695
 - for configuration scripts, 164
 - for device files, 45
 - displaying contents in Nautilus, 112
 - managing in Nautilus, 114–115
 - modifying group ownership, 691, 692
 - modifying ownership, 691, 692
 - modifying permissions, 691–692
 - root directory (/), 143–144
 - for runlevel scripts, 133
 - vsftpd download directory, 495
- directory not empty message, 889
- disaster recovery plan
 - backup schemes, 26
 - cost issues, 25
 - defined, 23–24
 - fault tolerance and, 9, 23
 - need for, 9, 23

- overview, 25–26
- questions to ask, 24
- redundancy, 26
- surge protection, 25–26
- UPS systems, 25
- writing it down, 27–28
- Disk Druid
 - Add Partition dialog box, 49–50
 - buttons and their functions, 48–49
 - fields for partitions, 47–48
 - preparing to partition with, 47–48
 - setting up partitions, 49–51
- display resolution. *See* screen resolution
- Display Settings dialog box
 - Color Depth drop-down list, 207
 - Dual Head tab, 209–210
 - Hardware tab, 207–208, 209
 - Resolution drop-down list, 206
 - Settings tab, 206–207
- displaying or viewing
 - all files in installed RPM, 752
 - all installed RPMs, 751
 - banner message with vsftpd, 500
 - calendar in GNOME, 101
 - current network configuration, 233–234
 - current partitioning and file systems, 155
 - debugging information, 747
 - /etc/sysconfig directory files list, 175–176
 - file system quota utilization, 742–744
 - folder contents with Nautilus, 112
 - I/O activity with iostat, 822–826
 - IRQs, 152–153
 - jobs in atd queue, 703–704
 - license agreement, 66–67
 - log files, 881–882
 - logical volumes on the system, 159
 - login information, 717–718
 - md devices list, 162
 - memory usage, 679–680, 806–812
 - NFS statistics, 421–422
 - partition information, 155
 - physical volumes on the system, 159
 - process behavior over time with top, 679–680, 817–821
 - process IDs using a file, 691
 - process listings, 150–151
 - processes users are running, 717–718
 - processes with ps, 676–678, 813–817
 - real-time CPU usage, 679–680
 - RPM descriptive information, 751–752
 - RPM name and version number, 751
 - RPM that installed a file, 753
 - running tasks, 812–821
 - SELinux current status, 842
 - services with chkconfig, 200, 201
 - software version numbers, 766–767
 - status message for users, 713
 - system date and time, 697, 698–702
 - volume groups on the system, 159–160
- dmesg command, 785–786, 879
- DNS (Domain Name System). *See also*
 - BIND (Berkeley Internet Name Daemon)
 - client performance tuning, 583–585
 - country domains, 445
 - /etc/hosts file and, 445–446
 - /etc/nsswitch.conf file and, 445
 - /etc/resolv.conf file and, 446
 - name address resolution by, 443–444
 - name resolution setup, 190–191
 - NIS domains versus DNS domains, 311
 - optimizing services, 583–587
 - overview, 437, 443–446
 - server performance tuning, 585–587
 - top-level U.S. domains, 444
- documentation
 - of disaster recovery plan, 27–28
 - installing documentation disks, 69–70
 - of network design, 26–27
 - security log, 27, 28
 - of security policy, 27
 - for troubleshooting, 878
- dollar sign (\$)
 - in Bash positional parameters, 911, 912
 - in Bash variable format, 910
 - \$mynetworks configuration variable, 484
 - \$PGDATA environment variable, 366–367, 376
 - \$PGDATABASE environment variable, 376

- dollar sign (\$) (*continued*)
 - \$PGUSER environment variable, 376
 - \$relay_domains configuration variable, 484
- Domain Name System. *See* DNS
- domainname command, 316
- domains (DNS)
 - country domains, 445
 - defined, 311
 - finding IP address for specific domain name, 464
 - NIS domains versus, 311
 - subdomains, 445
 - top-level U.S. domains, 444
 - types of domain servers, 447–448
- domains (NIS)
 - associating client and server domains, 327
 - DNS domains versus, 311
 - multiple-domain configurations, 311, 312–314
 - overview, 310–311
 - partitioning, 423
 - setting the domain name, 316, 326
 - single-domain configurations, 311–312, 313
- domains (NT, for Samba), 343, 344
- DOS file system support, 661
- dot (.) before shell configuration script names, 165
- Dovecot IMAP server, 485–486
- drivers. *See* device drivers
- dropdb utility (PostgreSQL), 375, 376–377
- dropuser utility (PostgreSQL), 376, 377
- dsksug.sh Bash script, 938–940
- DSOs (Dynamic Shared Objects), 514
- du command, 695, 696
- dual monitor configuration, 209–210
- dump package, 789–790
- DVD drives
 - file systems support, 661
 - kernel configuration options, 646–647
 - other backup methods versus, 781
 - overview, 33
- dynamic acquisition, 233

- Dynamic Host Configuration Protocol.
See DHCP
- Dynamic Shared Objects (DSOs), 514

E

- echo command, 246, 932
- Edit Bookmarks dialog box, 113–114
- edquota command, 695, 738, 740–742
- EEPROM (erasable programmable read-only memory) BIOS chip, 128
- EISA (Extended Industry Standard Architecture) bus, 31
- emacs, checking version of, 766
- email. *See also specific programs and protocols*
 - aliases files, 169, 476–478, 481
 - biffs (mail notifiers), 470
 - bombing, 488
 - delivery process for, 468–470
 - file system quotas for mailboxes, 472
 - firewall configuration for server, 58
 - IMAP4 for, 471, 472, 484–486
 - key components for, 468
 - local delivery agents (LDAs), 469
 - log for mail messages, 173
 - mail delivery agents (MDAs), 469
 - mail spool or mailbox file, 469
 - mail transfer agents (MTAs), 469
 - mail user agents (MUAs), 468–469
 - MIME implementation, 470–471
 - optimizing services, 587–590
 - overview, 467–470
 - POP3 for, 471, 472, 484–486
 - Postfix mail server, 169, 479–484
 - security, 486–489
 - Sendmail program, 169, 186, 472–478
 - server not required for, 5
 - SMTP for, 58, 470–471, 488–489
 - spamming, 488
 - spoofing, 488
 - SSH tunneling for POP mailer, 431
 - troubleshooting, 472
 - Web-based email, 563–567
- emergency boot disk, 904. *See also* troubleshooting
- Emery, Van (“Learning NFSv4 with Fedora Core 2”), 269

- encryption
 - for email, 487
 - GPG, 599, 600–601
 - kernel configuration options, 664–665
 - MD5 encryption, 84
 - NTP autokey encryption, 399–401
 - SSH, 430–431
 - SSL, 547
- environment variables (PostgreSQL), 366–367, 376
- environmental settings. *See* system environmental settings
- erasable programmable read-only memory (EEPROM) BIOS chip, 128
- error messages. *See also* log files and logging; troubleshooting
 - cyrus-imapd startup error, 900
 - device is busy, 887, 890
 - directory not empty, 889
 - fatal signal 7 or fatal signal 11, 902–903
 - ht:/ /Dig errors, 900
 - mount failed, 898
 - NFS failure messages, 896–897
 - no such file or directory, 889–890
- etc directory. *See also specific files and subdirectories*
 - described, 143
 - permissions for files in, 163–164
 - system environmental settings files in, 168–174
- /etc/aliases file, 169, 477–478, 481, 558
- /etc/amanda/test/amanda.conf file
 - compression types, 801
 - disklist file, 802
 - dumping strategies, 801–802
 - dumptype block, 799–802
 - dumptype options, 799–801
 - holdingdisk blocks, 798
 - interface block, 798–799
 - options, 797
 - tapetype block, 798
- /etc/auto.master file, 301–302
- /etc/dhcpd.conf file, 249–250
- /etc/exports file (NFS)
 - all_squash and no_all_squash options, 278
 - described, 273
 - dir specification, 274
 - export options (table), 276–277
 - format, 274
 - hide and nohide options, 279
 - host specification, 274, 275–276
 - mp option, 279
 - NFS Server Configuration tool for, 292–295
 - no_wdelay option, 279
 - root_squash and no_root_squash options, 277–278, 306, 730
 - security issues, 305
 - space after opening parenthesis, avoiding, 274–275
 - subtree_check and no_subtree_check options, 278, 279
 - sync option, 279
- /etc/fstab file, 155, 169–170, 738–739
- /etc/group file, 501–502, 718, 860–861
- /etc/grub.conf file, 170–171
- /etc/gshadow file, 720, 722–723
- /etc/gssapi_mech.conf file, 280
- /etc/hosts file
 - adding computers not in DNS, 261–262
 - DNS name resolution and, 445–446
 - /etc/nsswitch.conf file and, 236, 330, 445–446
 - hostname setup and, 190
 - IP address to hostname mappings in, 191–192
 - local network computers in, 236, 237
 - monitored servers in, 191
 - name resolution and, 446
- /etc/hosts.allow file, 281–283
- /etc/hosts.deny file, 281–283
- /etc/htdig/htdig.conf file, 575–576
- /etc/httpd/conf.d/mailman.conf file, 557
- /etc/httpd/conf/httpd.conf file
 - AcceptPathInfo directive, 532
 - AddCharset directive, 525, 536

- /etc/httpd/conf/httpd.conf file
(*continued*)
- AddDefaultCharset directive,
525, 536
- AddHandler directive, 525, 535, 536
- AddIcon directives, 525, 535
- AddIconByEncoding directives,
525, 535
- AddIconByType directives, 525, 535
- AddLanguage directives, 525, 536
- AddOutputFilter directive, 525, 540
- AddType directive, 525, 535, 540
- Allow directives, 532–533, 592
- AllowOverride directive, 532
- Apache startup process and, 520
- BrowserMatch directives, 536
- for CGI, 543
- default server configuration directives,
524–527
- default server configuration
discussion, 530–536
- default server configuration listing,
527–530
- DefaultType directive, 526, 534
- Deny directives, 532–533, 592
- <Directory></Directory> block,
526, 530–531, 540–541
- DirectoryIndex directive, 526, 534
- DocumentRoot directive, 526, 530–531
- ErrorLog directive, 526, 534
- forcing Apache to reread the file, 540
- global behavior configuration, 521–524
- Group directive, 526, 530
- HostNameLookups directive, 526, 591
- KeepAlive directive, 521, 523
- KeepAliveTimeout directive, 521, 523
- LanguagePriority directive,
526, 536
- Listen 80 directive, 521, 523
- LogFormat directives, 527, 534–535
- MaxClients directive, 521, 523–524,
591
- MaxKeepAliveRequests directive,
521, 591
- MaxRequestsPerChild directive,
521, 591
- MaxSpareServers directive, 521, 523
- MIMEMagicFile directive, 533, 534,
535
- MinSpareServers directive, 521, 523
- number of directives for, 520
- Options directives, 531–532, 540,
541, 592
- Order directive, 532–533
- ScriptAlias directive, 527, 543
- ServerLimit directive, 522, 524
- ServerName directive, 527, 530
- for SSI implementation, 540–541
- StartServers directive, 522, 523
- Timeout directive, 522, 591
- TypesConfig directive, 527, 534, 535
- User directive, 517, 530
- UserDir directive, 527, 533
- virtual server configuration, 537–539
- /etc/httpd/conf/httpd.conf.rpm
new file, 892
- /etc/httpd/conf/httpd.conf.rpm
orig file, 892
- /etc/httpd/conf/httpd.conf.rpm
save file, 892
- /etc/idmapd.conf file, 279–280, 290
- /etc/inittab file, 136–137, 197–198
- /etc/issue file, 168
- /etc/issue.net file, 168–169
- /etc/krb5.conf file, 867–868, 870
- /etc/logrotate.d/named file,
586–587
- /etc/mail/sendmail.cf file,
474–475, 587–588
- /etc/mail/sendmail.mc file,
475, 476
- /etc/mime.types file, 520
- /etc/modprobe.conf file, 886
- /etc/motd file, 168
- /etc/named.conf file
acl statement, 451, 455
- caching DNS server configuration, 461
- caching-only server example, 449–450
- client performance tuning, 583–585
- comment lines, 450
- described, 449
- example configuration, 583–584
- include statement, 451, 454
- key statement, 451

- logging categories, 456–457
- logging statement, 451, 455–457
- options statement, 451–454, 584–586
- primary master DNS server
 - configuration, 462–464
- restarting named daemon after
 - changing, 464
- secondary master DNS server
 - configuration, 462
- server performance tuning, 585–586
- server statement, 451, 457
- zone statements, 451, 457–458
- /etc/nsswitch.conf file
 - DNS name resolution and, 445
 - /etc/hosts file and, 236, 330, 445–446
- example configuration, 192–193
- hosts line in, 192, 236
- NIS client daemon setup, 330–331
- /etc/ntp.conf file, 397–401, 403–404
- /etc/passwd file. *See also* user accounts
 - creating LDIF files from, 860–861
 - entry example, 709
 - /etc/shadow file and, 709, 711
 - fields, 709
 - format, 708
 - gecos field, 709
 - overview, 708–709
 - parsing with Bash, 933
 - removing ftp user from, 501–502
 - shell field security measures, 709
- /etc/postfix/main.cf file, 480–481, 482–483
- /etc/postfix/master.cf file, 588–590
- /etc/postfix/transport file, 588
- /etc/profile script, 165–167
- /etc/raidtab file, 161
- /etc/rc.d/init.d/named script, 449
- /etc/resolv.conf file, 236–237, 446, 461
- /etc/selinux/config file, 842–843
- /etc/services file, 849
- /etc/shadow file. *See also* user accounts
 - creating LDIF files from, 860–861
 - /etc/passwd file and, 709, 711
 - fields, 710–711
 - Kickstart Configurator setting, 84
 - password expiration information in, 710
 - passwords stored in, 709
 - removing ! from password in, 711
- /etc/squid/squid.conf file, 409–411
- /etc/sudoers file, 733–737
- /etc/sysconfig directory. *See also specific files and subdirectories*
 - files and configuration options, 175–176
 - storing configuration scripts in, 164
- /etc/sysconfig/apmd file, 176
- /etc/sysconfig/apm-scripts directory, 187
- /etc/sysconfig/authconfig file, 177
- /etc/sysconfig/clock file, 177
- /etc/sysconfig/crond file, 178
- /etc/sysconfig/daemons directory, 187
- /etc/sysconfig/desktop file, 178
- /etc/sysconfig/firstboot file, 178
- /etc/sysconfig/grub file, 178
- /etc/sysconfig/harddisks file, 178–179
- /etc/sysconfig/hwconf file, 179
- /etc/sysconfig/i18n file, 179
- /etc/sysconfig/init file, 179–180
- /etc/sysconfig/iptables file, 180, 195
- /etc/sysconfig/irda file, 181
- /etc/sysconfig/kernel file, 181
- /etc/sysconfig/keyboard file, 181–182
- /etc/sysconfig/kudzu file, 182
- /etc/sysconfig/mouse file, 182–183
- /etc/sysconfig/named file, 183
- /etc/sysconfig/netdump file, 183
- /etc/sysconfig/network file, 184, 190, 237
- /etc/sysconfig/networking directory, 187

- `/etc/sysconfig/network-scripts`
 - directory
 - `ifcfg-eth0` file, 235, 250
 - `ifcfg-networkinterfacename` file, 196
 - `ifup` and `ifdown` symlinks, 196
 - network configuration files in, 188, 196
- `/etc/sysconfig/ntpd` file, 184
- `/etc/sysconfig/pcmcia` file, 184–185
- `/etc/sysconfig/resolv.conf` file, 191
- `/etc/sysconfig/rhn` directory, 188
- `/etc/sysconfig/samba` file, 186
- `/etc/sysconfig/selinux` file, 185
- `/etc/sysconfig/sendmail` file, 186, 587–588
- `/etc/sysconfig/static-routes` file, 195
- `/etc/sysconfig/system-config-users` file, 185
- `/etc/sysconfig/system-logviewer` file, 185–186
- `/etc/sysconfig/vncservers` file, 186–187
- `/etc/sysconfig/xinetd` file, 187
- `/etc/sysconfig/xinetd.conf` file, 193–194, 438–439
- `/etc/sysconfig/yppasswdd` daemon, 315, 321
- `/etc/sysctl.conf` file, 409
- `/etc/syslog.conf` file, 172–174
- `/etc/vsftpd/ftpusers` file, 495, 496–497
- `/etc/vsftpd/user_list` file, 495, 496–497, 502
- `/etc/vsftpd/vsftpd.conf` file
 - configuring `vsftpd` features, 497–501
 - described, 495
 - enabling anonymous uploads, 504, 505
 - enabling guest user accounts, 506
 - running `vsftpd` over SSL, 508
- `/etc/xinetd.d` directory, 438, 441
- `/etc/yp.conf` file, 326–330
- `/etc/ypserv.conf` file, 315, 316, 317–318

- Ethernet cards. *See* NICs (network interface cards)
- `e2fsck` command, 683, 687
- exclamation mark (!)
 - in `/etc/group` file, 720, 721
 - in `/etc/shadow` file, 711
 - to reverse Bash set notation, 907
 - as Sudo shell access command, 737
- `exportfs` command, 285, 287–288
- Extended File System (XFS) of SGI, 148
- Extended Industry Standard Architecture (EISA) bus, 31
- extended partitions on x86 machines, 156
- `ext2` (extended 2) file system
 - `ext3` compatibility with, 145
 - overview, 146
 - resizing, 683, 687
 - special attributes, 691, 692–693
- `ext3` (extended 3) file system, 145–146, 683, 687

F

- failover clustering, 24
- FAT file system, 147, 661
- fatal signal 7 or fatal signal 11 message, 902–903
- fault tolerance
 - clustering solutions for, 24–25
 - cost issues, 24
 - defined, 9, 23
 - disaster recovery plan and, 9, 23
 - questions to ask, 24
- `fdisk` command, 155, 683–684
- Fedora Extras Web site, 769, 780
- FedoraProject.org site, 769–770
- `feed.rss` file, 570
- FHS (Filesystem Hierarchy Standard), 141–142
- FibreChannel disk arrays, 419
- File Management Preferences dialog box (Nautilus), 115–116
- file sharing. *See* NFS (Network File System)
- file system quotas
 - creating quota files, 739–740
 - enabling capability for, 738–739

- for mailboxes, 472
- need for, 737
- programs for managing, 695, 738
- set per file, not per disk, 738
- setting and modifying, 740–742
- steps for initializing, 738
- turning on, 740
- viewing utilization, 742–744
- file systems. *See also* backing up;
 - Nautilus File Manager (GNOME);
 - NFS (Network File System)
- accessing Windows file systems, 890
- checking and repairing, 683, 687
- choosing type for partition, 50, 79
- commands for creating and
 - manipulating, 683, 685–691
- commands for working with files and
 - directories, 691–695
- creating on a device, 683, 685–686
- deleting files, troubleshooting, 889
- disk management, 155–162
- ext2, 145, 146
- ext3, 145–146, 683, 687
- FAT, 147
- FHS, 141–142
- fstab configuration file for, 169–170
- IBM JFS, 147
- icon in GNOME, 100
- importance of understanding, 141
- Internet services performance and, 582
- kernel configuration options, 660–663
- LDAP performance and, 582
- maintaining from the command line,
 - 683–696
- managing disk space usage, 695–696
- mounting, 46, 683, 688–690
- moving data to another partition, 142
- network file systems, 662
- NFS performance and, 419
- NTFS file system, 147
- partition creation and manipulation
 - commands, 683–685
- pseudo-file-systems, 662
- quotas, 472, 737–744
- reiserfs, 146
- reporting disk usage for, 938–940
- resizing ext2 and ext3 file systems,
 - 683, 687
- root directory (/), 143–144
- root user powers restricted by, 730–731
- SGI XFS, 148
- Sistina GFS, 148–149
- structure, 141–144
- symbolic links, 135, 683, 688
- SystemV, 147
- troubleshooting, 888–891
- ufs, 147
- unmounting, 683, 690, 694–695
- Veritas FREEVxFS, 145
- VFS layer, 145
- virtual file systems, 149–154
- Windows versus Linux, 143
- file test operators (Bash), 917–919
- file transfer protocol. *See* FTP
- files. *See also* configuration files; *specific files*
 - Bash file test operators, 917–919
 - commands for working with, 691–695
 - deleting, troubleshooting, 889
 - displaying PIDs using, 691, 693–694
 - identifying files opened by a process,
 - 691, 694–695
 - managing in Nautilus, 114–115
 - modifying ownership, 691, 692
 - modifying permissions, 691–692
 - opening in Nautilus, 112–113
 - showing detailed information, 691, 693
 - special attributes on ext2 files,
 - 691, 692–693
 - static, 142
 - variable, 142
- Filesystem Hierarchy Standard (FHS),
 - 141–142
- finger service, 435
- firewalls. *See also* Netfilter (iptables)
 - firewall
 - configuring during standard
 - installation, 56–58
 - creating, 847–851
 - dedicated hardware appliances,
 - 847–848
 - defined, 847

- firewalls (*continued*)
 - egress control, 847
 - for email security, 487
 - /etc/sysconfig/iptables file, 180, 195
 - ingress control, 847
 - Kickstart Configurator settings, 86–87
 - modifying rules for Squid, 411–412
 - NFS security improved by, 305–306
 - payload examination by, 847
 - planning the network and, 23
 - running Postfix behind, 482–483
 - SMTP connections and, 488–489
 - VNC server configuration, 384–386
 - FireWire (IEEE 1394), 649, 781
 - Firstboot, 65–70, 87, 178
 - flash memory
 - cramfs virtual file system for, 149
 - other backup media versus, 781
 - floppy disk drives
 - boot floppy creation, 35–36, 93–94, 618–619
 - booting from, 32
 - device is busy error message, 890
 - file names for, 45
 - icon in GNOME, 100
 - other backup methods versus, 781
 - support for, 32
 - flow control (Bash)
 - case structure, 920, 924–926
 - for statement, 920, 922–923
 - if statements, 920–922
 - overview, 919–920
 - select control statement, 920, 924–925, 926–928
 - until statement, 920, 923–924
 - while statement, 920, 923–924
 - folders. *See* directories or folders; *specific folders*
 - font server, X performance and, 417
 - for statement (Bash), 920, 922–923
 - FQDN (fully qualified domain name), 311
 - free command, 805, 806–808
 - FREEVxFS (Veritas file system), 148
 - freshrpms.net site, 768
 - fsck.mount program, 169
 - FTP (file transfer protocol). *See also* vsftpd (Very Secure FTP Daemon)
 - core RFC, 492
 - ftpd daemon, 436
 - kernel source download via, 625–626
 - Nautilus for accessing sites, 113
 - NcFTPd for, 492
 - optimizing services, 590
 - outsourcing FTP hosting, 20–21
 - overview, 434
 - ProFTPD for, 492–493
 - scp as replacement for, 431, 434
 - sftp client for, 433, 510
 - sftp-server for, 509–510
 - FTP server. *See also* vsftpd (Very Secure FTP Daemon)
 - firewall configuration for, 57
 - help for, 434
 - kickstart installation from, 76
 - not required for remote access, 5
 - ftpd FTP daemon, 436, 493
 - fully qualified domain name (FQDN), 311
 - functions (Bash), 928–929
 - fuser command, 691, 693–694
- G**
- gateways
 - default, 232
 - defined, 246
 - IP forwarding, 246–247
 - routers, 232–233
 - running Postfix behind, 482–483
 - getenforce command (SELinux), 842
 - getopts command (Bash), 934–935
 - Gettys, Jim (“X Window System Network Performance”), 416
 - GFS (Global File System) of Sistina, 148–149
 - GFS (grandfather-father-son) tape rotation, 784
 - GIDs (group identification numbers), 718
 - GMT (Greenwich Mean Time), 59, 74
 - GNOME (GNU Network Object Model Environment)
 - Actions menu, 109–110
 - Applications menu, 103–104, 107–108, 109

- Computer icon, 99–100
- configuring, 118–119
- date and time display, 101
- as default desktop, 87, 97
- Desktop menu, 106–107
- home directory icon, 101
- Kickstart Configurator setting, 87
- logging out, 119
- managing applets on the panel, 102–103
- Nautilus File Manager, 99, 110–118
- panel overview, 101–102
- performance costs, 416
- Places menu, 105
- RHEL versus Fedora Core, 99, 101
- showing and hiding windows, 102
- shutting down from, 138
- starting applications, 109
- switching programs in, 102
- switching workspaces in, 102
- system volume setting, 101
- trash icon, 101
- VNC `xstartup` file and, 387–388
- `gpaswd` command, 719, 720–722
- GPG (GNU Privacy Guard), 599, 600–601, 606
- GPU (graphics processing unit), 416
- Grand Unified Boot Loader. *See* GRUB
- grandfather-father-son (GFS) tape rotation, 784
- graphical user interfaces. *See* GUIs
- Greenwich Mean Time (GMT), 59, 74
- `grep` command
 - finding ports and protocols used by services, 849
 - for keywords in configuration files, 202
 - for NTP-related system log entries, 405
 - pipng `rpmquery` output through, 765
 - for ssh server PID number, 150–151
- Groth, David (*Network Plus*), 235
- group accounts. *See also* user accounts
 - adding administrator to group, 720
 - adding users, 720, 722, 723
 - administrative commands, 719
 - assigning administrator for, 720
 - changing passwords, 721
 - creating, 719, 728–729
 - deleting groups, 723, 729–730
 - deleting users, 722
 - `/etc/group` file, 501–502, 718
 - excluding nonmembers from joining
 - with `newgrp`, 719, 721
 - GIDs for, 718
 - joining with `newgrp`, 721, 723
 - modifying with User Manager, 729–730
 - shadowed group files, 720, 722–723
 - user accounts versus, 718
 - user private groups (UPGs), 723–725
 - group identification numbers (GIDs), 718
- `groupadd` command, 719
- `groupdel` command, 719, 723
- `groupmod` command, 719, 720
- GRUB (Grand Unified Boot Loader)
 - Advanced Boot Loader Configuration screen, 53–54
 - boot floppy for kernel upgrade, 618–619
 - Boot Loader Configuration screen, 52–53
 - boot process and, 128–132
 - command-line interface, 131–132
 - configuration file, 139–140, 170–171, 670
 - installing with Red Hat, 51–54
 - kernel selection by, 129, 130
 - Kickstart Configurator settings, 77
 - location for, 53, 77
 - OS selection using, 129, 130
 - passing arguments at boot time, 178
 - passing kernel parameters, 130–131
 - password for, 52–53, 77
 - stage one process, 129–130
 - stage two process, 130
 - updating for new kernel, 670
 - using during boot, 130–132
- GUIs (graphical user interfaces). *See also* X Window system; *specific GUIs*
 - command line versus, 673–674
 - XFree86 server for, 5
- `gunzip` command, 773
- gzipped tarballs, 771

H

Hacking RSS and Atom (Orchard), 573
hard drives. *See also* file system quotas;
file systems; partitioning hard disks
Apache performance and, 593
boot process and, 128
buying, 32
changing drive order, 53
configuration file for tuning, 178–179
disk failure and metadevices, 156–157
distributing NFS exports across,
272, 419
file names for, 45
kernel configuration options, 648–649
kickstart installation from, 76
LDAP performance and, 582
managing disk space usage, 695–696
mirrored disks for backups, 783
NAS devices, 420
NFS performance and, 419
other backup media versus, 781
reporting usage for mounted file
systems, 938–940
support for, 32
hardware clock. *See also* date and time,
system; time server
setting, 697–698
for time server, 395, 405–406
Hardware Compatibility List, 893
hardware RAID, 160. *See also* RAID
(Redundant Array of Independent
Disks)
hardware support database, 20, 34
hash character (#) for comments, 139
Hesiod authentication, 85
home directories
defined, 6
icon in GNOME, 101
NFS and, 266–267, 272
opening in Nautilus, 112
of root user (root), 144
Samba configuration for, 343–344
shell configuration scripts in, 164–165
home directory, 143
host DNS server tool, 464

hostnames
entering manually when installing Red
Hat, 56
local file of hostname to IP address
mappings, 191–192
managing hosts, 261–262
name service resolution order setup,
192–193
obtaining information automatically
via DHCP, 56
setting up, 190
HP JetDirect print queue configuration,
223
ht://Dig
/etc/htdig/htdig.conf file,
575–576
features, 574–575
for files that are not straight HTML,
576
further information, 578
fuzzy searching algorithms, 574–575
installed by default, 574
rundig script for, 575–576, 578
rundig.cron script for, 578
start_url statement, 575–576
testing, 576–578
troubleshooting, 900
updating search databases, 578
HTTP Configuration Tool (Apache), 539
HTTP protocol
firewall configuration for, 57
HTTP acceleration feature, 406
overview, 436
HTTP servers. *See* Apache Web server;
Web servers
HTTPS protocol, 57
hubs versus switches, 420
hwclock command, 697–698

I

Ibiblio.org site, 770–771
IBM
JFS (journaling file system), 147
MCA bus support, 31
Token Ring system, 16

- IEEE 1394 (FireWire), 649, 781
- if statements (Bash), 920–922
- ifconfig command
 - assigning multiple addresses to single NIC, 238
 - loopback device configuration, 234
 - NIC configuration, 234–235
 - viewing current network configuration, 233–234
- IMAP4 (Internet Message Access Protocol version 4)
 - cyrus-imapd startup error, 900–901
 - file system quotas for mailboxes and, 472
 - overview, 471
 - POP3 versus, 472, 485
 - server setup, 485–486
 - SquirrelMail with, 563, 564–565, 566
- indeterminate loops (Bash), 923–924
- Industry Standard Architecture (ISA) bus, 31
- InfiniBand support, 660
- infrared devices, 181
- init scripts
 - default /etc/inittab file, 197–198
 - defined, 196
 - managing by hand, 198–200
 - managing using chkconfig, 200–202
 - runlevels and, 196–197
 - starting network services from, 194–195
- initialization scripts
 - for NFS server, 274, 285–286
 - for Squid, 409
- initrd directory, 143, 144
- install program, 777–778
- installation verification. *See* rpmquery command
- installing. *See also* kickstart installation; RPM (Red Hat Package Manager); standard installation
 - AMANDA, 796
 - BIND, 446–447
 - dhcpd program, 248–249
 - KDE, 120
 - kernel RPM, 619–620
 - kernel SRPM, 621–623
 - Kickstart Configurator, 72–73
 - mt-st package, 785
 - new kernel, 669–670
 - packages built from RPMs, 763
 - RPMs, 756–757
 - Samba, 340–341
 - software from SRPMs, 771–778
 - Squid Web proxy, 407
- Intelligent Input/Output (I2O), 649
- Internet connection
 - IP masquerading configuration, 263
 - modem connection, 257–259, 893–895
 - NFS and security of, 270
 - NSFv4 and, 268, 269
 - sharing, security and, 14
- Internet Message Access Protocol version 4. *See* IMAP4
- Internet Printing Protocol (IPP), 220
- Internet protocols, 429. *See also specific protocols*
- Internet resources
 - Apache configuration information, 520
 - Apache history, 512
 - Apache MPM information, 517
 - Axel Thimm's RPMs (ATrpms) site, 768–769
 - BIND (latest release), 446
 - CAs (certificate authorities), 554
 - database server optimization tips, 425
 - email security information, 487
 - Fedora Extras site, 769, 780
 - FedoraProject.org site, 769–770
 - FHS standard information, 141
 - freshrpms.net site, 768
 - further information, 492
 - Hardware Compatibility List, 893
 - hardware support database, 20, 34
 - ht://Dig project site, 578
 - Ibiblio.org site, 770–771
 - installation manual, 29
 - kernel mirror sites, 624
 - kernel source code, 623
 - kernel update information, 620
 - LCAP site, 731
 - LDAP information, 852
 - Linux Journal site, 573
 - Linux Performance Tuning site, 422
 - Mailman project site, 556

Internet resources (*continued*)

- NcFTPd site, 493
- NFS information, 269, 422, 896
- NIS configuration information, 315
- NIST site, 395
- NTP information, 396, 406
- NTP pool time servers, 397
- OpenLDAP FAQ, 583
- OpenSLP, 319
- PPP configuration information, 894
- print driver information, 225
- ProFTPD site, 492
- RDF information, 568
- Red Hat Network site, 605, 608–614
- Red Hat site, 608
- Red Hat/Fedora Repository site, 768
- RFC information, 246
- RPM PBone Search site, 768
- rpmfind.net site, 769
- rpm.livna.org site, 769
- RPMs, third-party sites for, 768–771
- RSS information, 568, 573
- RSS tools, 573
- Samba, 340
- security mailing list, 23
- SELinux information, 58, 845–846
- Sendmail performance tips, 588
- sound card configuration information, 886
- SquirrelMail project site, 567
- SSL information, 546
- troubleshooting help, 879
- VNC information, 382–383
- vsftpd information, 492
- “X Window System Network Performance,” 416
- X.500 standard information, 852
- XFree86, 903

Internet services. *See also* optimizing

- Internet services; *specific services*
- centralizing, 581
- defined, 429
- disabling unnecessary services, 581
- iptables firewall package for, 441
- less secure services, 434–436
- secure services, 430–433
- server protocols, 436–437

- standalone services, 440–441
- TCP/IP as basis for, 429
- xinetd server configuration, 437–439
- xinetd-started services, 439–440

Internetwork Packet Exchange protocol, 348

InterNIC domain database, 232, 238

interrupt requests (IRQs), 152–153, 154

I/O activity monitoring, 822–826

I/O redirection (Bash), 929–932

I/O scheduling algorithms, 647

iostat command

- checking overall task load, 825–826
- default output, 822
- described, 805
- getting current information, 822–823
- interpreting output from, 823–824
- per-disk statistics, 824–825
- per-partition statistics, 824
- syntax, 822
- sysstat package required for, 822

IP addresses

- assigned to network interfaces, 238
- device aliases, 260
- entering manually when installing Red Hat, 56
- finding for specific domain name, 464
- IP forwarding, 246–247
- IP masquerading, 263, 851
- local file of hostname to IP address mappings, 191–192
- for loopback device, 233–234
- multiple addresses with one NIC, 238, 260
- network classes and, 231–233, 234–235
- Network Layer addresses, 230, 232
- for NIC configuration, 234–235
- obtaining information automatically via DHCP, 56
- reserved network numbers, 235
- reverse lookups, 465–466
- setting up, 189–190
- static, aliases for NICs and, 238
- static, configuring, 260
- subnetting, 238–247
- unusable addresses, 241

- IP forwarding
 - for NICs, 246–247
 - for Squid, 408–409
- IP masquerading, 263, 851
- IPP (Internet Printing Protocol), 220
- iptables firewall. *See* Netfilter firewall
- ipxutils package, 348
- IRQs (interrupt requests), 152–153, 154
- ISA (Industry Standard Architecture)
 - bus, 31
- I2C support, 654–655
- I2O (Intelligent Input/Output), 649

J

- JetDirect (HP) print queue
 - configuration, 223
- JFS (journaling file system) of IBM, 147
- job control with Bash scripts, 936–941
- joystick support, 653

K

- `kadmin.local` command (Kerberos), 869
- `kdb5_util` command (Kerberos), 868
- KDC (Key Distribution Center) system, 865
- KDE (K Desktop Environment)
 - Applications menu, 122–124
 - checking if installed, 120
 - installing, 120
 - Kickstart Configurator setting, 87
 - Konqueror File Manager, 124–125, 432–433
 - logging out, 123, 126
 - managing applets, 121–122
 - overview, 119–121
 - panel, 120–121
 - performance costs, 416
 - running on Windows via VNC, 391–392
 - selecting at login, 98
 - switching programs in, 121
 - switching workspaces in, 121
 - VNC `xstartup` file and, 387–388
- `kdestroy` utility (Kerberos), 866, 870
- Kerberos authentication service
 - `authconfig` applications and, 872–873
 - core utilities, 866–867

- development of, 864
- enabling clients and applications, 870–871
- KDC system, 865
- Kickstart Configurator setting, 85
- for login authentication, 871–873
- machine roles, 865
- overview, 864–865
- packages, 865–866
- realms, 865
- reliability, 865
- server configuration, 867–870
- starting Kerberos-related services, 870
- terminology, 865
- timestamps, 864
 - as trusted third-party service, 865
- kernel. *See also* customizing the kernel;
 - kernel configuration options
 - boot process and, 129, 132–133
 - changes in latest version, 615
 - compiling, 666–667
 - configuration file for packet-filtering services, 180
 - configuring, 629–637
 - deciding to upgrade or not, 616–617
 - defined, 615
 - displaying memory usage, 810–812
 - enabling IP forwarding, 408–409
 - installing kernel RPM, 619–620
 - installing new kernel, 669–670
 - NCP support, 339, 349
 - obtaining the kernel source, 620–628
 - overview, 615
 - passing parameters in GRUB, 130–131
 - patches, 616, 623, 627–628
 - preparing to upgrade, 618–619
 - pristine RPM for, 623–628
 - Red Hat-blessed, support and, 617
 - selection by GRUB, 129, 130
 - size considerations, 667–668
 - SMB support, 339
 - source code site, 623
 - SRPM (source RPM), 621–623
 - startup log, 879–881
 - updating GRUB for new kernel, 670
 - upgrading versus customizing, 618
 - `up2date` utility and, 181

kernel (*continued*)

- verifying and unpacking the archive, 626–627
- verifying Squid configuration, 408–409
- kernel configuration options
 - ATA/ATAPI/MFM/RLL support section, 647–648
 - Auditing support, 639
 - Automatic kernel module loading, 640
 - Block devices options, 646–647
 - BSD Process Accounting, 638–639
 - Bus options section, 643–644
 - CD-ROM/DVD file systems support, 661
 - Character devices section, 654
 - Code maturity level options section, 637
 - Configure standard kernel features, 639
 - Cryptography options section, 664–665
 - Device Drivers section, 645–660
 - DOS/FAT/NT File systems, 661
 - Enable loadable module support, 640
 - Executable file formats section, 644
 - File systems configuration section, 660–663
 - Forced module unloading, 640
 - Fusion MPT device support, 649
 - General setup options section, 637–639
 - Generic Driver Options section, 645
 - Graphics support section, 655–656
 - help for, 637
 - IEEE 1394/FireWire support, 649
 - InfiniBand support, 660
 - Input Device Drivers section, 653–654
 - ISDN support, 653
 - I2C support, 654–655
 - I2O support, 649
 - Kernel .config support, 639
 - Kernel hacking section, 664
 - Library routines section, 665
 - Linux telephony support, 653
 - Loadable module support options section, 640
 - Local version - append to kernel release, 637–638
 - Memory Technology Device (MTD) support, 645

- MMC/SD card support, 660
- Module unloading, 640
- Module versioning support (EXPERIMENTAL), 640
- Multi-device support section, 649
- Multimedia devices section, 655
- Native Language Support, 663
- network file systems, 662
- Networking support section, 649–653
- Old CD-ROM drivers section, 648
- Parallel port support, 645–646
- PC-style hardware, 645–646
- Plug and Play support, 646
- Power management options section, 643
- Processor type and features options section, 640–642
- Profiling support section, 663–664
- Pseudo-file-systems, 662
- saving the configuration, 665–666
- SCSI device support section, 648
- Security options section, 664
- Sound section, 656
- Source checksum, 640
- Support for paging of anonymous memory (swap), 638
- Sysctl support, 639
- USB support section, 656–659
- Key Distribution Center (KDC) system, 865
- keyboard
 - configuration during standard installation, 39
 - configuration file for, 181–182
 - kernel configuration options, 653
 - Kickstart Configurator settings, 74
 - support for, 33
- Kickstart Configurator (Fedora Core)
 - Authentication screen, 83–85
 - Basic Configuration screen, 73–75
 - Boot Loader Options screen, 76–77
 - Display Configuration screen, General tab, 87–88
 - Display Configuration screen, Monitor tab, 89
 - Display Configuration screen, Video Card tab, 88–89

- Firewall Configuration screen, 85–87
 - Installation Method screen, 75–76
 - installing, 72–73
 - Network Configuration screen, 82–83
 - overview, 71
 - Package Selection screen, 90–91
 - Partition Information screen, 77–82
 - Post-Installation Script screen, 92
 - Pre-Installation Script screen, 91–92
 - saving the `ks.cfg` file, 93
 - starting, 73
 - kickstart installation. *See also* Kickstart Configurator (Fedora Core); standard installation
 - choosing configuration file location, 93
 - creating a bootable CD-ROM, 94
 - creating a bootable floppy, 93–94
 - creating `ks.cfg` file for, 71–93
 - making installation tree available, 93
 - manual online, 29
 - starting the installation, 95–96
 - `kill` command, 680–681
 - `killall` command, 680, 682
 - killing processes
 - all processes owned by user, 936–938
 - all processes specified by name, 680, 682
 - `kill` command for, 680–681
 - matching given pattern, 680, 681
 - running process on VNC server, 392
 - `top` command for, 821
 - `kinit` utility (Kerberos), 866, 870
 - `klist` utility (Kerberos), 866, 871
 - `kmod` kernel module autoloader, 640
 - Konqueror File Manager (KDE), 124–125, 432–433
 - `krbafs` package (Kerberos), 866
 - `krbafs-devel` package (Kerberos), 866
 - `krbafs-utils` package (Kerberos), 866
 - `krb5-auth-dialog` package (Kerberos), 866
 - `krb5-devel` package (Kerberos), 866
 - `krb5-libs` package (Kerberos), 866
 - `krb5-server` package (Kerberos), 865
 - `krb5-workstation` package (Kerberos), 865
 - `krb-telnet` Kerberos application, 870–871
 - `ks.cfg` file, 93, 95–96. *See also* Kickstart Configurator (Fedora Core)
 - kudzu, 179, 182
 - `kujob.sh` Bash script, 936–938
- ## L
- LAMP (Linux, Apache, MySQL, and PHP), 353–354
 - languages
 - additional language support, 58–59, 75
 - choosing for Red Hat installation, 38
 - configuration file for default, 179
 - Kickstart Configurator settings, 74, 75
 - Native Language Support, 663
 - selecting from login window, 98
 - LANs (local area networks). *See* TCP/IP networking
 - laptop video problems, troubleshooting, 901–902
 - `last` command, 717
 - LCAP (Linux Kernel Capability Bounding Set Editor), 731
 - LDAP (Lightweight Directory Access Protocol)
 - `attributetype` definition, 854
 - client-server architecture, 852
 - configuration files, 857–859
 - core OpenLDAP server files, daemons, and utilities, 856–857
 - creating LDIF files, 860–861
 - directory defined for, 851
 - directory organization, 852–855
 - distinguished names (DNs), 852–853
 - example schema entries, 853–854
 - further information, 852
 - Kickstart Configurator setting, 85
 - LDIF (LDAP Data Interchange Format), 855
 - Object Identifiers (OIDs), 853
 - `objectclass` definition, 854
 - OpenLDAP FAQ, 583
 - OpenLDAP packages for Linux, 855–856
 - OpenLDAP server configuration, 857–859

LDAP (*continued*)

- optimizing services, 582–583
- OSI layer model not used by, 852
- overview, 851–852
- populating the server with data, 859
- relative distinguished names (RDNs), 852–853
- starting the OpenLDAP server, 859
- system authentication using, 860–864
- updating client systems to use
 - authentication, 861–864
- uses for directories, 859–860
- X.500 standard for, 852

LDAP Data Interchange Format (LDIF), 855, 860–861

ldapadd utility (LDAP), 856

ldap.conf file (LDAP), 857

ldapdelete utility (LDAP), 856

ldapmodify utility (LDAP), 856

ldapsearch utility (LDAP), 856

LDAs (local delivery agents), 469

ldconfig program, 174

LDIF (LDAP Data Interchange Format), 855, 860–861

ld.so.conf file, 174

leader MPM (Apache 2), 517

“Learning NFSv4 with Fedora Core 2” (Emery), 269

Lerner, Reuven (“At the Forge—Syndication with RSS”), 573

lib directory, 143

libdbi-dbd-mysql package, 355

Lightweight Directory Access Protocol. *See* LDAP

Linux, Apache, MySQL, and PHP (LAMP), 353–354

Linux Journal Web site, 573

Linux Kernel Archives, 623–626

Linux Kernel Capability Bounding Set Editor (LCAP), 731

Linux Performance Tuning Web site, 422

listing. *See* displaying or viewing

LKM (loadable kernel module), 633

local area networks (LANs). *See* TCP/IP networking

local delivery agents (LDAs), 469

Locally-Connected print queue, 219, 220

locking passwords for user accounts, 713

log files and logging

- Apache logging support, 515
- of attempted attacks, 27
- for authentication privilege messages, 172–173
- BIND performance and, 586–587
- BIND server logging, 451, 455–457
- for cron messages, 173
- finding the use of, 882
- keeping hard copies, 27
- kernel startup log, 879–881
- LDAP performance and, 582
- for mail messages, 173
- NTP-related entries in system log, 405
- Postfix mail server, 482
- by sar, 826
- security log, 27, 28
- SELinux messages in system log, 840–841
- Sudo log file, 733
- syslog.conf configuration file for, 172–174
- system log, 405, 733, 840–841, 879
- tracking system changes, 761
- as troubleshooting resources, 879–882
- as variable files, 142
- viewing, 881–882
- vsftpd transfer log, 499
- xinetd log capabilities, 437

Log Viewer program, 185–186, 881–882

logging in

- changing user login shell, 714
- /etc/passwd file shell field and, 709
- graphical login window, 97–99
- Kerberos authentication for, 871–873
- messaging users at login, 168
- to Novell server, 350
- prelog banner, 168
- to Red Hat Network Web site, 608
- as root user, 5
- to Samba client, 345–346
- selecting KDE for session, 120
- to SquirrelMail, 563–564
- Telnet prelog banner, 168–169

- troubleshooting, 883–884
- viewing login information, 717–718
- from Windows PC to Samba, 347–348
- logging out
 - of GNOME, 119
 - of KDE, 123, 125
 - from Samba client, 345–346
- logical partitions, 50, 80
- logical user accounts, 707–708
- Logical Volume Manager (LVM), 157–160
- logical volumes, 159, 272
- logrotate.conf file, 174–175
- loopback device, 233–234
- loops (Bash)
 - defined, 919
 - for statement, 920, 922–923
 - until statement, 920, 923–924
 - while statement, 920, 923–924
- lsattr command, 691, 692
- LSI Logic Fusion card, 649
- lsdf command, 691, 694–695
- lsraid command, 162
- LVM (Logical Volume Manager), 157–160
- lvm command
 - lvm help command, 158–159
 - lvm lvdisplay command, 159
 - lvm pvdisplay command, 159
 - lvm vgdisplay command, 159–160
 - syntax, 157

M

- MAC (Media Access Control) addresses, 230
- mail delivery agents (MDAs), 469
- mail host, running Postfix on, 483–484
- mail notifiers (biffs), 470
- mail servers. *See* email
- mail spool or mailbox file, 469
- mail transfer agents (MTAs), 469. *See also* Postfix mail server; Sendmail program
- Mail Transport Agent Switcher, 480, 481
- mail user agents (MUAs), 468–469
- mailbox file or mail spool, 469

- mailing lists. *See also* Mailman mailing list manager
 - defined, 555
 - for Red Hat Linux security, 23
- Mailman mailing list manager
 - adding multiple list members, 562
 - administrative tasks, 561–563
 - aliases files, 558
 - built-in functionality, 556
 - checking if installed, 556
 - configuration categories, 560–561
 - configuring, 556–558
 - creating a mailing list, 559
 - further information, 556
 - hiding a mailing list, 562
 - mailing list configuration, 560–561
 - passwords, 557
 - popularity of, 555–556
 - restricting archives access, 563
 - site moderator, 557
 - starting at boot time, 558
 - starting the service, 558
 - URL for administration page, 562
 - Web site, 556
- make allnoconfig command, 630
- make command, 666–667, 775–776
- make config command, 629
- make defconfig command, 631
- make gconfig command, 629
- make install command, 777–778
- make menuconfig command, 629, 634–637
- make oldconfig command, 629, 630, 632
- Make RAID Device dialog box, 81–82
- make xconfig command, 629, 633–634, 665–666
- MBR (Master Boot Record), 53, 77, 128
- MCA (Micro Channel Architecture)
 - bus, 31
- md devices, 161, 162. *See also* RAID (Redundant Array of Independent Disks)
- MDAs (mail delivery agents), 469
- MD5 encryption, 84
- Media Access Control (MAC) addresses, 230

- media directory, 144
- memory. *See* RAM (random access memory)
- Memory Technology Device (MTD)
 - support, 645
- merely mortal user, 731. *See also* Sudo (superuser do)
- metacharacters (Bash), 906–909
- metadevices. *See also specific metadevices*
 - defined, 156
 - disk failure and, 156–157
 - logical volumes, 157–160
 - RAID, 80–82, 156, 160–162
- m4 macro processor (Sendmail), 475–476
- Micro Channel Architecture (MCA)
 - bus, 31
- Microsoft network connection. *See* Samba or SMB (Server Message Block)
- Microsoft Windows
 - accessing file systems, 890
 - file systems support, 661
 - running KDE via VNC, 391–392
 - using Linux VNC server from, 390–391
- Mills, David (NTP implementation programmer), 396
- MIME (Multipurpose Internet Mail Extensions), 470–471, 520
- minus sign (–), deleting files starting with, 889
- mixed networks, 18, 19–20
- mke2fs command, 683, 685–686
- mkisofs command, 787
- mkswap command, 683, 686–687
- MMC card support, 660
- mnt directory, 144
- mod_auth_mysql package, 354
- mod_authz_ldap package, 856
- modem connection, 257–259, 893–895
- Monitor dialog box, 207–208
- monitoring performance. *See* performance monitoring
- monitors
 - changing type, 207–208
 - dual monitor configuration, 209–210
 - Kickstart Configurator settings, 89
 - X Window system and, 31–32
- mount command, 304, 683, 688–690
- mount failed message, 898
- mount point
 - defined, 46
 - Kickstart Configurator setting, 79
 - manual partitioning with Disk Druid and, 50
 - for RAID partitions, 81
- mounting
 - autofs automount service for NFS, 301–304
 - CD-ROM drives, troubleshooting, 885
 - file system during standard installation, 46
 - file systems exported with NFS, 266
 - file systems with mount, 683, 688–690
 - fstab configuration file for, 169–170
 - NFS exports, troubleshooting, 897–898
 - NFS exports with autofs, 301–304
 - NFS-specific mount options, 296–297
 - NFSv4-specific mount options, 299
 - Novell file system, 350
 - OS partitions/slices, 156
 - partitions, troubleshooting, 891
 - Samba client file system, 346–347
- mouse
 - configuration file for, 182–183
 - kernel configuration options, 653–654
 - Kickstart Configurator settings, 74
 - support for, 33
- Mozilla Firefox
 - root certificates with, 548
 - testing Squid with, 412–413
 - X performance and, 417
- MPMs (multiprocessing modules), 516–517
- MS-DOS, CD-ROM access using, 35
- msqld server daemon, starting, 355
- mt command, 786–787
- MTAs (mail transfer agents), 469. *See also* Postfix mail server; Sendmail program
- MTD (Memory Technology Device)
 - support, 645
- mt-st package, 784–787
- MUAs (mail user agents), 468–469

- multihomed servers (virtual hosts), 514, 523, 537–539
- multiprocessing modules (MPMs), 516–517
- Multipurpose Internet Mail Extensions (MIME), 470–471, 520
- mutt email client, 469, 766
- \$mynetworks configuration variable, 484
- MySQL
 - anonymous user, 355
 - checking if installed, 354
 - client programs, 359–361
 - deleting anonymous accounts, 358
 - mysql command, 356–357, 361
 - mysqladmin command, 355–356, 359–360
 - mysqlshow command, 355, 356, 360
 - packages, 354–355
 - as part of LAMP, 353–354
 - popularity of, 352, 353
 - PostgreSQL versus, 362–364
 - restoring installation to original state, 358
 - root user, 355
 - securing the installation, 355–358
 - server optimization tips, 425
 - setting the anonymous user password, 357–358
 - setting the root user password, 356–357
 - starting the msqld server daemon, 355
 - third-party Apache modules supporting, 353–354
 - verifying if running, 355–356
 - mysql command, 356–357, 361
 - mysqladmin command, 355–356, 359–360
 - mysql-bench package, 355
 - mysqlshow command, 355, 356, 360
- N**
- Name Switch Cache service, 85
- named daemon, 183. *See also* /etc/named.conf file
- names and naming
 - fully qualified domain name (FQDN), 311
 - hard disk and floppy drive names, 45
 - partition names, 46
 - print queue, 219, 226
 - for shell configuration scripts, 165
- NAS (network-attached storage), 420
- NAT (Network Address Translation), 263, 851
- National Institute of Standards and Technology (NIST), 395
- National Security Agency. *See* NSA
- Nautilus File Manager (GNOME)
 - accessing FTP sites, 113
 - bookmarks, 113–114
 - changing background and icon emblems, 117
 - displaying folder contents, 112
 - location bar, 110
 - menu bar, 110
 - opening files, 112–113
 - opening from GNOME, 99, 110
 - opening home folder from, 112
 - preferences, 115–116
 - resize handle, 111
 - showing and hiding items, 118
 - status bar, 111
 - toolbar, 110
 - window panes, 111
- NcFTPd, 492, 493
- NCP. *See* Novell
- ncpfs package, 348
- ncpmount command, 349–350
- netdump service, 183
- Netfilter (iptables) firewall. *See also* firewalls
 - activating changes, 851
 - enabling support for, 408
 - enabling the firewall, 848
 - /etc/sysconfig/iptables configuration file, 180, 195
 - finding ports and protocols used by services, 849
 - for Internet services, 441
 - ipchains firewall rules and, 195

- Netfilter (iptables) firewall (*continued*)
 - iptables package for, 441
 - kernel configuration options, 650–651
 - least-access approach for, 849
 - modifying rules for Squid, 411–412
 - Security Level Configuration tool for, 848–850
 - specification port access, 850–851
 - trusted devices, 849, 850
 - Trusted services list, 848–849, 850
- Network Address Translation (NAT), 263, 851
- network classes
 - CIDR and, 244–245
 - IP numbers, 239–240
 - overview, 231–233
 - reserved network numbers, 235
 - subnets and subnet masks, 242–244
- network configuration files
 - for DNS setup, 190–191
 - in `/etc/sysconfig` subdirectories, 187–189
 - `/etc/sysconfig/network` file, 184, 190
 - in `/etc/sysconfig/network-scripts` directory, 196
 - for hostname setup, 190
 - for IP address setup, 189
 - list of, 892
 - for local file of hostname to IP address mappings, 191–192
 - for name service resolution order setup, 192–193
 - for Netfilter (iptables) firewall, 195
- Network Configuration tool for, 189
- for setting up or moving the system, 188–189
- for starting services, 193–195
- for static routes, 195
- troubleshooting, 892
- Network Configuration tool
 - changing NIC configuration, 259–260
 - described, 189
 - Ethernet device configuration, 252–254
 - IP masquerading configuration, 263
 - managing DNS settings, 261
 - managing hosts, 261–262
 - modem connection configuration, 257–259
 - opening, 251
 - profiles for multiple configuration sets, 262
 - removing a NIC, 259
 - tabbed pages, 251–252
 - wireless NIC configuration, 254–256
- Network File System. *See* NFS
- Network Information Service Plus (NIS+), 334, 423, 590
- Network icon (GNOME), 100
- Network Information Service. *See* NIS
- network interface cards. *See* NICs
- Network Layer addresses, 230, 232
- network mask, 235, 240–241, 246–247.
 - See also* subnetting
- Network Plus* (Groth), 235
- network security. *See also* authentication; firewalls; passwords
 - Kerberos authentication for, 864–873
 - LDAP for, 851–864
 - least-access approach, 849
 - Netfilter configuration, 847–851
- Network Time Protocol. *See* NTP
- network-attached storage (NAS), 420
- networked print queue types
 - CUPS (IPP), 220
 - JetDirect, 223
 - Novell (NCP), 222–223
 - UNIX (LPD), 220–221
 - Windows (SMB), 221–222
- `newaliases` command, 481, 558
- `newgrp` command, 719, 721, 723
- `newusers` utility, 715–716
- NFS (Network File System)
 - autofs automount service for, 301–304
 - for centralized storage of home directories, 266–267
 - client configuration, 296–301
 - client-server architecture, 266–267
 - design rules, 271–273
 - distributing exports across multiple disks, 272, 419
 - exported directory won't mount, 898
 - exporting a file system, 266
 - failure messages, 896–897

- file locking, 307
- files on mounted file system not visible, 897–898
- further information, 269, 422, 896
- hot spots, 419
- isolating NFS traffic on network segment, 421
- kickstart installation from NFS server, 76
- minimizing write-intensive exports, 420–421
- network traffic and, 419
- NFS export, 266
- NFS mount, 266
- NFSv4 and RPCSEC_GSS
 - configuration files, 279–280
- NFSv4 features and enhancements, 268–269
- NIS with, 267, 334–337, 423
- overview, 265–271
- performance and, 270, 273
- performance tuning, 418–422
- planning an installation, 271–273
- ports and network protocols, 305–306
- pros and cons, 269–271
- reboots and, 268–269
- root squashing, 271, 277–278, 306–307, 730
- Samba versus, 348
- security, 305–307
- security issues, 270–271
- server configuration, 273–295
- troubleshooting, 896–898
- NFS client
 - configuring a client system, 296–297
 - configuring an NFSv4 client, 299–300
 - example configuration, 300–301
 - file locking, 307
 - NFS-specific mount options, 296–297
 - NFSv4-specific mount options, 299
 - security issues, 305–307
- NFS server
 - access control files, 281–283
 - CIDR with, 275
 - commands, 274, 286–289
 - configuration files, 273, 274–280
 - daemons, 273–274, 283–285
 - dedicated server for, 420
 - displaying statistics for, 421–422
 - /etc/exports file, 273, 274–279
 - example configuration, 290–292
 - exportfs command, 285, 287–288
 - file-locking daemons, 286
 - hosts.allow and hosts.deny files, 281–283
 - initialization scripts, 274, 285–286
 - kickstart installation from, 76
 - NFS Server Configuration tool, 292–295
 - nfs startup script, 285–286
 - nfsstat command, 421–422
 - portmap script, 285
 - rpcinfo command, 286–287
 - RPCSEC_GSS configuration files and, 279–280
 - secure NFS, 290, 307
 - security issues, 306–307
 - showmount command, 288–289
 - start order for daemons, 284
 - starting manually, 286
 - status information files, 273, 280–281
- NFS Server Configuration tool
 - adding a new export, 292–293
 - Basic tab, 292–293
 - General Options tab, 293–294
 - modifying mount options for exports, 293–294
 - overview, 292
 - remapping UIDs or GIDs, 294
 - removing an NFS share, 295
 - root-squashing options, 294
 - User Access tab, 294–295
- nfs startup script, 285–286
- NFS-HOWTO Web page, 422
- nfsstat command, 421–422
- nice command, 418, 680
- NICs (network interface cards)
 - aliases and DHCP, 238
 - checking if configured, 234
 - configuring during standard installation, 55–56
 - configuring for internal network, 234–237
 - configuring single NIC with ifconfig, 234–235

NICs (*continued*)

- disabling on boot, 55
- Ethernet device configuration, 252–254
- IP forwarding, 246–247
- kernel configuration options, 651–652
- Kickstart Configurator settings, 83
- loopback interface configuration, 233–234
- multiple IP addresses for single NIC, 238
- NFS performance and, 420
- replacing older Ethernet cards, 20
- support for, 20, 34
- using two Ethernet cards, 896
- wireless NIC configuration, 254–256

NIS (Network Information Service)

- client configuration, 326–334
- client programs, 314
- client-server architecture, 310
- development of, 309
- domains, 310–311
- further information, 315
- Kickstart Configurator setting, 84
- master versus slave servers, 310
- multiple-domain configurations, 311, 312–314
- NFS with, 267, 334–337, 423
- overview, 309–311
- packages, 311
- performance tuning, 422
- planning an installation, 311–314
- recursive netgroups, 423
- server configuration, 315–326
- for sharing authentication information, 310
- single-domain configurations, 311–312, 313
- vsftpd optimization and, 590

NIS clients

- client programs, 314
- commands, 331–333
- configuring startup files, 331
- confirming daemon registration, 331
- editing `/etc/nsswitch.conf`, 330–331
- example configuration, 333–334
- setting the domain name, 326

- starting `ypbind`, 330, 331
- steps for configuring, 326
- testing client-server communication, 330
- `ypbind` configuration, 326–330
- `ypcat` command, 331
- `ypmatch` command, 331, 332–333
- `yppasswd` command, 331, 333
- `ypwhich` command, 331, 332

NIS+ (Network Information Service Plus), 334, 423, 590

NIS servers

- checking if portmapper is running, 319
- checking if server is running, 319
- configuration commands and files, 315–316
- creating the NIS databases, 319–320
- `domainname` command, 316
- `/etc/ypserv.conf` configuration file, 315, 316, 317–318
- example configuration, 324–326
- master versus slave servers, 310
- `nisdomainname` command, 315, 316
- `portmap` daemon configuration, 318
- `rpcinfo` command, 319
- setting the domain name, 316
- slave server configuration, 320–321
- starting servers at boot time, 322–323
- starting the portmapper, 319
- starting `yppasswdd`, 321
- starting `ypxfrd`, 321–322
- steps for configuring, 315
- `/var/yp/securenets` configuration file, 315, 316–317
- `ypinit` command, 315, 319–321
- `ypserv` configuration, 316–318
- `ypwhich -m` command, 219–220
- `nisdomainname` command, 315, 316, 326

NIST (National Institute of Standards and Technology), 395

nmbd daemon, 186

no such file or directory message, 889–890

Nottingham, Mark (“RSS Tutorial for Content Publishers and Webmasters”), 573

- Novell
 - kernel support for NCP, 339, 349
 - network connection, 339, 348–350
 - networked print queue type, 222–223
 - NSA (National Security Agency)
 - as SELinux developer, 835, 836, 837
 - SELinux-related Web sites, 58, 845
 - NSCD daemon, 85
 - nss_ldap package, 856
 - NT file system support, 661
 - ntalk real-time chat protocol, 435–436
 - NTFS file system, 147
 - NTP (Network Time Protocol). *See also* time server
 - autokey encryption, 399–401, 403–404
 - daemon, 396
 - defined, 395, 702
 - as distributed application, 397
 - further information, 396, 406
 - installing software, 397
 - joining public access server pool, 406
 - Kerberos timestamps and, 864
 - ntpd configuration file, 397–398
 - pool servers, 397
 - precision of, 395
 - primary servers, 397
 - secondary servers, 397
 - syncing system clock with time server, 702
 - utilities, 396
 - ntpd command, 697, 702
 - ntpd daemon, 396
 - ntpstat command, 404
- O**
- Open Shortest Path First (OSPF)
 - protocol, 233
 - opening. *See* starting or opening
 - OpenLDAP. *See also* LDAP (Lightweight Directory Access Protocol)
 - core server files, daemons, and utilities, 856–857
 - FAQ for, 583
 - packages, 855–856
 - populating the server with data, 859
 - server configuration, 857–859
 - starting the server, 859
 - system authentication using, 860–864
 - openldap package, 855
 - openldap-clients package, 855
 - openldap-devel package, 856
 - openldap-servers package, 855
 - openldap-servers-sql package, 856
 - OpenPGP, 626
 - OpenSLP, 319
 - operating systems. *See also specific operating systems*
 - mounting OS partitions/slices, 156
 - selection using GRUB, 129, 130
 - operators (Bash)
 - arithmetic, 916–917
 - comparison, 913–916
 - file test, 917–919
 - for sets, 906–907
 - OProfile, 663–664
 - opt directory, 144
 - optimizing Internet services
 - centralizing services, 581
 - disabling unnecessary services, 581
 - DNS services, 583–587
 - FTP services, 590
 - LDAP services, 582–583
 - mail services, 587–590
 - overall performance and, 581–582
 - overview, 581–582
 - Web services, 590–593
 - optimizing network services
 - database servers, 424–425
 - NFS, 418–422
 - NIS, 422
 - Samba, 423–424
 - X Window system, 416–418
 - Oracle, 424, 425
 - Orchard, Leslie (*Hacking RSS and Atom*), 573
 - OSPF (Open Shortest Path First)
 - protocol, 233
 - OSTYPE macro (Sendmail), 475–476
 - outsourcing Web and FTP hosting, 20–21

P

- Package Management tool, 72–73, 120
- packages. *See also* RPM (Red Hat Package Manager); `rpmquery` command; `up2date` agent
 - AMANDA, 795
 - building in stages, 763–764
 - building using SRPMs, 761–764
 - built from RPMs, installing, 763
 - `cdrecord`, 787–789
 - defined, 62
 - `dump`, 789–790
 - Kerberos, 865–866
 - Kickstart Configurator settings, 90–91
 - `mt-st`, 784–787
 - MySQL, 354–355
 - for Netfilter (iptables) firewall, 441
 - NIS, 311
 - for Novell network connection, 348
 - NTP, 397
 - OpenLDAP, 855–856
 - PostgreSQL, 365–366
 - querying dependencies, 750–751
 - querying uninstalled packages, 753
 - Red Hat Network Configuration dialog box options, 600–602
 - Samba, 340
 - security issues, 62
 - selecting during standard installation, 62–64
 - Sendmail, 473, 475, 476
 - `sysstat`, 822, 826
 - VNC, 383
- Packard, Keith (“X Window System Network Performance”), 416
- packet filtering
 - configuration file for services, 180
 - kernel configuration options, 650–651
 - by Netfilter (iptables) firewall, 441
 - NFS security improved by, 305–306
 - SMTP and, 489
- packet sniffers, 487
- packets (TCP/IP), 230–231
- paging, 47
- `pam_krb5` package, 866
- panel (GNOME)
 - Actions menu, 109–110
 - Applications menu, 103–104, 107–108, 109
 - calendar window, 101
 - Desktop menu, 106–107
 - managing applets, 102–103
 - overview, 101–102
 - Places menu, 105
 - RHEL versus Fedora Core, 101
 - speaker icon, 101
 - system status icon, 101
 - Workspace Switcher, 102
- panel (KDE)
 - Applications menu, 122–124
 - managing applets, 121–122
 - overview, 120–121
 - Workspace Switcher, 121
- `parted` command, 683, 684–685
- partitioning hard disks. *See also* Disk Druid
 - `anaconda-ks.cfg` file as guide for, 79
 - automatic partitioning, 43–44
 - commands for, 683–685
 - database server performance and, 424
 - editing partition tables, 683, 684–685
 - foreign disk partition support, 662
 - hard disk and floppy drive names, 45
 - Kickstart Configurator settings, 78–82
 - manually, with Disk Druid, 45–51
 - mounting a file system, 46
 - mounting a partition, 891
 - partition names, 46
 - primary versus logical partitions, 50, 80
 - root partition (/), 47
 - during standard installation, 42–51
 - swap partition, 31, 47, 683, 686–687
 - upgrading an existing installation and, 51
 - viewing current partitioning, 155
 - on x86 machines, 156
- partitioning NIS domains, 423
- `passwd` command, 708, 713–714

- passwords. *See also* `/etc/passwd` file; `/etc/shadow` file
- changing for group accounts, 721
- changing for multiple user accounts, 715, 716–717
- changing from script, 714
- encryption, 717
- entering in login window, 99
- expiration policy for, 714–715
- for GRUB, 52–53, 77
- locking or unlocking, 713
- for mailing list site moderator, 557
- for Mailman mailing lists, 557
- for MySQL anonymous user, 357–358
- for MySQL root user, 356–357
- newgrp and, 721
- overview, 7
- policy for, 22
- removing for user accounts, 713
- resetting for users, 883
- root, lost or forgotten, 884
- root, setting, 61, 75
- Samba, 342–343
- shadow passwords, 84, 709, 710
- shadowed group files and, 723
- updating only expired tokens, 713
- patching the kernel, 616, 623, 627–628.
 - See also* upgrading the kernel
- PCI (Peripheral Component Interconnect) bus, 30–31, 643–644
- PCI-X (PCI extended) bus, 643, 644
- PCMCIA configuration file, 184–185
- peer-to-peer networks, 19–20
- perchild MPM (Apache 2), 516
- perfork MPM (Apache 2), 516
- performance. *See also* performance monitoring; speeds
 - Apache advantages for, 512–513
 - database servers, tuning, 424–425
 - disabling unnecessary services for, 581
 - DNS services, optimizing, 583–587
 - FTP services, optimizing, 590
 - LDAP services, optimizing, 582–583
 - mail services, optimizing, 587–590
 - NFS and, 270, 273
 - NFS, tuning, 418–422
 - NIS, tuning, 422
 - RAM versus swap space and, 31, 47
 - running X across a network and, 391
 - Samba networking, tuning, 423–424
 - system administrator duties for, 9–10
 - tmpfs virtual file system and, 149
 - usability and security balanced with, 415
 - Web services, optimizing, 590–593
 - X Window system, tuning, 416–418
- performance monitoring
 - CPU usage, 679–680, 817–821, 829–830
 - I/O activity, 822–826
 - memory usage, 806–812, 827–828
 - overview, 10
 - sar for, 826–830
 - system-performance-monitoring tools, 805–806
 - viewing running tasks, 812–821
- Peripheral Component Interconnect (PCI) bus, 30–31, 643–644
- perl-LDAP package, 856
- pg_ctl utility (PostgreSQL), 372
- \$PGDATA environment variable, 366–367, 376
- \$PGDATABASE environment variable, 376
- pg_dump utility (PostgreSQL), 376, 377–378
- PGP (Pretty Good Privacy), 601
- pgrep command, 676, 678–679
- pg_restore utility (PostgreSQL), 376, 377, 378
- \$PGUSER environment variable, 376
- PHP, 353, 354, 545–546
- php-ldap package, 856
- php-mysql package, 355
- physical volumes, displaying, 159
- pidof command, 676, 679
- pskill command, 680, 681
- Places menu (GNOME), 105
- planning the network
 - client-server versus peer-to-peer, 18–20
 - deciding how the network will be used, 13
 - disaster recovery plan, 23–26
 - documenting the plan, 26–28
 - Internet connection sharing and, 14

- planning the network (*continued*)
 - mixed networks and, 19–20
 - need for, 13
 - questions to ask, 14
 - security and, 21–23
 - system requirements, 20–21
 - topologies, 14, 15–18
 - Web and FTP hosting decisions, 20–21
- Plug and Play support, 646
- POP3 (Post Office Protocol version 3), 431, 471, 472, 485
- portmap daemon, 318
- portmap script, 285
- POSIX capabilities, 731
- Postfix mail server
 - advantages of, 479
 - aliases file for, 169, 481
 - configuring, 480–482
 - /etc/postfix/main.cf file, 480–481, 482–483
 - /etc/postfix/master.cf file, 588–590
 - /etc/postfix/transport file, 588
 - logging messages, 482
 - Mailman with, 558
 - newaliases command, 481
 - performance tuning, 588–590
 - popularity of, 472–473
 - running behind a firewall or gateway, 482–483
 - running on a mail host, 483–484
 - starting, 481
 - switching to, 479–480
 - verifying startup at boot time, 481
- PostgreSQL
 - access privileges, 368–372
 - access rules, 370–372
 - authentication methods, 369
 - checking if installed, 365
 - client programs, 375–378
 - createdb utility, 372–373, 376
 - createuser utility, 373–374, 376
 - creating a test database, 372–374
 - database cluster creation, 367–368
 - deleting a database, 375, 376–377
 - deleting a user, 377
 - dropdb utility, 375, 376–377
 - dropuser utility, 376, 377
 - environment variables, 366–367, 376
 - initializing the installation, 366–368
 - MySQL versus, 362–364
 - packages, 365–366
 - pg_ctl utility, 372
 - pg_dump utility, 376, 377–378
 - pg_restore utility, 376, 377, 378
 - popularity of, 362
 - psql utility, 376
 - Red Hat Database and, 352
 - reloading the access control file, 372
 - restoring a database dump, 378
 - saving database data, 377–378
 - server optimization tips, 425
 - server programs, 378
 - starting the server, 368
 - steps for finalizing the installation, 366
 - testing connectivity to test database, 374–375
 - trust authentication, 368–372
 - /var/lib/pgsql/data/pg_hba.conf file, 368–372
- PostScript, 227
- predefined variables (Bash), 911–912
- preferences
 - log viewer, 882
 - Nautilus File Manager, 115–116
 - screensaver (GNOME), 118–119
- Pretty Good Privacy (PGP), 601
- primary partitions, 50, 80, 156
- print driver, 224–227
- Print Notifier, 227–228
- print queue
 - default type, 219
 - defined, 219
 - editing settings for, 226
 - Locally-Connected type, 219, 220
 - managing print jobs, 227–228
 - naming, 219
 - Networked CUPS (IPP) type, 220
 - Networked JetDirect type, 223
 - Networked Novell (NCP) type, 222–223
 - Networked UNIX (LPD) type, 220–221
 - Networked Windows (SMB) type, 221–222
 - setting type for, 219
- print server, 5

- Printer Configuration tool
 - adding a new printer, 218–219
 - deleting a printer, 227
 - editing the printer configuration, 225–227
 - print driver selection, 224–225
 - print queue configuration, 219–224
 - setting the default printer, 227
 - starting, 217–218
- printers. *See also* print queue
 - adding a new printer, 217–225
 - character devices, 654
 - deleting, 227
 - editing settings for, 225–227
 - managing print jobs, 227–228
 - parallel port support for, 645–646
 - Samba configuration for, 344
 - setting the default printer, 227
- proc directory
 - described, 144
 - hardware information in, 150, 152–154
 - important files and subdirectories, 154
 - md device information, 161
 - software information in, 150–152
 - Squid and, 408
 - subdirectories named after PIDs, 150–152
 - useful files in `/proc/PID`, 152
 - viewing process listings, 150–151
 - as virtual file system, 150
 - `/proc/cpuinfo` file, 153, 154
- processes. *See also specific commands*
 - Bash scripting for, 936–941
 - displaying list of PIDs using a file, 691, 693–694
 - identifying files opened by, 691, 694–695
 - kill command, 680–681
 - killall command, 680, 682
 - killing all owned by user, 936–938
 - managing from command line, 675–682
 - modifying priorities for, 682
 - nice command, 680
 - obtaining information about, 676–680
 - pgrep command, 676, 678–679
 - pidof command, 676, 679
 - pkill command, 680, 681
 - process accounting options, 638–639
 - ps command, 676–678, 813–817
 - pstree command, 676, 679
 - renice command, 680
 - sending signals, 680–682
 - tload command, 676, 680
 - top command, 676, 679–680, 817–821
 - viewing processes users are running, 717–718
- processors (CPUs)
 - Apache performance and, 593
 - customizing the kernel for, 640–642
 - displaying real-time usage with `top`, 679–680, 817–821
 - monitoring usage with `sar`, 829–830
 - `/proc/cpuinfo` file for, 153, 154
 - support for, 30
 - symmetric multiprocessing (SMP)
 - support, 30
 - X performance and, 416
 - `/proc/interrupts` file for IRQ
 - information, 152–153, 154
- procmail MDA, 469
- profile configuration script, 165–167
- profiles for multiple configuration sets, 262
- ProFTPD, 492–493
- programs. *See* application software; *specific programs*
- proxy server, caching, 393, 406–407.
 - See also* Squid Web proxy
- ps command
 - basic process selection, 676, 813–814
 - complexity of, 813
 - described, 676
 - modifying output format, 677, 816–817
 - output columns addressing memory usage, 817
 - process selection by category, 676–677, 814
 - standard output formats, 677, 814–815
 - Unix98-style versus BSD-style options, 815
- pseudo-file-systems, 662
- psql utility (PostgreSQL), 376
- pstree command, 676, 679
- Python, 353
- python-ldap package, 856

Q

question mark (?) as Bash wildcard, 906–908
quota command, 695, 738, 743–744
quotacheck command, 695, 738, 739–740
quotactl command, 695
quotaoff command, 695, 738
quotaon command, 695, 738, 740
quotas. *See* file system quotas
quotastats command, 695

R

RAID (Redundant Array of Independent Disks)
 command-line tools, 162
 getting information about, 161
 hardware versus software, 160–161
 kernel configuration options, 646, 649
 levels, 160
 as metadvice, 156
 NFS performance and, 419–420
 partitioning during kickstart installation, 80–82
RAID Options dialog box, 80–81
raidreconf command, 162
raidstart command, 162
raidstop command, 162
RAM (random access memory)
 Apache performance and, 593
 buying a new PC and, 31
 database server performance and, 424
 displaying kernel usage with `slabtop`, 810–812
 displaying real-time usage with `top`, 679–680
 displaying usage with `free`, 806–808
 displaying usage with `vmstat`, 808–810
 kernel size considerations, 667–668
 monitoring usage with `sar`, 827–828
 system requirements, 31
 virtual file systems in, 149
 X performance and, 416, 417
ramdisks, virtual file system for, 150
ramfs virtual file system, 150

RAWRITE.EXE boot disk creation
 program, 35–36
rc scripts. *See* init scripts
RCP service, 431, 435
rc.sysinit program, 133
rdate command, 697
RDBMSs (relational database management systems). *See* database servers; MySQL; PostgreSQL
RDF (Resource Description Framework) Site Summary. *See* RSS feeds
read command (Bash), 932–934
README files, 882–883
Really Simple Syndication. *See* RSS feeds
real-time chat protocols, 435–436
RealVNC, 383, 390–391
rebooting. *See also* booting
 avoiding file system checks during, 891
 from login window, 98
 NFS losses and, 268–269
 shutdown command for, 138
 troubleshooting, 899–900
RECORD extension (X Windows), 417
recursive netgroups (NIS), 423
Red Hat Network. *See also* up2date
 agent
 accessing with Web browser, 608–614
 advantages of, 597
 Alert icon, 598
 Channels dialog box, 607
 Fedora Core and, 605
 kernel upgrade and, 616, 619
 registering with, 605–608
 Up2date Activate dialog box, 607, 608
 Up2date Create Login dialog box, 606–607
 Up2date Login Page, 606
 Web site, 605, 608–614
Red Hat Network Configuration dialog box, 599–602
Red Hat Package Manager. *See* RPM
Red Hat Web site, 608
Red Hat/Fedora Repository Web site, 768
redirecting I/O (Bash), 929–932
redundancy, disaster recovery and, 26

- Redundant Array of Independent Disks.
 - See* RAID
- registering with Red Hat Network, 605–608
- reiserfs (Reiser file system), 146
- relational database management systems (RDBMSs). *See* database servers; MySQL; PostgreSQL
- \$relay_domains configuration variable, 484
- remote access and control. *See* VNC (virtual network computing)
- removing. *See* deleting or removing
- renice command, 680
- repquota command, 695, 738, 743, 744
- Requests for Comment. *See* RFCs
- resize2fs command, 683, 687
- Resource Description Framework (RDF) Site Summary. *See* RSS feeds
- restarting. *See also* rebooting
 - DHCP server after configuration, 250
 - named daemon after changing /etc/named.conf file, 464
 - Netfilter (iptables) firewall, 851
 - network after configuring DHCP client, 250
 - SSH daemon, 509
 - X Window system to fix memory issues, 417
- restore command
 - interactive mode options, 792–793
 - options, 790–791
 - overview, 790
 - restoring the file system, 791–792
 - using interactively, 792–793
- restoring
 - extracting a tar archive, 794–795
 - file system, 790–793
 - MySQL to original state, 358
 - PostgreSQL database dump, 378
 - system administrator duties for, 7–9
 - testing the restore process, 8
- return values, 921
- reverse zone file, 449, 460
- reversing set notation (Bash), 907
- RFCs (Requests for Comment)
 - for CIDR, 245
 - defined, 246
 - for FTP, 492
 - further information, 246
- Rich Site Summary. *See* RSS feeds
- ring topology, 16–17
- RIP (Routing Information Protocol), 233
- rlogin remote login program, 435
- ROM BIOS chip, 128
- romfs virtual file system, 150
- root account. *See* root user or superuser
- root directory, 144
- root directory or partition (/)
 - creating with Disk Druid, 47
 - moving data to another partition, 142
 - overview, 143–144
 - root directory versus, 144
- root squashing, NFS and, 271, 277–278, 306–307, 730
- root user or superuser. *See also* Sudo (superuser do)
 - ACLs for, forthcoming, 731
 - building software as, avoiding, 775
 - capability bounding set, 731
 - defined, 4–5
 - etc directory files and, 163–164
 - home directory (root), 144
 - logging in as, 5
 - for MySQL, 355
 - NFS and root squashing, 271, 277–278, 306–307, 730
 - overview, 730–731
 - password lost or forgotten, 884
 - restrictions on powers of, 730–731
 - setting password, 61, 75
 - shutdown privileges and, 138
 - system administrator as, 4–5
- route command, 234, 238, 247
- routers, 232–233. *See also* gateways
- Routing Information Protocol (RIP), 233
- rpcinfo command, 286–287, 319
- RPCSEC_GSS configuration files, 279–280, 290
- RPM (Red Hat Package Manager).
 - See also* rpmquery command
 - BIND version, 446–447
 - building packages in stages, 763–764
 - building packages using SRPMs, 761–764

RPM (Red Hat Package Manager)
 (continued)
 building RPMs from source tarballs, 764
 checking software versions, 764–765
 command line options, 746–747
 components, 746
 dhcpd program installation, 248–249
 installing a kernel RPM, 619–620
 installing RPMs, 756–757
 kernel SRPM installation, 621–623
 location of installation and, 6
 overview, 745–746
 query mode, 748–755
 removing database entries, 747
 removing RPMs, 758
 RPM installation options, 756
 test runs (safe), 747
 third-party sites for RPMs, 768–771
 upgrading RPMs, 575–578
 valid build stages for `-b` mode, 761
 verifying RPMs, 758–761
RPM PBone Search site, 768
rpmfind.net site, 769
rpm.livna.org site, 769
rpmquery command
 Apache installation, verifying, 519
 checking software versions, 764–765
 Dovecot installation, verifying, 485
 formatting output, 754–755
 general form, 748
 listing all files in installed RPM, 752
 listing all installed RPMs, 751
 listing RPM information, 751–752
 listing RPM that installed a file, 753
 Mailman installation, verifying, 556
 MySQL installation, verifying, 354
 NTP installation, verifying, 397
 OpenSSH installation, verifying, 509
 options, 748–749
 PostgreSQL installation, verifying, 365
 querying package dependencies,
 750–751
 querying uninstalled packages, 753
 Sendmail installation, verifying, 473
 SquirrelMail installation, verifying, 563
 syntax, 748
 VNC installation, verifying, 383

vsftpd installation, verifying, 493
 which command with, 753
rsh remote command program, 435
RSS aggregators, 568–570
RSS feeds
 Atom-compliant RSS, 569
 BBC news example, 567–568
 content selection for, 570
 creating the feed file, 570–572
 creation and validation tools, 573
 defined, 567
 development of, 569
 further information, 568, 573
 meanings possible for acronym, 567
 minimum required elements, 571
 RSS aggregators for tracking, 568–570
 specification versions, 569
 syndication defined, 567
 turning on, 572–573
 uses for, 567–568
“RSS Tutorial for Content Publishers
 and Webmasters” (Nottingham), 573
rsync file transfer program, 435
rundig script ([ht://Dig](http://Dig)), 575–576, 578
rundig.cron script ([ht://Dig](http://Dig)), 575–576
runlevels
 changing, 136–137
 default, 136
 overview, 136
 purposes of, 136
 `rc.d` directories for scripts, 133
 runlevel 5 script, 133–135
 for troubleshooting, 137

S

sadc program, 822
Samba or SMB (Server Message Block)
 checking if installed, 340
 client, defined, 341
 connecting to client, 345–347
 connecting to server from Windows
 PC, 347–348
 creating users, 344–345
 described, 339, 340
 domains, 343, 344
 downloading the RPM file, 340
 exiting client connection, 346

- installing, 340–341
- kernel support required for, 339
- Kickstart Configurator setting, 85
- mounting client file system, 346–347
- NFS versus, 348
- packages, 340
- passwords, 342–343
- performance tuning, 423–424
- print queue configuration for, 221–222
- security, 343, 344
- server, defined, 341
- server configuration, 341–344
- smbclient utility, 345–346
- smb.conf file, 341–344, 423–424
- starting the server, 345
- Saou, Matthias (`freshrpms.net` webmaster), 768
- sar (system activity report) tool
 - command line options, 826–827
 - daily reports, 826
 - described, 805
 - invoked by `sa2` shell command, 826
 - monitoring CPU usage, 829–830
 - monitoring memory, 827–828
 - sadc program with, 822
 - sysstat package required for, 826
- SATA (Serial Advanced Technology Attachment) drives, 32
- `sa2` shell command, 826
- `sbin` directory, 144, 675
- `/sbin/ifdown` file, 196
- `/sbin/ifup` file, 196
- `/sbin/init` program, 133–135
- scheduling
 - database server performance and process schedulers, 424
 - I/O scheduling algorithms, 424
 - jobs with `cron`, 704–705
 - one-shot jobs with `at`, 702–704
- `scp` (Secure Copy), 431–433
- screen blanking, disabling, 903
- screen resolution
 - changing, 206, 214–215
 - Kickstart Configurator setting, 87
 - setting with Firstboot, 68–69
- Screensaver Preferences dialog box (GNOME), 118–119
- screensavers, 903
- scripts. *See also* Bash shell; configuration files
 - aliases in shell scripts, 928
 - automating, 674, 702–705
 - CGI scripts, 543–544
 - `/etc/rc.d/init.d/named`, 449
 - for `ht://Dig`, 575–576, 578
 - Kickstart Configurator settings, 91–92
 - managing init scripts, 196–202
 - NFS initialization, 274, 285–286
 - for one-shot jobs, 702–704
 - `passwd` command with, 714
 - PHP, 545–546
 - post-installation, 92
 - pre-installation, 91–92
 - for scheduled jobs, 704–705
 - shell configuration, 164–167
 - Squid initialization, 409
 - SquirrelMail configuration, 565–566
- SCSI (Small Computer System Interface)
 - controller, checking support for, 33
 - hard drives, support for, 32
 - kernel configuration options, 648
 - `mt-st` package for tape backups, 784–787
- SD card support, 660
- search functionality for Web sites, 574–578. *See also* `ht://Dig`
- Secure Copy (`scp`), 431–433
- Secure File Transfer Program (`sftp`), 433, 509–510
- Secure Shell. *See* SSH
- secure shell daemon (`sshd`), 436
- Secure Sockets Layer. *See* SSL
- security. *See also* authentication; network security; passwords; SELinux (Security-Enhanced Linux)
 - balancing utility with safety, 11, 12, 415
 - email, 486–489
 - importance of, 10–11, 21
 - installing only needed packages, 62
 - internal threats, 21–22
 - Internet connection sharing and, 14
 - Internet services, less secure, 434–436
 - Internet services, secure, 430–433
 - kernel configuration options, 664

security (*continued*)

- mailing list for Red Hat Linux, 23
- MySQL, 355–358
- NFS issues, 270–271, 305–307
- NIS+ improvements, 334
- NTP autokey encryption, 399–401, 403–404
- policy, 22–23, 27
- Postfix mail server, 482–483
- Samba, 343, 344
- secure NFS, 290
- security log, 27, 28
- Sendmail, 478
- server and service issues, 5–6
- shadow password system, 84, 709, 710
- SMTP, 488–489
- SSL, 507–509, 546–554
- system administrator duties for, 4, 10–12
- for system configuration files, 163–164
- update procedure, 23
- for VNC, 384–386
- xinetd features, 437

Security Level Configuration tool

- enabling SELinux, 838, 839
- enforcing SELinux security, 842, 843
- modifying SELinux targeted policy, 843–845

Netfilter configuration, 848–850

VNC firewall configuration, 384–386

select control statement (Bash), 920, 924–925, 926–928

SELinux (Security-Enhanced Linux)

- changing policies, 839, 840
- checking current status, 842
- community resources, 846
- configuration file for, 185
- daemon executables, 841
- default configuration, 838
- discretionary access control (DAC), 836
- enabling manually, 842–843
- enabling with Security Level Configuration tool, 838–839
- enforcing security, 842, 843
- ensuring changes have taken effect, 840, 843
- Fedora Project and Red Hat resources, 845

- firewall configuration for, 58
- further information, 58, 845–846
- General Public License for, 836
- getenforce command, 842
- mandatory access control (MAC), 837
- modifying targeted policy, 843–845
- NSA resources, 58, 845
- overview, 835–836
- role-based access control (RBAC), 837
- Security Level Configuration tool for, 838–839, 842, 843–845
- sestatus command, 840, 843
- setenforce command, 842
- strict versus targeted policies, 838
- system log messages indicating problems, 840–841
- uses for, 835–836

selinux directory, 144

semicolon (;) terminating MySQL commands, 357

sendmail package, 473

Sendmail program

- aliases file for, 169, 476–478
- checking if installed, 473
- checking if running, 473–474
- configuring, 474–475
- email aliases file, 169
- /etc/mail/sendmail.cf file, 474–475, 587–588
- /etc/mail/sendmail.mc file, 475, 476
- /etc/sysconfig/sendmail file, 186, 587–588
- files and directories used by, 478
- mail queue management, 476
- Mailman with, 558
- m4 macro processor, 475–476
- OSTYPE macro, 475–476
- packages, 473, 475, 476
- performance tuning, 587–588
- popularity of, 472
- security, 478
- starting, 474
- stopping, 479
- verifying startup at boot time, 473

sendmail-cf package, 473, 475, 476

- sendmail-devel package, 473
- sendmail-doc package, 473
- Serial Advanced Technology
 - Attachment (SATA) drives, 32
- serial port support, 654
- Server Message Block (SMB). *See* Samba
- servers, 5–6. *See also specific kinds*
- server-side includes (SSI), 540–542
- Service Configuration tool, 494
- Service Location Protocol (SLP), 319
- service network restart
 - command, 250
- services. *See also* Internet services;
 - optimizing network services; *specific services*
 - finding ports and protocols used by, 849
 - firewall configuration for, 57–58
 - listing with `chkconfig`, 200, 201
 - not requiring servers, 5
 - requests for network services, 393
 - rule of thumb for, 14
 - security issues, 5–6
 - standalone services, 440–441
 - turning on or off with `chkconfig`, 202
 - unnecessary, disabling, 581
 - xinetd-started services, 439–440
- `sestatus` command (SELinux), 840, 843
- set operator (Bash), 906–907
- `setenforce` command (SELinux), 842
- `setquota` command, 695
- `sftp` (Secure File Transfer Program), 433, 509–510
- SGI XFS (Extended File System), 148
- shadow passwords. *See* `/etc/shadow` file
- shadowed group files, 720, 722–723
- sharing files. *See* NFS (Network File System)
- shell configuration scripts
 - `bashrc`, 165
 - `csh.cshrc`, 165
 - described, 164
 - dot in front of names, 165
 - `/etc/profile`, 165–167
 - files read by shells on startup, 167
 - in home directories, 164–165
 - `zshrc`, 165
- shell functions (Bash), 928–929
- `showmount` command, 288–289
- shutting down. *See also* stopping
 - from GNOME, 138
 - from login window, 98
 - man page for, 138
 - need for understanding, 127
 - root user needed for, 138
 - shutdown command for, 138
 - troubleshooting, 899–900
- Simple Mail Transfer Protocol. *See* SMTP
- Sistina GFS (Global File System), 148–149
- “skeleton” configurations, 6–7
- `slabtop` command, 806, 810–812
- `slapadd` utility (LDAP), 856
- `slapcat` utility (LDAP), 856
- `slapd` LDAP server, 856
- `slapd.conf` file (LDAP), 582–583, 857–859
- `slapindex` utility (LDAP), 856
- `slappasswd` utility (LDAP), 856, 858–859
- slash (/). *See* root directory or partition (/)
- SLP (Service Location Protocol), 319
- Small Computer System Interface. *See* SCSI
- smart host, running Postfix on, 483–484
- SMB (Server Message Block). *See* Samba
- `smbclient` utility, 345–346
- `smb.conf` file
 - example configuration, 341–342
 - [global] section, 342–343
 - [homes] section, 343–344
 - overview, 341
 - performance tuning, 423–424
 - [printers] section, 344
- `smbd` daemon, 186
- `smbmount` command, 346–347
- SMP (symmetric multiprocessing), 30
- SMTP (Simple Mail Transfer Protocol).
 - See also* email
 - firewall configuration for, 58
 - MIME implementation, 470–471
 - overview, 470–471

SMTP (*continued*)

- Postfix optimization for, 588–589, 590
- security, 488–489
- software RAID, 161. *See also* RAID (Redundant Array of Independent Disks)
- sound cards
 - configuration information, 886
 - kernel configuration options, 656
 - support for, 33
 - troubleshooting, 885–887
- source RPMs. *See* SRPMs
- spamming, 488
- special characters (Bash), 906–909
- speeds. *See also* performance
 - bus topology, 16
 - PCI and PCI-X bus, 30–31
 - RAM versus swap space, 31, 47
 - star topology, 15
- spoofing, email, 488
- square brackets ([]) as Bash set operators, 906–907
- Squid Web proxy
 - configuration file, 409–411
 - initialization script, 409
 - installing, 407
 - IP forwarding for, 408–409
 - key configuration parameters, 409–410
 - Netfilter (iptables) configuration for, 411–412
 - overview, 407
 - starting, 412
 - steps for configuring, 408
 - testing the configuration, 412–413
 - verifying kernel configuration, 408–409
- SquirrelMail
 - Apache required for, 563
 - checking if installed, 563
 - connecting to, 563–565
 - further information, 567
 - IMAP4 for, 563, 564–565, 566
 - reconfiguring, 565–567
 - themes, 566
- srm.conf file, 520
- SRPMs (source RPMs)
 - building packages using, 761–764
 - building the software package, 775–776

- configuring build environment, 772
- configuring source code, 773–775
- defined, 746
- installing the software package, 777–778
- for kernel, 621–623
- testing the build, 776
- unpacking the source code, 772–773
- srv directory, 144
- SSH (Secure Shell)
 - firewall configuration for, 57
 - overview, 434–436
 - scp with, 431–433
 - sftp over, 433, 509–510
 - Telnet versus, 430, 434
 - tunneling protocols through, 431
- sshd (secure shell daemon), 436
- SSI (server-side includes), 540–542
- SSL (Secure Sockets Layer)
 - CAs, 547, 548, 549, 554
 - creating a self-signed certificate, 549–551
- CSR, 550
- encryption, 547
- further information, 546
- information in certificates, 547–548
- obtaining certificates from CAs, 554
- overview, 546–549
- root certificates, 548
- running vsftpd over, 507–509
- self-signed versus CA-signed certificates, 549
- trust, 547
- using a self-signed certificate, 551–554

standard installation. *See also* kickstart installation

- additional CDs or documentation disks, 69–70
- additional language support, 58–59
- boot disk creation, 35–36
- configuration steps, 51
- disk-checking process, 37
- firewall configuration, 56–58
- Firstboot, running, 65–70
- GRUB installation, 51–54
- hardware detection during, 37
- hardware inventory, 30–34

- hardware support, checking, 34
- help messages, 38
- installation type, 41–42
- keyboard configuration, 39
- language for Red Hat installation, 38
- manual online, 29
- network configuration, 54–56
- new system versus upgrade, 40
- package group selection, 62–64
- partitioning the hard disk, 42–51
- RHEL versus Fedora Core, 29
- root password setting, 61
- starting the installation, 36–42
- time zone setting, 59–60
- star topology, 15, 16–17
- starting or opening. *See also* booting;
 - rebooting; restarting
 - Apache startup process, 520–521
 - Apache Web server, 539
 - applications in GNOME, 109
 - autofs service, 303
 - Kerberos-related services, 870
 - Kickstart Configurator, 73
 - kickstart installation, 95–96
 - log viewer, 881
 - mailman service, 558
 - mysqld server daemon, 355
 - Nautilus File Manager, 99, 110
 - Network Configuration tool, 251
 - network services, 193–195, 196
 - NFS server manually, 286
 - NIS client daemon, 330, 331
 - NIS portmapper, 319
 - NIS yppasswdd daemon, 321
 - NIS ypxfrd daemon, 321–322
 - OpenLDAP server, 859
 - Postfix mail server, 481
 - PostgreSQL server, 368
 - Printer Configuration tool, 217–218
 - programs at boot time, 137
 - RAID devices, 162
 - Red Hat Network Configuration
 - dialog box, 599
 - Samba server, 345
 - Sendmail program, 474
 - Squid Web proxy, 412
 - standard installation, 36–42
 - start order for NFS daemons, 284
 - up2date agent, 602
 - up2date Agent Configuration tool, 599
 - User Manager application, 725
 - VNC server, 383–384, 388
- stat command, 691, 693
- static files, 142
- static IP addresses, 238, 260
- stopping. *See also* shutting down
 - Apache Web server, 539
 - network services, 196
 - RAID devices, 162
 - Samba client connection, 346
 - Sendmail, 479
- string comparison operators (Bash),
 - 913–916
- string I/O (Bash), 932–934
- strong quote (') in Bash, 908–909
- subnet mask or subnetwork mask,
 - 235, 240–241, 246–247
- subnetting
 - assigning numbers for, 242–244
 - class A subnets and subnet masks,
 - 243–244
 - class B subnets and subnet masks, 243
 - class C subnets and subnet masks, 242
 - gateway configuration, 246–247
 - IP numbers for network classes,
 - 239–240
 - network mask and, 240–244
 - preparation for, 241–242
 - unusable IP addresses created by, 241
- Sudo (superuser do)
 - aliases for, 734–735
 - basic philosophy, 732
 - /etc/sudoers configuration file,
 - 733–737
 - features, 732
 - global configuration defaults, 733
 - log file, 733
 - need for, 731–732
 - privilege specifications, 733
 - tips, 737
 - typical session, 732–733
 - visudo configuration file editor,
 - 733, 734–737
- supernetting, 244–245

- superuser. *See* root user or superuser
- superuser do. *See* Sudo
- surge protection, 25–26
- swap partition or swap space
 - creating with Disk Druid, 47
 - creating with mkswap, 683, 686–687
 - defined, 31
 - disabling, 683, 691
 - enabling, 683, 690–691
 - monitoring activity with *sar*, 827
 - overview, 47
 - paging, 47
- swapoff command, 683, 691
- swapon command, 683, 690–691
- switches versus hubs, 420
- symbolic links, 135, 683, 688
- symlinks command, 683, 688
- symmetric multiprocessing (SMP), 30
- sys directory, 144
- sysfs virtual file system, 155
- syslog daemon, 172–174, 482
- syslog.conf file, 172–174
- syslogd system logger, 733
- sysstat package, 822, 826
- system activity report tool. *See* *sar* tool
- system administrator. *See also* system administrator duties
 - learning and skills needed by, 4
 - responsibilities of, 4
 - as root user, 4–5
 - size of installation and, 4
 - uniqueness of Linux for, 3–4
- system administrator duties
 - backing up and restoring files, 7–9
 - configuring security, 10–12
 - creating and maintaining user accounts, 7
 - database decisions and, 352
 - freedom and flexibility implied by, 5
 - installing and configuring application software, 6–7
 - installing and configuring servers, 5–6
 - monitoring and tuning performance, 9–10
 - monitoring security, 12
 - overview, 12
- system clock. *See* date and time, system

- system environmental settings
 - for cron daemon, 171–172
 - for email aliases, 169
 - for file system information, 169–170
 - for GRUB, 139–140, 170–171
 - for log file rotation, 174–175
 - for prelogin banner, 168
 - for shared libraries, 174
 - for syslog daemon, 172–174
 - for Telnet prelogin banner, 168–169
 - for user messages at login, 168
- system failure, 8, 9
- system log
 - log viewer for, 185–186, 881–882
 - messages indicating SELinux problems, 840–841
 - NTP-related entries, 405
 - Sudo logging to, 733
 - as troubleshooting resource, 879
- system logger (*syslogd*), 733
- system requirements, 20–21
- system status icon (GNOME), 101
- system user, 69
- SystemV file system, 147

T

- talk real-time chat protocol, 435–436
- tape drives. *See also* backing up
 - mt-st package for backups, 784–787
 - other backup methods versus, 781
 - support for, 654
 - tape rotation, 783–784
- tar command for backups, 783,
793–795
- tar command
 - for backups, 783, 793–795
 - creating an archive, 794
 - extracting an archive, 794–795
 - options, 793
 - for unpacking source code, 772–773
- TCP/IP (Transmission Control Protocol/Internet Protocol). *See also* IP addresses; TCP/IP networking
 - development of, 229
 - Internet protocols referred to as, 429
 - MAC addresses, 230
 - Network Layer addresses, 230, 232

- packets, 230–231
- protocols in family, 230
- as stream-oriented protocol, 233
- typical services, 233
- TCP/IP networking. *See also specific protocols*
 - boot process hangs, 895
 - changing the NIC configuration, 259–260
 - configuring LANs during standard installation, 54–56
 - configuring with Network Configuration tool, 251–259
 - DHCP configuration, 247–250
 - DNS settings, 261
 - editing the configuration, 259–263
 - Ethernet device configuration, 252–254
 - gateways and routers, 246–247
 - information on physical requirements, 235
 - internal network configuration, 235–237
 - IP masquerading configuration, 263
 - kernel configuration options, 649–653
 - Kickstart Configurator settings for LANs, 83
 - managing hosts, 261–262
 - modem connection, 257–259, 893–895
 - network classes overview, 231–233
 - network configuration files, 188–196, 892
 - NIC setup, 233–237
 - profiles, 262
 - removing a NIC, 259
 - subnetting, 238–247
 - TCP/IP overview, 229–231
 - troubleshooting, 891–896
 - using two Ethernet cards, 896
 - wireless NIC configuration, 254–256
- tcsh shell, 165, 167
- Telnet
 - checking if Sendmail is running, 474
 - krb-telnet Kerberos application, 870–871
 - overview, 434
 - prelog banner, 168–169
 - SSH as replacement for, 430, 434
 - testing Dovecot IMAP server, 486
- text mode installation
 - installation CD not in drive and, 36
 - Kickstart Configurator setting, 75
- threadpool MPM (Apache 2), 517
- time, system. *See* date and time, system
- time server. *See also* date and time, system; NTP (Network Time Protocol)
 - checking if NTP package is installed, 397
 - configuring, 396–397
 - defined, 394
 - grep for NTP-related system log entries, 405
 - guidelines for wise use of NTP, 405–406
 - hardware clock for, 395, 405–406
 - internal, 405
 - as nonessential service, 393
 - NTP autokey encryption for, 399–401, 403–404
 - NTP client configuration, 401–405
 - NTP information online, 396
 - ntpd command for syncing with, 697, 702
 - ntpd time server daemon
 - configuration, 397–401
 - querying with ntpstat, 404
 - rdate command for setting system clock from, 697
 - selecting a solution for, 395–396
 - selecting reference clocks, 397–401, 406
 - system size and, 405–406
 - uses for, 394–395
- time zone
 - clock configuration file and, 177
 - Kickstart Configurator settings, 74
 - setting during installation, 59–60
- timetest command, 776
- tload command, 676, 680
- tmp directory, 144
- tmpfs virtual file system, 149
- Token Ring system (IBM), 16
- top command
 - default display columns, 819
 - described, 676, 806, 817–818
 - interactive keystrokes with, 820–821

`top` command (*continued*)
 invoking with no options, 818
 killing a task using, 821
 options, 818
 signals recognized by, 821
 syntax, 818
 using, 679–680
 windows displayed by, 819
topologies, 14–18
Transmission Control Protocol/Internet Protocol. *See* TCP/IP
trash can (GNOME), 101
tree topology, 17–18
troubleshooting. *See also* error messages
 boot problems, 899–900
 CD-ROM drive does not mount, 885
 CD-ROM drive not detected during installation, 884–885
 cyrus-imapd startup error, 900–901
 debugging DNS servers, 465
 displaying debugging information with RPM, 747
 email problems, 472
 emergency boot disk for, 904
 entering runlevel 1 for, 137
 file system problems, 888–891
 ht://Dig won't run, 900
 Internet resources for, 879
 IRQs, viewing, 152–153
 laptop video problems, 901–902
 log files as resources for, 879–882
 login problems, 883–884
 networking problems, 891–896
 NFS problems, 896–898
 package installation, 747
 README files for, 882–883
 screensavers and power management problems, 903
 shell commands not working, 888
 signal 7 and signal 11 problems, 902–903
 situations leading to problems, 875–876
 sound not working, 885–887
 step 1: identify the problem, 876
 step 2: reproduce the problem, 876–877

 step 3: look for changes, 877
 step 4: determine the most likely cause, 877–878
 step 5: implement a solution, 878
 step 6: keep documentation, 878
 unable to unmount a drive, 887
 X Window system startup problems, 903
true clustering, 24
tuning the system. *See* performance
twisted pair cabling, 15

U

ufs file system, 147
UIDs (user identification numbers), 707–708
umount command, 683, 690, 694–695, 887
uninterruptible power supply (UPS), 25
United States Naval Observatory (USNO), 395
unlocking passwords for user accounts, 713
until statement (Bash), 920, 923–924
upgrading RPMs, 757–758
upgrading the kernel. *See also* kernel configuration options
 customizing versus, 618
 deciding to upgrade or not, 616–617
 defined, 618
 getting information, 620
 installing kernel RPM, 619–620
 obtaining the kernel source, 620–628
 preparing to upgrade, 618–619
 pristine RPM for, 623–628
 Red Hat Network subscription and, 616, 619
 SRPM (source RPM) for, 621–623
 up2date agent and, 616, 619, 620
UPGs (user private groups), 723–725
UPS (uninterruptible power supply), 25
up2date agent. *See also* Red Hat Network
 advantages of, 597
 Alert icon, 598
 Channels dialog box, 603

- configuration file for, 181
 - configuring, 599–602
 - GPG for, 599, 600–601, 606
 - installed by default, 597
 - kernel upgrade and, 616, 619, 620
 - Package List dialog box, 604
 - Package Retrieval dialog box, 604–605
 - Red Hat Network Configuration dialog box, 599–602
 - Skipped Packages dialog box, 603–604
 - starting, 602
 - updating your system, 602–605
 - up2date Agent Configuration tool, 599
 - USB hard drives for backups, 781
 - USB support
 - audio devices, 657
 - gadget drivers, 659
 - host controller drivers (HCD), 657
 - host-side USB, 656–657
 - human interface devices (HID), 658
 - imaging devices, 658
 - mass storage devices, 657–658
 - miscellaneous drivers, 659
 - multimedia devices, 658
 - network adapters, 659
 - Serial Converter support, 659
 - user accounts. *See also* `/etc/passwd` file; `/etc/shadow` file; group accounts
 - changing login shell, 714
 - changing passwords from scripts, 714
 - command-line tools for, 708
 - creating multiple accounts, 715–716
 - creating single accounts, 711–712, 883–884
 - creating with User Manager, 726–727
 - deleting, 713, 728
 - disabling, 713
 - expiration policy for passwords, 714–715
 - group accounts versus, 718
 - importance of, 707, 708
 - killing all processes owned by a user, 936–938
 - locking or unlocking passwords, 713
 - modifying default values, 712–713
 - modifying existing accounts, 713
 - modifying multiple accounts at once, 715–717
 - modifying with User Manager, 727–728
 - passwords, 7
 - real versus logical, 707–708
 - removing passwords, 713
 - resetting a password, 883
 - system administrator duties for, 7
 - UIDs for, 707–708
 - updating only expired passwords, 713
 - user database files, 708–711
 - viewing login and process information, 717–718
 - viewing status message for users, 713
 - user database files. *See* `/etc/passwd` file; `/etc/shadow` file
 - user identification numbers (UIDs), 707–708
 - User Manager application
 - configuration file for, 185
 - creating group accounts, 728–729
 - creating user accounts, 726–727
 - deleting user accounts, 728
 - modifying group accounts, 729–730
 - modifying user accounts, 727–728
 - overview, 725–726
 - starting, 725
 - user private groups (UPGs), 723–725
 - `useradd` command, 708, 711–713
 - user-defined variables (Bash), 910–911
 - `userdel` command, 708, 713
 - `usermod` command, 708, 713
 - USNO (United States Naval Observatory), 395
 - `usr` directory, 144
 - `/usr/sbin` directory, 675
 - `/usr/share/squirrelmail/config` `/config/conf.pl` file, 565–566
 - UTC (Coordinated Universal Time), 59–60, 74, 698, 701
- ## V
- `var` directory, 144
 - `/var/ftp/pub` directory, 495
 - variable files, 142

- variables
 - Bash, 909–912
 - configuration (Postfix), 484
 - environment (PostgreSQL), 366–367, 376
 - /var/kerberos/krb5kdc/kadm5
 - .acl file, 869
 - /var/kerberos/krb5kdc/kdc.conf file, 868–869
 - /var/lib/nfs/etab status information file, 273, 281
 - /var/lib/nfs/rmtab status information file, 273, 280–281
 - /var/lib/pgsql/data/pg_hba.conf file, 368–372
 - /var/log directory, 879, 882
 - /var/log/messages file
 - messages indicating SELinux problems, 840–841
 - NTP-related entries, 405
 - Sudo logging to, 733
 - as troubleshooting resource, 879
 - /var/log/named.log file, 586
 - /var/log/sa directory, 826
 - /var/named/named.ca file, 449
 - /var/named/named.local file, 449
 - /var/spool/cron directory, 172
 - /var/yp/securenets file, 315, 316–317
- verifying. *See also* rpmquery command
 - kernel archive, 626–627
 - kernel configuration for Squid, 408–409
 - MySQL is running, 355–356
 - NIC is configured, 234
 - NIS portmapper is running, 319
 - Postfix startup at boot time, 481
 - RPMs, 758–761
 - Sendmail startup at boot time, 473
 - software versions, 764–767
- Veritas FREEVxFS file system, 148
- Very Secure FTP Daemon. *See* vsftpd
- VESA local bus (VL-bus), 31
- VFS (Virtual File Systems) layer, 145
- video cards. *See also* color depth; screen resolution
 - changing type with X Configuration tool, 208–209
 - kernel configuration options, 655–656
 - Kickstart Configurator settings, 87–88
 - support for, 31, 32
 - troubleshooting laptop video
 - problems, 901–902
 - video chipset, 32
 - XFree86 and, 31, 32
- Video For Linux, 655
- viewing. *See* displaying or viewing
- virtual block devices. *See* metadevices
- virtual file systems, 149–154
- Virtual File Systems (VFS) layer, 145
- virtual hosts or virtual servers, 514, 523, 537–539
- virtual memory, 31, 47. *See also* swap
 - partition or swap space
- virtual network computing. *See* VNC
- virtual private networks (VPNs), 381, 382
- visudo editor, 733, 734–737
- VL-bus (VESA local bus), 31
- vmstat command
 - changing the display unit, 808
 - column headings in output, 809–810
 - described, 806
 - displaying average memory usage, 808
 - displaying current memory usage, 808–810
 - free command versus, 808
 - redirecting output to a file, 940–941
 - syntax, 808
- vmstat.sh Bash script, 940–941
- VNC (virtual network computing)
 - advantages of, 381–382
 - authentication, 388–389
 - checking if installed, 383
 - client or viewer, 388–390
 - configuration file for, 186–187
 - defined, 381
 - further information, 382–383
 - for help desk applications, 382
 - overview, 381–383
 - packages, 383
 - running KDE on Windows via, 391–392
 - security, 384–386

- server setup, 383–388
- starting the server, 383–384, 388
- testing, 388–392
- uses for, 381, 382
- VPNs versus, 382
- VNC Authentication dialog box, 388–389
- VNC client or viewer, 388–390
- vnc package, 383
- VNC server
 - customizing, 386–388
 - firewall configuration for, 384–386
 - killing the running process, 392
 - starting, 383–384, 388
 - vncserver Perl script, 383–384, 388, 392
 - vncviewer command, 388
 - X Window system and, 387–388
 - xstartup file, 384, 386–388
- vnc-server package, 383
- vncserver Perl script, 383–384, 388, 392
- vncviewer command, 388
- volume groups, displaying, 159–160
- VPNs (virtual private networks), 381, 382
- vsftpd (Very Secure FTP Daemon)
 - banner message display, 500
 - checking if installed, 493
 - checking stock installation, 494–495
 - chroot mode, 500–501
 - configuration files, 495
 - configuring features, 497–501
 - configuring user-level access, 496–497
 - default configuration, 497–498
 - denying access based on email addresses, 500
 - disabling anonymous FTP, 496, 498, 501–502
 - disabling local user access, 498
 - download directory, 495
 - ease of use, 491, 492
 - enabling anonymous uploads, 498–499, 503–504, 505
 - enabling guest accounts, 504–507

- optimizing services, 590
- overview, 492–493
- project Web site, 492
- running from xinetd, 502–503
- running over SSL, 507–509
- showing messages for directories, 499
- starting at boot time, 501
- transfer log, 499
- vsftpd.xinetd file, 503, 506
- VxTools package, 148

W

- w command, 717–718
- wallpaper, X performance and, 417
- warnquota command, 738, 743
- weak quote (“”) in Bash, 909
- Web hosting, outsourcing, 20–21
- Web proxy. *See* caching proxy server; Squid Web proxy
- Web servers. *See also* Apache Web server
 - content negotiation, 515
 - firewall configuration for, 57
 - kickstart installation from, 76
 - overview, 517–519
- Web services. *See also specific services*
 - mailing lists, 555–563
 - optimizing, 590–593
 - RSS feed, 567–573
 - search functionality, 574–578
 - Web-based email, 563–567
- Web sites. *See* Internet resources
- Web-based email. *See* SquirrelMail
- wheel user, 731. *See also* Sudo (superuser do)
- while statement (Bash), 920, 923–924
- who command, 717
- Wieers, Dag (Red Hat/Fedora Repository webmaster), 768
- wildcards (Bash), 906–909
- Windows. *See* Microsoft Windows
- Winer, Dave (RSS designer), 569
- WinModems, 894–895
- wireless NIC configuration, 254–256
- worker MPM (Apache 2), 517

Workspace Switcher

GNOME, 102

KDE, 121

X

X Configuration tool

changing display color depth, 207

changing display resolution, 206

changing monitor type, 207–208

changing video card type, 208–209

dual monitor configuration, 209–210

overview, 205

root password needed for, 206

X server

configuring manually, 210–215

configuring with X Configuration tool,
205–210

restarting after configuration changes,
206

updating drivers for, 417

X Window system. *See also* GNOME

(GNU Network Object Model

Environment); KDE (K Desktop

Environment); X server

checking application versions, 767

improvements forthcoming, 416

Kickstart Configurator setting, 87

monitors and, 31–32

performance tuning, 416–418

RAM requirements, 31

restarting to fix memory issues, 417

running across a network,
performance and, 391

specifying desktop manager for, 178

startup problems, 903

unloading unused modules, 417

video cards and, 31, 32

VNC *xstartup* file and, 387–388

“X Window System Network

Performance” (Packard and Gettys),
416

x86 machines, disk partitioning on, 156

X.500 standard, 852

XFree86, 5, 31, 32, 903. *See also* X server;

X Window system

XFS (Extended File System) of SGI, 148

xinetd daemon

access control using, 437

configuration files for, 187, 193–194

log capabilities, 437

server configuration, 437–439

starting network services from, 193–194

vsftpd configuration, 502–503

xinetd.conf file, 193–194, 438–439

xinetd-started services, 439–440

xorg.conf file

changing color depth, 214

changing screen resolution, 214–215

man pages, 215

section names and their uses, 213–214

typical server configuration, 210–213

X Configuration tool as front end for,
205

xstartup file (VNC), 384, 386–388

XTRAP extension (X Windows), 417

Y

YP (Yellow Pages). *See* NIS (Network
Information Service)

ypbind NIS client daemon, 326–331

ypbind package, 311

ypcat command, 331

ypinit command, 315, 319–321

ypmatch command, 331, 332–333

yppasswd command, 331, 333

yppasswdd NIS password daemon,
315, 321

ypserv NIS server daemon, 316–318

ypserv package, 311

ypserv.conf file, 315, 316, 317–318

yp-tools package, 311

ypwhich command, 219–220, 331, 332

ypxfrd NIS server transfer daemon,
316, 321–322

Z

Zip drives, 645–646, 781

zone files, 449, 458–460

zsh shell, 165–167

zshrc configuration script, 165

GNU General Public License

Version 2, June 1991

Copyright © 1989, 1991 Free Software Foundation, Inc.

675 Mass Ave, Cambridge, MA 02139, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The “Program”, below, refers to any such program or work, and a “work based on the Program” means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term “modification”.) Each licensee is addressed as “you”.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program’s source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:
 - a) You must cause the modified files to carry prominent notice stating that you changed the files and the date of any change.
 - b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
 - c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:
 - a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
 - b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
 - c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.
5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.
6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and “any later version”, you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.
12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS