

100%

ONE HUNDRED PERCENT

COMPREHENSIVE
AUTHORITATIVE
WHAT YOU NEED

ONE HUNDRED PERCENT

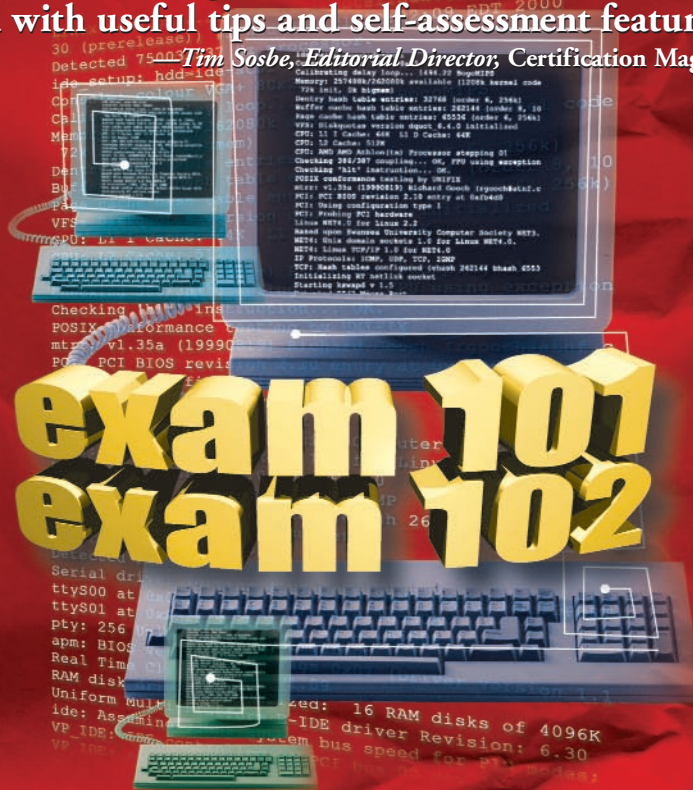
Master the material
for Linux Professional
Institute exams **101**
and **102**

Test your knowledge
with assessment
questions, scenarios,
and labs

Practice on state-
of-the-art test-
preparation software

“The ultimate reference guide to LPI Level 1 certification,
packed with useful tips and self-assessment features.”

Tim Sosbe, Editorial Director, Certification Magazine



LPI C 1

Certification Bible

Angie Nash and Jason Nash

Hungry Minds
Test Engine powered by



LPIC 1 Certification Bible

LPIC 1 Certification Bible

Angie Nash and Jason Nash



Hungry Minds™

Best-Selling Books • Digital Downloads • e-Books • Answer Networks • e-Newsletters • Branded Web Sites • e-Learning

Indianapolis, IN ♦ Cleveland, OH ♦ New York, NY

LPIC 1 Certification Bible

Published by
Hungry Minds, Inc.
909 Third Avenue
New York, NY 10022
www.hungryminds.com

Copyright © 2001 Hungry Minds, Inc. All rights reserved. No part of this book, including interior design, cover design, and icons, may be reproduced or transmitted in any form, by any means (electronic, photocopying, recording, or otherwise) without the prior written permission of the publisher.

Library of Congress Control Number: 2001090743

ISBN: 0-7645-4772-0

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

1P/R/Y/Q/W/Q/R/IN

Distributed in the United States by Hungry Minds, Inc.

Distributed by CDG Books Canada Inc. for Canada; by Transworld Publishers Limited in the United Kingdom; by IDG Norge Books for Norway; by IDG Sweden Books for Sweden; by IDG Books Australia Publishing Corporation Pty. Ltd. for Australia and New Zealand; by TransQuest Publishers Pte Ltd. for Singapore, Malaysia, Thailand, Indonesia, and Hong Kong; by Gotop Information Inc. for Taiwan; by ICG Muse, Inc. for Japan; by Intersoft for South Africa; by Eyrolles for France; by International Thomson Publishing for Germany, Austria, and Switzerland; by Distribuidora Cuspide for Argentina; by LR International for Brazil; by Galileo Libros for Chile; by Ediciones ZETA S.C.R. Ltda. for Peru; by WS Computer Publishing Corporation, Inc., for the

Philippines; by Contemporanea de Ediciones for Venezuela; by Express Computer Distributors for the Caribbean and West Indies; by Micronesia Media Distributor, Inc. for Micronesia; by Chips Computadoras S.A. de C.V. for Mexico; by Editorial Norma de Panama S.A. for Panama; by American Bookshops for Finland.

For general information on Hungry Minds' products and services 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.

For sales inquiries and reseller information, including discounts, premium and bulk quantity sales, and foreign-language translations, please contact our Customer Care department at 800-434-3422, fax 317-572-4002 or write to Hungry Minds, Inc., Attn: Customer Care Department, 10475 Crosspoint Boulevard, Indianapolis, IN 46256.

For information on licensing foreign or domestic rights, please contact our Sub-Rights Customer Care department at 212-884-5000.

For information on using Hungry Minds' products and services in the classroom or for ordering examination copies, please contact our Educational Sales department at 800-434-2086 or fax 317-572-4005.


For press review copies, author interviews, or other publicity information, please contact our Public Relations department at 317-572-3168 or fax 317-572-4168.

For authorization to photocopy items for corporate, personal, or educational use, please contact Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923, or fax 978-750-4470.

LIMIT OF LIABILITY/DISCLAIMER OF WARRANTY: THE PUBLISHER AND AUTHOR HAVE USED THEIR BEST EFFORTS IN PREPARING THIS BOOK. THE PUBLISHER AND AUTHOR MAKE NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE ACCURACY OR COMPLETENESS OF THE CONTENTS OF THIS BOOK AND SPECIFICALLY DISCLAIM ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. THERE ARE NO WARRANTIES WHICH EXTEND BEYOND THE DESCRIPTIONS CONTAINED IN THIS PARAGRAPH. NO WARRANTY MAY BE CREATED OR EXTENDED BY SALES REPRESENTATIVES OR WRITTEN SALES MATERIALS. THE ACCURACY AND COMPLETENESS OF THE INFORMATION PROVIDED HEREIN AND THE OPINIONS STATED HEREIN ARE NOT GUARANTEED OR WARRANTED TO PRODUCE ANY PARTICULAR RESULTS, AND THE ADVICE AND STRATEGIES CONTAINED HEREIN MAY NOT BE SUITABLE FOR EVERY INDIVIDUAL. NEITHER THE PUBLISHER NOR AUTHOR SHALL BE LIABLE FOR ANY LOSS OF PROFIT OR ANY OTHER COMMERCIAL DAMAGES, INCLUDING BUT NOT LIMITED TO SPECIAL, INCIDENTAL, CONSEQUENTIAL, OR OTHER DAMAGES.

Linux Professional Institute and the LPI logo are trademarks of Linux Professional Institute, Inc. The Linux Professional Institute does not endorse any third party exam preparation material or techniques. For further details please contact info@lpi.org.

Trademarks: All trademarks are property of their respective owners. Hungry Minds, Inc. is not associated with any product or vendor mentioned in this book.

 is a trademark of
Hungry Minds® Hungry Minds, Inc.

About the Authors

Angie Nash is an IT Consultant for her own firm, Tarheel Solutions. She works primarily with Linux and Microsoft operating systems to provide solutions for small businesses. Her free time is spent adding new products to her arsenal. She can be reached at angie@the-nashes.net.

Jason Nash is an independent consultant experienced with Linux, Solaris, and several BSD variants. He has written several books for the Microsoft world, but now spends most of his time in Linux and BSD. He can be reached at jason@the-nashes.net.

Credits

Acquisitions Editors

Nancy Maragioglio
Katie Feltman

Project Editors

Brian MacDonald
Kevin Kent

Technical Editor

Theresa Hadden-Martinez

Copy Editor

Kevin Kent

Editorial Managers

Ami Frank Sullivan
Kyle Looper

Project Coordinator

Emily Wichlinski

Graphics and Production Specialists

Gabriele McCann
Brian Torwelle

Quality Control Technicians

Andy Hollandbeck
Susan Moritz
Carl Pierce
Charles Spencer

Permissions Editor

Laura Moss

Media Development Specialist

Megan Decraene

Media Development Coordinator

Marisa Pearman

Proofreading and Indexing

TECHBOOKS Production Services

This book is dedicated to two women who have made a profound impact on both our lives. The unconditional love and support that we have received from Marie Ward and Melva Hamby are always with us.

Preface

Welcome to the *LPIC 1 Certification Bible*. This book is designed to help you prepare for the Linux Professional Institute Certification Level 1 exams 101 and 102. The Linux Professional Institute is a distribution-independent, nonprofit organization. The exams are developed to certify an individual's expertise with Linux systems. Level 1 certification is based upon many general tasks involving Linux systems. This book provides all the information you need to perform these tasks. Because of this, the book is useful as a study guide as well as a general Linux reference. We believe that this book will prove to be a useful tool when preparing for the LPIC exams and that you will want to keep it nearby as a handy resource while working with Linux systems.

How This Book Is Organized

This book is organized into four major parts, followed by several appendixes, a robust glossary, an index, and a compact disc.

Here's what you'll find in this book:

Part I: Installing Linux and Getting Started

Part I presents basic information about Linux. It covers the basic installation and configuration of the Linux operating system. This part introduces the shell environment and its usage. Finally, this part covers software installation, including the packaging systems used on Debian and Red Hat distributions.

Part II: Getting Around in Linux

Part II covers the basics of using Linux. This part explains many of the text processing tools available for Linux. Information on working with Linux partitions and file systems is also covered in this section. Another task required to get around in Linux is managing files and directories. This part introduces the commands most often used for this task. The documentation resources available to aid with the proper use of these commands and many of the tasks required on Linux are also covered here. Part II also explores the boot process, detailing the use of various configuration files and run levels. Finally, this part explores XFree86, the graphical user interface available for Linux.

Part III: Administering Linux

Part III is all about administering and securing resources on a Linux computer. This part begins by explaining how to manage users and groups. It also presents detailed instructions on how to administer the system. This discussion covers a variety of tasks such as system logging, making backups, and managing quotas. Additionally, Part III explores the ins and outs of managing printing. This part also shows you how to work with and even upgrade the kernel. Then Part III ends by building on Part I, including more detailed information on shell usage.

Part IV: Managing the Network

Part IV covers the various concerns of a networked Linux computer. This part introduces the basics of TCP/IP protocols, files, and tools. Additionally, Part IV explains how to create and configure network and dial-up connections. It also covers the various server functions Linux can provide on a network. Finally, Part IV shows you how to efficiently secure a Linux system.

At the end of the book are several valuable appendixes. You'll find full practice exams for both the 101 and 102 tests, a table of the actual exam objectives for both LPIC exams (including cross-references to the section in this book where each objective is covered), important information and tips on how to prepare for the exams, and a complete listing and description of the contents of the compact disc included with this book.

CD-ROM

This book includes a CD-ROM with several useful programs and utilities. First you'll find the Hungry Minds test engine, which is powered by Boson Software and features practice test questions to help you prepare for the exam. The disk also includes an electronic version of the book in PDF format along with Adobe Acrobat Reader so you can easily navigate this resource. Also included are several useful Linux guides, FAQs, and HOWTOs, documentation that can help you grow in your facility with and understanding of Linux systems.

How Each Chapter Is Structured

When this book was designed, a lot of thought went into its structure, particularly into the specific elements that would provide you with the best possible learning and exam preparation experience.

Here are the elements you'll find in each chapter:

- ◆ A list of exam objectives (by exam) covered in that chapter
- ◆ A Chapter Pre-Test that enables you to assess your existing knowledge of the topic
- ◆ Clear, concise text on each topic
- ◆ Step-by-step instructions on how to perform Linux tasks
- ◆ A Key Point Summary
- ◆ A comprehensive Study Guide that contains the following:
 - Exam-style Assessment Questions
 - Scenario problems for you to solve, as appropriate
 - Lab Exercises to perform on your computer, as appropriate
 - Answers to Chapter Pre-Test questions, Assessment Questions, and Scenarios

How to Use This Book

This book can be used either by individuals working independently or by groups in a formal classroom setting.

For best results, we recommend the following plan of attack as you use this book. First, take the Chapter Pre-Test and then read the chapter and the Key Point Summary. Use this summary to see if you've really got the key concepts under your belt. If you don't, go back and reread the section(s) you're not clear on. Then do all of the Assessment Questions and Scenarios at the end of the chapter. Finally, do the Lab Exercises. Remember that the important thing is to master the tasks that are tested by the exams.

The chapters of this book have been designed to be studied sequentially. In other words, it would be best if you complete Chapter 1 before you proceed to Chapter 2. A few chapters could probably stand alone, but all in all, we recommend a sequential approach. The Lab Exercises have also been designed to be completed in a sequential order and often depend on the successful completion of the previous labs.

After you've completed your study of the chapters and reviewed the Assessment Questions and Lab Exercises in the book, use the test engine on the compact disc included with this book to get some experience answering practice questions. The practice questions help you assess how much you've learned from your study and also familiarize you with the type of exam questions you'll face when you take the real exams. Once you identify a weak area, you can restudy the corresponding chapters to improve your knowledge and skills in that area.

Prerequisites

Although this book is a comprehensive study and exam preparation guide, it does not start at ground zero. We assume you have the following knowledge and skills at the outset:

- ◆ Basic terminology and basic skills to use Linux systems
- ◆ Basic software and hardware terms used with computers and networking components

If you meet these prerequisites, you're ready to begin this book.

How to determine what you should study

Your individual certification goals will ultimately determine which parts of this book you should study. If you want to pass both LPIC exams or simply want to develop a comprehensive working knowledge of Linux, we recommend you study the entire book in sequential order.

If you are preparing only for the 101 exam, we suggest you follow the recommended study plan shown in Table 1.

Table 1
Chapters to Study for Exam 101

<i>Chapter Number</i>	<i>Chapter Title</i>
2	Using the Shell
4	Processing Text
5	Using Partitions and File Systems
6	Managing Files
7	Using Documentation
8	Understanding the Boot Process
10	Managing Users and Groups
11	Administering the System

If you are preparing only for the 102 exam, we suggest you follow the recommended study plan shown in Table 2.

Table 2
Chapters to Study for Exam 102

<i>Chapter Number</i>	<i>Chapter Title</i>
1	Installing Linux
3	Installing Software
5	Using Partitions and File Systems
8	Understanding the Boot Process
9	Using X
12	Printing
13	Working with the Kernel
14	Using Shells and Scripts
15	Networking Fundamentals
16	Managing Network Services
17	Managing Security

Hardware and Software You Need

You need access to various hardware and software to be able to do the Lab Exercises in this book. It's extremely important that you do these labs to acquire the skills tested by the LPIC Level 1 exams.



Some of the Lab Exercises in this book have the potential to erase or corrupt data on existing hard disks. Make sure you back up all important data and programs before you attempt to perform the labs. Better yet, do the labs on a computer that doesn't contain any vital data or programs.

Here are the minimum hardware requirements:

- ◆ Intel-based computer with Pentium/133MHz processor, 256MB of RAM, and 2GB of hard disk space.
- ◆ Keyboard
- ◆ CD-ROM drive
- ◆ Mouse or other pointing device
- ◆ VGA monitor and graphics card
- ◆ Network adapter card

We strongly recommend that you use only hardware found on the Linux Hardware Compatibility List. This list can be located on the Web sites of all Linux distributions such as <http://www.redhat.com/support/hardware/> for Red Hat and <http://www.ibiblio.org/mdw/HOWTO/Hardware-HOWTO.html> for Debian.

Optional equipment that you might benefit from using includes the following:

- ◆ Printer
- ◆ Tape drive
- ◆ Modem and Internet connection (so you can access online resources)

The software you need includes Linux Installation Software. This book covers both Red Hat- and Debian-based distributions.

Conventions Used in This Book

Every book has its own set of conventions, so we'll explain the ones we've used in this book to you right up front.

New terms

How could we talk about Linux and other computer stuff without using all kinds of fancy acronyms and terms, without using that alphabet soup you throw into everyday conversation around the dinner table that causes your family members to roll their eyes?

We've chosen to italicize new or potentially unfamiliar terms, such as *pwd*, as we define them. Normally, we define a new term right after its first mention. If you happen to see an unfamiliar word that is italicized, such as *pwd*, but is not followed by a definition, you can flip to the glossary to read the definition of the term.

Code

All code listings and commands in this book are presented in monospace font, like this:

```
# ls -al
```

We've also used this type of font to identify names of files, folders, Web addresses, and character-based screen content when presented verbatim.

When you see monospace font presented in italics, the italicized text represents a variable that could actually have a different name. This can be used to represent a filename or perhaps a directory, like this:

```
l dd file_name
```

When you see monospace font presented in bold, the bold text represents text that you would type, usually at the command prompt, like this:

```
[root@redhat jason]# ls /etc/pine*
```

Icons

Several different icons are used throughout this book to draw your attention to matters that deserve a closer look:



This icon is used to warn you that something unfortunate could happen if you're not careful. It also points out information that could save you a lot of grief. It's often easier to prevent a tragedy than it is to fix it afterwards.



This icon points you to another place in this book for more coverage of a particular topic. It may point you back to a previous chapter where important material has already been covered, or it may point you ahead to let you know that a topic will be covered in more detail later on.



This icon points out important information or advice for those preparing to take the LPIC exams.



Sometimes things work differently in the real world than books—or product documentation—say they do. This icon draws your attention to the authors' real-world experiences, which will hopefully help you on the job, if not on the LPIC exams.



This icon appears at the beginning of certain parts of the chapter to alert you that objective content is covered in this section. The text of the objective appears next to this icon for your reference.



This icon is used to draw your attention to a little piece of friendly advice, a helpful fact, a shortcut, or a bit of personal experience that might be of use to you.

How to Contact Us

We've done our very best to make sure the contents of this book are technically accurate and error free. Our technical reviewer and editors have also worked hard toward this goal.

However, we know that perfection isn't a possibility in the real world, and if you find an error, or have some other comment or insight, we'd appreciate hearing from you. You can contact us via the Internet at angie@the-nashes.net and jason@the-nashes.net.

We always read all of our readers' e-mail messages and, when possible, include your corrections and ideas in future printings. However, because of the high volume of e-mail we receive, we can't respond to every message. Please don't take it personally if we don't respond to your e-mail message.

Also, one last note: although we enjoy hearing from our readers, please don't write to us for product support or for help in solving a particular Linux problem you're experiencing on your computer or network. In this book we cover various places available for locating support with these types of problems.

Well, that about wraps up the general comments. From here you can get started on the nuts and bolts of learning about Linux and get ready to pass those exams. We wish you great success!

Acknowledgments

I must first give my thanks to Nancy Maragioglio, Acquisitions Editor, Brian MacDonald, Senior Project Editor, and Kevin Kent, Project and Copy Editor, for this wonderful opportunity and all the work they have done to ensure that this book reaches its full potential. This could not be done without all of their hard work and dedication. The people at Hungry Minds have contributed to making this book a positive experience, even amidst the frantic pace that seems to surround me.

I must acknowledge my husband and coauthor Jason, but words fail me. He is truly my best friend and partner through all that life brings. I also have a wonderfully supportive and loving family that has always made me feel special. My parents, Martin and Kathy Brummitt, have selflessly sacrificed for my benefit. The debt I owe them can never be repaid. I must also send my love and thanks to my siblings: Jenny, Michael, and Chris; to Debbie, Jerry, Jeff, Steve, Renee, and Kim Hamby; and to Nettie Cope and all the others in my family who have contributed to my life.

I must include a special thanks to Tim and Margaret Franks. You are wonderful people and deserve only the best life has to offer. I love you both. I also need to thank my wonderful friends, Cathelene Shanaberger and Lisa Anderson. They have been there for me whenever I have needed an ear or a shoulder. To all of my friends online and through life, I could never fit everyone in this book. You know who you are and so do I. Thank you.

—Angie Nash

Every acknowledgment you read, if you take the time to do so, starts with the author thanking the publishing group they work with. I always figured that this was a simple gratuitous action, but not anymore. The people at Hungry Minds are still excellent to work with. On this project I would like to thank Nancy Maragioglio, our Acquisitions Editor, Brian MacDonald, our Senior Project Editor, and Kevin Kent, our Project and Copy Editor. Good people make projects like this go much smoother, especially when deadlines seem to appear from nowhere.

At the top of my list is my wife Angie, who coauthored this book. She is by far the most important thing in my life and my best friend, who I love very much. When you look back in your life, it sometimes surprises you the influence that others had, and without them you would not be where you are today. My mother and stepfather, Peggy and Tim Franks, and my grandmother Marie Ward have helped me more than this entire book could hold. Special thanks are also due to my father Bill Nash, my sister Jeanie, and my grandparents Homer and Frances Nash.

This is the part I hear about long after the book is published. First, I'd like to say thank you to some of my close friends: Jacob Hall, Robert Mowlds, Johnathan Harris, Todd Shanaberger, and Lee Johnson. We have a large number of online friends, and instead of listing them here and hearing about a couple we forgot, we've just used their names in examples throughout the book. You know who you are. Finally, I'd like to thank the members of the open source community that are leading a revolution. Without them none of this would be possible.

—*Jason Nash*

Contents at a Glance

Preface	ix
Acknowledgments	xvii
Part I: Installing Linux and Getting Started	1
Chapter 1: Installing Linux	3
Chapter 2: Using the Shell	63
Chapter 3: Installing Software	93
Part II: Getting Around in Linux	157
Chapter 4: Processing Text	159
Chapter 5: Using Partitions and File Systems	201
Chapter 6: Managing Files	231
Chapter 7: Using Documentation	275
Chapter 8: Understanding the Boot Process	295
Chapter 9: Using X	331
Part III: Administering Linux	385
Chapter 10: Managing Users and Groups	387
Chapter 11: Administering the System	409
Chapter 12: Printing	443
Chapter 13: Working with the Kernel	477
Chapter 14: Using Shells and Scripts	513
Part IV: Managing the Network	559
Chapter 15: Networking Fundamentals	561
Chapter 16: Managing Network Services	599
Chapter 17: Managing Security	695
Appendix A: What's on the CD-ROM	747
Appendix B: Practice Exams	753
Appendix C: Objective Mapping	775
Appendix D: Exam Tips	787
Glossary	791
Index	803
End-User License Agreement	845
CD-ROM Installation Instructions	848

Contents

Preface. ix
Acknowledgments. xvii

Part I: Installing Linux and Getting Started 1

Chapter 1: Installing Linux 3

- History of Linux and GNU 5
- The GNU General Public License 6
- What Does *Free* Mean? 7
- Why Use Linux? 7
 - Linux is multiuser 8
 - Linux is multitasking 8
 - Linux is stable 8
 - Linux has lots of available software 9
 - Linux has a wide range of supported hardware 9
 - Linux is fast 9
- Overview of the Linux Architecture 9
 - Kernel space 9
 - User space 10
- Linux Distributions 10
 - Red Hat 10
 - Mandrake 11
 - Debian 11
 - SuSE 11
 - Slackware 11
 - Caldera 11
 - Turbolinux 12
- Preparing Hardware 12
 - CPU requirements 13
 - Memory requirements 13
 - Hard disk controller requirements 13
 - Hard disk space requirements 14
 - Video requirements 14
 - BIOS settings 14
 - Peripherals and other hardware 15
- Resolving Conflicts and Configuring Plug-and-Play Hardware 18
 - Hardware addresses 18
 - Viewing configuration addresses 18
 - Configuring Plug-and-Play devices 20

Partitioning Schemes	21
Using fdisk	23
Using Disk Druid	25
Using cfdisk	26
Boot Managers	28
Installing Linux	28
Red Hat installation	28
Debian installation	41
Assessment Questions	59
Scenarios	61
Answers to Chapter Questions	61
Chapter Pre-Test	61
Assessment Questions	61
Scenarios	62
Chapter 2: Using the Shell	63
Understanding Shells	65
Using the Command Line	67
Command completion	70
Editing commands with the Readline Library	70
Command substitution	71
Using the history file	71
fc	72
Environment Variables and Settings	72
Editing the PATH variable	73
The init process and the PATH variable	76
Prompt	76
HOME	77
Managing Processes	77
Modifying Process Priorities	82
Assessment Questions	85
Scenarios	89
Lab Exercises	89
Answers to Chapter Questions	90
Chapter Pre-Test	90
Assessment Questions	90
Scenarios	92
Chapter 3: Installing Software	93
Installing Software from Source Code	95
Obtaining the source code	96
Decompressing the tarball	97
Running the configure script	98
Making changes to the Makefile	100
Compiling the software	101
Installing the software	102

Managing Shared Libraries	103
Viewing required shared libraries	104
Setting library paths	104
Configuring shared libraries	104
Red Hat Package Manager	105
Package files	106
The RPM database	107
The rpm tool	107
Debian Package Management	118
Using dpkg	119
Using dselect	127
Using apt-get	132
Using alien	137
Assessment Questions	141
Scenarios	144
Lab Exercises	145
Red Hat labs	145
Debian labs	147
Answers to Chapter Questions	153
Chapter Pre-Test	153
Assessment Questions	153
Scenario Answers	155

Part II: Getting Around in Linux

157

Chapter 4: Processing Text 159

Working with Input and Output	161
Redirection	161
Pipes	164
tee	165
xargs	166
Modifying Text with Filters	167
Sorting lines of a file	167
Cutting text	170
Pasting text	171
Converting tabs to spaces	172
Formatting paragraphs	173
Deleting or substituting characters	175
Viewing the beginning of a file	176
Viewing the end of a file	176
Joining multiple files	177
Dividing files into multiple pieces	179
Displaying files in other formats	180
Converting files for printing	181
Displaying files backwards	182
Displaying numeric details of a file	183
Adding line numbers to a file	183

Using the stream editor	185
Using grep	187
Enhancing Searches with Regular Expressions	188
Assessment Questions	192
Scenarios	195
Lab Exercises	195
Answers to Chapter Questions	197
Chapter Pre-Test	197
Assessment Questions	197
Scenarios	199
Chapter 5: Using Partitions and File Systems	201
Linux File Systems Overview	203
File system types	204
Considerations when making a file system	206
Creating Partitions and File Systems	207
Partition types	207
File system tools	208
Checking the File System	212
fsck	212
du	215
df	216
Mounting and Unmounting File Systems	217
Mounting file systems	218
Unmounting file systems	218
Checking available file systems with /etc/fstab	219
Checking mounted file systems with /etc/mntab	220
Assessment Questions	222
Scenarios	225
Lab Exercises	225
Answers to Chapter Questions	228
Chapter Pre-Test	228
Assessment Questions	228
Scenarios	229
Chapter 6: Managing Files	231
Managing Files	233
Changing directories	233
Listing directory contents	235
Determining a file type	239
Changing file time stamp	240
Copying files	241
Moving files	245
Deleting files	246
Creating directories	246
Understanding File System Hierarchy	247
Standard file locations	247
System directories	248

Locating Files	248
find	249
locate	249
which	250
whereis	251
Creating File Links	251
Hard links	252
Symbolic links	252
Working with Permissions	253
Symbolic and numeric permissions	253
Files, directories, and special files	253
User and group permissions	254
SUID and SGID	258
Sticky bit	258
Using Compression Tools	259
tar	259
gzip and gunzip	260
compress	261
bzip2	262
Managing Quotas	263
quota	263
edquota	264
repquota	265
quotaon and quotaoff	265
Assessment Questions	268
Scenarios	271
Answers to Chapter Questions	271
Chapter Pre-Test	271
Assessment Questions	272
Scenarios	273

Chapter 7: Using Documentation 275

Getting Help with Man Pages	277
Locating man pages	279
Searching man page sections	281
Using Documentation Stored in /usr/doc	284
Documentation on the Internet	285
Linux Documentation Project	285
Vendor sites	286
Newsgroups	286
Mailing lists	286
Creating Documentation	287
Providing Technical Support	287
Assessment Questions	290
Scenarios	292
Answers to Chapter Questions	292
Chapter Pre-Test	292
Assessment Questions	293
Scenarios	293

Chapter 8: Understanding the Boot Process 295

Using LILO	297
Configuring LILO	298
Installing and updating LILO	303
Viewing boot messages	304
Understanding Runlevels and init	305
Using runlevels	305
Configuring the init process	308
Customizing the Boot Process	311
BSD startup	311
Sys V startup	311
Troubleshooting the Boot Process	316
Troubleshooting LILO	317
Booting to single-user mode	319
Creating a boot disk	320
Creating repair disks	320
Assessment Questions	323
Scenarios	326
Lab Exercises	326
Answers to Chapter Questions	328
Chapter Pre-Test	328
Assessment Questions	328
Scenarios	330

Chapter 9: Using X 331

Overview of the X Window System	333
History of X	333
Architecture overview	333
Window managers	334
Desktop environments	335
Installing X	336
Installing with RPMs	336
Installing on Debian	336
Installing with binary packages	337
Versions of XFree86	338
Configuring X	340
Manually configuring the XF86Config file	341
Using XF86Setup	349
Using xf86config	355
Detecting video hardware	356
Fine tuning video	357
Configuring fonts	358
Starting X	359
Starting X manually	359
Using XDM	361
Using X	366
Choosing a window manager or environment	367
Using X clients	368

Using a terminal emulator	368
Customizing X applications	369
Using special keys	370
Managing bad applications	370
Running X and Clients Remotely	371
Configuring X security	371
Configuring remote clients	373
Configuring remote login	373
Assessment Questions	377
Scenarios	380
Lab Exercises	380
Answers to Chapter Questions	382
Chapter Pre-Test	382
Assessment Questions	382
Scenarios	383

Part III: Administering Linux

385

Chapter 10: Managing Users and Groups 387

Special Users	389
root	389
nobody	390
bin	390
Manually Adding Users and Groups	390
Storing user information	391
Storing group information	392
Picking numeric user and group ids	392
Creating a user by hand	392
Managing Users and Groups	393
Managing User and Group Accounts	394
Assigning and Using Passwords	398
Configuring Global and User Settings	400
/etc/profile	400
/etc/skel	401
Assessment Questions	403
Scenarios	405
Lab Exercises	405
Answers to Chapter Questions	406
Chapter Pre-Test	406
Assessment Questions	406
Scenarios	407

Chapter 11: Administering the System 409

Starting and Stopping Daemons	411
Using the /etc/rc.d scripts	411
Using the kill command	412

System Logging	413
Configuring system logging	414
Rotating system logs	416
Identifying problems using log files	418
Scheduling Jobs	421
Using the at utility	421
Using the batch utility	423
Using the crond daemon	423
Performing Backups	425
Planning the backups	425
Backup methods	427
Backup media	428
Other considerations	429
Backup commands	429
Limiting Core Dump Files	434
Assessment Questions	436
Scenarios	439
Lab Exercises	440
Answers to Chapter Questions	440
Chapter Pre-Test	440
Assessment Questions	441
Scenarios	442
Chapter 12: Printing	443
Installing Printers	445
Configuring the /etc/printcap file	446
Creating the spool directory and log file	448
Controlling printer access	449
Using Print Filters	450
Installing Apsfilter	450
Installing Magicfilter	451
Using PrintTool	452
Managing the Printer Services	453
Managing the printer daemon	453
Managing printers	453
Managing print queues	457
Managing print jobs	459
Printing Files	461
Using lpr	461
Using a2ps	464
Troubleshooting Printing Problems	464
lpd problems	464
Queue problems	465
Printer problems	466
File and directory problems	467
Space problems	467
Assessment Questions	469
Scenarios	472
Lab Exercises	473

Answers to Chapter Questions	474
Chapter Pre-Test	474
Assessment Questions	474
Scenarios	475
Chapter 13: Working with the Kernel	477
Kernel Overview	479
Kernel development	479
Kernel types	480
Managing modules	482
Reconfiguring and Installing a New Kernel	490
Obtaining the kernel source	490
Updating your source with patches	491
Configuring the kernel	493
Compiling the kernel	497
Installing the kernel	498
Creating a ramdisk	499
Configuring LILO	499
Testing the new kernel	500
Assessment Questions	502
Scenarios	505
Lab Exercises	505
Answers to Chapter Questions	510
Chapter Pre-Test	510
Assessment Questions	510
Scenarios	511
Chapter 14: Using Shells and Scripts	513
Using vi	515
Text editing practices	515
Opening files for editing	516
Exiting vi and saving files	516
Moving the cursor	517
Adding text	519
Deleting text	520
Copying and pasting	521
Searching for text	524
Undoing changes	527
Customizing the Shell Environment	527
Environment variables	528
Aliases	536
Special files	536
Options	539
Writing Simple Scripts	541
Starting a shell script	542
Writing a basic script	542
Testing conditions	543
Flow control	545
Reading user input	548
Script considerations	549

Assessment Questions	551
Scenarios	554
Answers to Chapter Questions	554
Chapter Pre-Test	554
Assessment Questions	555
Scenarios	556

Part IV: Managing the Network

559

Chapter 15: Networking Fundamentals 561

The TCP/IP Protocol Suite	564
Addresses	564
Network classes	565
Dividing networks with subnet masks	566
Protocols	569
Ports	571
Applications	573
Configuration and Troubleshooting	580
Managing network interfaces	580
Managing network configuration files	586
Configuring PPP	588
Assessment Questions	592
Scenarios	595
Answers to Chapter Questions	595
Chapter Pre-Test	595
Assessment Questions	596
Scenarios	597

Chapter 16: Managing Network Services 599

Using the Internet Super Server	602
Configuring inetd	602
Restarting the inetd process	604
Configuring Basic Network Services	605
Configuring an FTP server	605
Configuring Telnet	617
Using sendmail	618
Customizing the sendmail.cf	619
Aliasing and forwarding mail	623
Managing sendmail	624
Using Apache	625
Starting and stopping httpd	626
Configuring Apache	627
Using NFS	637
Configuring exports	638
Mounting exported directories	639
Managing the NFS server	640
Security considerations	642

Using Samba	643
Configuring Samba	643
Managing Samba	646
Client connections	647
Using DNS	651
Overview of DNS	651
The DNS namespace	651
DNS and BIND	654
Configuring BIND v8	654
Configuring a caching-only name server	667
Using BIND v4	668
Configuring client DNS	669
Using DNS tools	671
Managing the DNS server	675
Assessment Questions	679
Scenarios	682
Lab Exercises	682
Answers to Chapter Questions	692
Chapter Pre-Test	692
Assessment Questions	693
Scenarios	694

Chapter 17: Managing Security 695

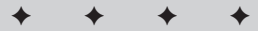
Performing Security Administration Tasks	697
Configuring TCP wrappers	697
SUID security issues	701
Managing packages	701
Using setgid	702
How and why not to use the r commands	703
Using SSH	705
Providing Host Security	718
Using shadow passwords	719
Removing unused services	720
Blocking unwanted connections with IP chains	721
Controlling authentication with Pluggable Authentication Modules	728
Monitoring security lists and sites	730
Limiting Users	732
Checking Security	734
nmap	734
SAINT	735
Nessus	735
crack	735
COPS	736
Tripwire	736
Bastille Linux	737
Assessment Questions	739
Scenarios	742
Lab Exercises	742

Answers to Chapter Questions	744
Chapter Pre-Test	744
Assessment Questions	745
Scenarios	746
Appendix A: What's on the CD-ROM	747
System Requirements	747
Using the CD with Microsoft Windows	748
Using the CD with Linux	748
What's on the CD	748
Hungry Minds test engine	748
Electronic version of LPIC 1 Certification Bible	749
Guides	749
FAQs	749
HOWTOs	750
Troubleshooting	752
Appendix B: Practice Exams	753
Exam 101	753
Exam 102	761
Exam 101 Answers	768
Exam 102 Answers	772
Appendix C: Objective Mapping	775
Appendix D: Exam Tips	787
Where Can I Take the Test?	787
How Do I Register?	787
How Much Does It Cost?	787
How Long Is the Test?	788
Can I Bring Anything with Me into the Testing Center?	788
How Do I Get My Results?	788
What Happens If I Pass?	788
What Do I Do If I Fail?	788
Can I Retake the Test? How Often?	788
What If I Have a Problem with the Test, or a Question on the Test?	789
What's the Next Step (the Next Exam To Take)?	789
Glossary	791
Index	803
End-User License Agreement	845
CD-ROM Installation Instructions	848

Installing Linux and Getting Started

To begin your studies for the LPI level 1 certification, you need to cover the basics of installing Linux and getting started. This part will prepare you to move on to the next sections in the book.

Chapter 1 provides instructions on installing Red Hat and Debian distributions of Linux. Chapter 2 introduces the Linux shell environment including the bash shell used by default on Linux systems. Chapter 3 covers the various methods and techniques used by the various distributions for installing software. Instructions on upgrading and removing software are included in Chapter 3 as well.

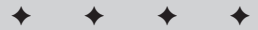


In This Part

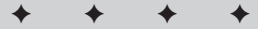
Chapter 1
Installing Linux

Chapter 2
Using the Shell

Chapter 3
Installing Software



Installing Linux



EXAM OBJECTIVES

Exam 102 ♦ General Linux, Part 2

1.1 Hardware & Architecture

- **Configure fundamental system hardware.** Demonstrate a proper understanding of important BIOS settings, set the date and time, ensure IRQ's and I/O addresses are correct for all ports including serial and parallel, make a note of IRQ's and I/O's, be aware of the issues associated with drives larger than 1024 cylinders
- **Setup SCSI and NIC Devices.** Manipulate the SCSI BIOS to detect used and available SCSI ID's, set the SCSI ID to the correct ID number for the boot device and any other devices required, format the SCSI drive - low level with manufacturer's installation tools - and properly partition and system format with Linux fdisk and mke2fs, set up NIC using manufacturer's setup tools setting the I/O and the IRQ as well as the DMA if required
- **Configure Modem, Sound cards.** Ensure devices meet compatibility requirements (particularly that the modem is NOT a win-modem), verify that both the modem and sound card are using unique and correct IRQ's, I/O, and DMA addresses, if the sound card is PnP install and run sndconfig and isapnp, configure modem for outbound dial-up, configure modem for outbound PPP | SLIP | CSLIP connection, set serial port for 115.2 Kbps

EXAM OBJECTIVES (CONTINUED)**2.2 Linux Installation and Package Management**

- **Design hard-disk lay-out.** Design a partitioning scheme for a Linux system, depending on the hardware and system use (number of disks, partition sizes, mount points, kernel location on disk, swap space).

CHAPTER PRE-TEST

1. What does the acronym “GNU” stand for?
2. What types of modems are normally not compatible with Linux?
3. Are you allowed to charge for Linux software?
4. Which SCSI ID is the boot drive normally set to?
5. Can Linux coexist with another operating system on the same computer?
6. Does Linux support new hardware such as AGP?
7. What is the largest swap partition you can create?
8. How many partitions are required to install Linux?
9. What is the default boot loader on most Linux distributions?
10. What is the difference between kernel space and user space?

The first part of this chapter covers the history behind Linux, GNU, and the free software movement. It is important to learn this history, because it has a large influence on the future of Linux. Linux has a longer history than many new users to it realize. There is also a lot of philosophy and belief in Linux that isn't in other commercial products. This comes out in the openness of the software.

The second part of the chapter covers the steps to go through when preparing to install Linux, such as checking hardware configurations and planning the system installation. At the end of the chapter there are two sections that walk you through installing Linux. One covers a Red Hat 7 installation; the other covers Debian 2.2.

The installation is sometimes the trickiest part of running Linux. Thankfully, the installation procedures on distributions today have come a long way compared to those of a few years ago. While you do not have to do as much preparation work as you used to, there are still some things to go over before jumping into the install.

History of Linux and GNU

The UNIX operating system was created at Bell Labs in the early 1970s. It was created to provide a multiuser working environment. UNIX is by far one of the most popular operating systems ever created with a long history that stretches back to the early days of computing. The problem with UNIX is that it has always been expensive and taken large computers to use. Some versions of UNIX have been available for personal computer-type hardware, but the cost has been very prohibitive and the support by multiple vendors has been lacking. These problems are what led to the development of Linux.



Many people associate Linux with Linus Torvalds. Linus initially developed the kernel and is still the technical lead, but the rest of the Linux system was written by other people.

A student at the University of Helsinki in Finland named Linus Torvalds created the Linux kernel. This original announcement was made to `comp.os.minix` newsgroup:

```
From: torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)
Newsgroups: comp.os.minix
Subject: What would you like to see most in minix?
Summary: small poll for my new operating system
Message-ID: <1991Aug25.205708.9541@klaava.Helsinki.FI>
Date: 25 Aug 91 20:57:08 GMT
Organization: University of Helsinki
```

```
Hello everybody out there using minix -
I'm doing a (free) operating system (just a hobby, won't be
big and professional like gnu) for 386(486) AT clones. This
has been brewing since april, and is starting to get ready.
I'd like any feedback on things people like/dislike in
minix, as my OS resembles it somewhat (same physical layout
of the file-system (due to practical reasons) among other
things).
I've currently ported bash(1.08) and gcc(1.40), and things
seem to work. This implies that I'll get something practical
within a few months, and I'd like to know what features most
people would want. Any suggestions are welcome, but I won't
promise I'll implement them :-)
```

Linus (torvalds@kruuna.helsinki.fi)
PS. Yes - it's free of any minix code, and it has a multi-
threaded fs. It is NOT protable (uses 386 task switching
etc), and it probably never will support anything other than
AT-harddisks, as that's all I have :-).

Linus had been working with Minix, another small UNIX operating system available free of charge. Linus began as a school project to create a version of UNIX that could run on IBM-based hardware. While Linus created the kernel and is still very active in the development today, the supporting tools, compilers, utilities, and programs were done by other people as part of the GNU project. You will often see Linux referred to as GNU/Linux for this reason.

**Tip**

GNU stands for “GNU is Not UNIX.” Recursive acronyms are popular in the hacker culture. GNU is pronounced as /g*noo/; the G is not silent.

The GNU General Public License

Linux is released under the GNU General Public License, or GPL. This license is very different from other commercial licenses with which you may be familiar.

**Tip**

A complete copy of the GNU license is available at <http://www.gnu.org/copyleft/gpl.html>.

Linux is deeply rooted in the free software movement, which existed well before Linux was even started. The GNU Project was started to help create software for the good of the community, which is the way software was originally used. Out of this project came the GPL, which is the license that almost all Linux software is released under today.

The GPL was created as a way to keep free software free. The Free Software Foundation is an organization that helps to promote and distribute free software. By using a method known as *copyleft*, the Free Software Foundation and authors use copyright as a defense against others from taking code and claiming it as their own in their own proprietary products. The copyleft method states that no one can place any new restrictions on GPL-covered software when modified or redistributed.

The GPL basically states that the source code to software should be available and that you should be able to modify that software to your needs as you wish. It does not state that the software has to be free of charge, only that the source code must be obtainable. If you make any changes to software covered by the GPL and release it, you are required to release the new source code for others to use.

What Does *Free* Mean?

There is a saying in the free software community: “Free as in speech, not free as in beer.” This sums up what the *free* in free software means. It does not mean that software must be given away for no cost; it means that you can use and modify the software to suit your needs.

To quote the Free Software Foundation’s definition:

0. The freedom to run the program, for any purpose (freedom 0).
1. The freedom to study how the program works, and adapt it to your needs (freedom 1). Access to the source code is a precondition for this.
2. The freedom to redistribute copies so you can help your neighbor (freedom 2).
3. The freedom to improve the program, and release your improvements to the public, so that the whole community benefits. (freedom 3). Access to the source code is a precondition for this.

Even though you can get the source code and modify it as you want, software released under the GPL is not considered to be in the public domain. When software is released to the public domain, no one owns it. This is not the case as the original author owns the copyright under GPL. GPL’d software is also not shareware since money is not required to use the software after distribution.

Many people ask, “How do you make money on open source software when people can just give copies away?” Many companies, such as Red Hat, SuSE, and others, now do business creating and distributing open source software. They make money by providing more convenient methods of obtaining the software, such as packaging Linux distributions, or by providing customization and support services.

Why Use Linux?

There are many different reasons to use Linux. From small network appliances to large back-end servers, Linux is quickly making a name for itself. The best reasons to use it include the following:

- ♦ Linux is multiuser.
- ♦ Linux is multitasking.

- ♦ Linux is stable.
- ♦ Linux has lots of available software.
- ♦ Linux has a wide range of supported hardware.
- ♦ Linux is fast.

Linux is multiuser

Multiuser does not always mean the same thing to different people. The real meaning of a multiuser operating system is one that lets multiple people log in and run processes on it at the same time. The OS distinguishes between the different users to provide security and separation. Examples of multiuser operating systems are Linux, FreeBSD, SunOS, and many other UNIX-like operating systems.

While many operating systems allow you to share resources to many people at once or let more than one person log in at separate times, they are not true multiuser systems. Examples of these are MS-DOS, Windows 9x, Windows NT (except Terminal Server), and MacOS. Many people think that Windows NT is multiuser, but it is not. Only one user can be interactively logged in to the local system at any time. Terminal Server adds the ability to remotely log in to a console on the Windows NT server, but this is a separate product.

Linux is multitasking

Multitasking is commonplace on most modern operating systems. This allows the system to run more than one job at the same time. Older operating systems such as MS-DOS allowed only one application to run at a time. For most server operating systems, this is not an option.

While most current operating systems do multitask, they do not all multitask equally as well. Performance and stability sometimes suffer due to bad implementations or other requirements placed on the design. Fortunately, Linux has a very good multitasking system that is both stable and fast.

Linux is stable

Linux has excellent stability due to its design and modularity. Just because one application crashes does not mean the entire system becomes unusable. Also, making configuration changes or installing new applications does not require you to restart the system, which increases the uptime and makes administration easier.

Linux has lots of available software

With many Linux distributions, you get the main operating system as well as six CDs worth of other software. Software is also freely available with source code that rivals many commercial products. Most Internet servers are run on UNIX with free Linux-compatible software such as Apache, BIND, and Sendmail.

Linux has a wide range of supported hardware

Linux runs on many different platforms and on a lot of older equipment that has been made obsolete by other operating systems a while ago. While an older Pentium or 486 won't handle Windows 98 well, Linux can turn that equipment into a nice network server, firewall, or router.

If you have any non-PC equipment that other vendors may have dropped support for, you may find excellent support in the Linux community. This can quickly bring older equipment back to life and make it useful again. This saves you money by using older equipment and by keeping you from updating equipment as often.

Linux is fast

In most cases Linux is very fast. Many benchmarks have shown it to be faster than commercial operating systems as a network or Internet server, even on specialized hardware. The big reason for high performance is that if a bottleneck or problem is found, it is quickly remedied by the community.



For an example of a Linux benchmark, you can read <http://www.heise.de/ct/english/99/13/186-1/>.

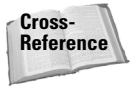
Overview of the Linux Architecture

The Linux architecture comprises two main sections: the *kernel space* and *user space*. As it sounds, the kernel space is where the kernel lives and controls all low level system calls. The user space is responsible for interfacing to the user and running their processes. As you will learn, this separation is very important and is key to stability in Linux.

Kernel space

The kernel space is where all of the system level processes happen. These processes are things that affect the entire system and have to be very stable and well maintained. A problem in kernel space can cause the system to crash.

The main resident in kernel space is, of course, the kernel. The kernel is the piece of software that manages memory allocation for processes and divides up the CPU's time appropriately. The kernel also contains the drivers for the hardware devices installed in the system. The kernel is the core of the Linux operating system.



The Linux kernel is covered in detail in Chapter 13.

User space

The user space manages the *user processes* run by people working on the system. User processes are things such as your e-mail client, Web browser, or word processor. These processes work with the kernel to handle low level functions such as printing to the screen or talking to storage hardware. But, since these functions are not handled in kernel space, a corrupted user application will not bring the entire system down.



Kernel space and user space provide an important separation. Drivers and other things that affect system stability run in kernel space.

Linux Distributions

A Linux *distribution* is a package containing the kernel, GNU utilities and applications, and often an installer. Distributions differ in the software bundled, the installation program, and custom tools. Before distributions, users were literally required to build their system piece by piece. A number of different distributions exist, and some are aimed at different market segments than others. Most distributions are available in more than one form. Some vendors ship separate distributions, with one targeted for the desktop and another for servers. In most cases a freely downloadable version is also available, but it usually does not include all the extra applications. This is more convenient to many users since they do not have to download one or more CDs worth of information. Commercial packages also contain more applications and printed documentation. For new users, the biggest advantage of purchasing a distribution is the included installation support via either e-mail or telephone. The following sections discuss the major distributions that are available.

Red Hat

One of the most popular distributions, especially in the server market, is Red Hat. Several server vendors currently ship Red Hat preinstalled on their systems. They also offer one of the most popular home distributions. More information is available at <http://www.redhat.com>.

Mandrake

Originally Mandrake was just Red Hat with some additions, but it is now a totally separate distribution. Mandrake is currently the most popular home distribution in the U.S., according to retail sales. The key to Mandrake's success is its ease of installation, cutting edge configuration tools, and the fact that Mandrake is often one of the first to put new features in production. Because of its roots, Mandrake's file and directory layout is very close to Red Hat's. This is useful if you want to be comfortable with Red Hat but would like the features of Mandrake. More information is available at <http://www.linux-mandrake.com>.

Debian

The maintainers of the Debian distribution truly believe in the idea of free software. They provide a totally free distribution that has no software encumbered by non-free licenses. Debian's installation is not as easy as most other distributions, but its more advanced software installation and upgrade system means that you will probably never have to run through another setup or full system upgrade. Debian is usually recommended for experienced Linux users. More information is available at <http://www.debian.org>.

SuSE

The most popular distribution in Europe, SuSE is now garnering a large following in the U.S. and the rest of world. SuSE offers an easy to use installation system, as well as very good tools to help maintain the system after installation. SuSE has also made some excellent partnerships in the server world and is a good choice in a large enterprise environment. More information is available at <http://www.suse.com>.

Slackware

Slackware was one of the first distributions, and one that many longtime Linux users started on. It is still popular today with advanced users. Unlike the other distributions it has no real packaging system for software management. More information is available at <http://www.slackware.com>.

Caldera

Usually considered the easiest distribution to install and use, Caldera is popular in the business desktop market. The problem with this simplicity is that some functions are hidden. Advanced users may not like this distribution, but it works well for casual users or those moving from Microsoft Windows. More information is available at <http://www.caldera.com>.

Turbolinux

Turbolinux is very popular in Asia and is quickly gaining recognition in the rest of the world for high performance cluster installs where multiple servers work together. Turbolinux complements the standard Linux distribution with some custom work to make clustering easier. More information is available at <http://www.turbolinux.com>.

Which distribution is for you? That depends on the intended function of the system. Mandrake and SuSE have excellent install programs and well done default desktop configurations that make them very good choices for desktops. Red Hat has gained a lot of support from server vendors, which makes it a good choice for file and service servers. Turbolinux is very suited for large cluster installs as they have created some of the best tools for this purpose. Power users and those that want the most control of their system usually use Debian. The configuration and management of distributions can vary greatly. The startup procedures and configuration files may differ greatly and could cause confusion during troubleshooting if you are working with an unfamiliar distribution.

Preparing Hardware

Objective

1.1 Hardware & Architecture

- **Configure fundamental system hardware.** Demonstrate a proper understanding of important BIOS settings, set the date and time, ensure IRQ's and I/O addresses are correct for all ports including serial and parallel, make a note of IRQ's and I/O's, be aware of the issues associated with drives larger than 1024 cylinders
- **Setup SCSI and NIC Devices.** Manipulate the SCSI BIOS to detect used and available SCSI ID's, set the SCSI ID to the correct ID number for the boot device and any other devices required, format the SCSI drive - low level with manufacturer's installation tools - and properly partition and system format with Linux fdisk and mke2fs, set up NIC using manufacturer's setup tools setting the I/O and the IRQ as well as the DMA if required

While having the latest and greatest hardware is often an advantage on other operating systems, it can sometimes be a problem in Linux. Since most Linux drivers are created by users and not companies, the most popular hardware usually has the best support. The problem arises when companies do not release the technical specifications for their products. Either the hardware cannot be supported, or the driver must be reverse engineered, which takes much longer and usually lacks some features.



The newest Hardware-HOWTO that shows supported Linux hardware is available at <http://www.linuxdoc.org/HOWTO/Hardware-HOWTO.html>. Also check with your distribution vendor as they may provide drivers not listed in the HOWTO.

CPU requirements

Linux currently supports many different platforms, including the following:

- ♦ Intel 386, 486, and all Pentium class CPUs
- ♦ x86 clones such as Cyrix and AMD K6 and Athlons
- ♦ Digital/Compaq Alpha
- ♦ MIPS
- ♦ PowerPC
- ♦ Sun SPARC
- ♦ Motorola 68000
- ♦ Handheld PC architectures

Note that not all of these platforms are as well supported as others, and there may not be a commercially supported version of every platform. If you have a 386 or 486SX but no math coprocessor, you will need to enable Math Coprocessor emulation in the kernel. Most distribution kernels already have this enabled, since it is used only if required.

Almost any motherboard that a supported CPU runs on should work. Linux supports ISA, EISA, MCA, VESA, PCI, and AGP buses.

Memory requirements

Linux can be streamlined to run on a very thin system. The suggested minimum is 8MB of RAM, but if you plan to run X Window, you will need more. The current 2.2 Linux kernel supports up to 2GB of RAM on 32-bit x86 systems. Some systems require a kernel parameter at boot to recognize more than 64MB.

Linux supports virtual memory, which lets you use disk space as additional RAM in cases of exceptional need. This process is called *swapping*, and it is not advised to do this all the time as the perceived speed of your system will degrade heavily.

Hard disk controller requirements

Linux supports almost all types of hard disk controllers, except for the ultra new or proprietary controllers. Almost any standard IDE, MFM, RLL, or ESDI controller should work with no problem. If a new faster standard is introduced, it may take a little time before a driver shows up in the kernel for it.

Linux handles SCSI controllers nicely as well. Major SCSI vendors such as Adaptec, Ultrastor, Future Domain, Western Digital, or others should pose no problems for you. Most SCSI controllers have a BIOS that can be accessed during boot and used

to configure the controller. Before installing a new Linux system go into the BIOS and make sure to note the SCSI IDs of all SCSI peripherals. The order of the SCSI IDs will dictate how the devices are named in Linux. Special note should be given to SCSI IDs 0 and 1, as they are normally the devices that the system tries to boot from in order.

If you are installing a new hard drive into a SCSI system you may be required to low level format the disk before it can be used. Check your controller's documentation to see if this is the case for your situation. To low level format the disk, use the tool supplied in the SCSI controller's BIOS.



Many new SCSI controllers can be configured to boot from any SCSI ID, but make sure you know which ID is configured as the boot device.

Hard disk space requirements

The amount of disk space that you need depends heavily on the role of the system. Consider the use of your Linux system in deciding how much disk space should be allocated. A small network services server can be run in a few hundred megabytes of disk space, or less. A network file server may require many gigabytes of space, depending on the data to be shared. Linux can be booted from a floppy disk if needed, and many people have done some excellent network appliance setups doing just that. Full installs of distributions may take anywhere from 300MB to 1.3GB, depending on the extra applications installed.

Video requirements

The video requirements of Linux vary depending on if you need to use a text interface only or if you also want to use a graphics interface. For a text terminal almost any video and monitor combination works fine. X Window adds some complexity to the configuration due to the various different video hardware available, but the XFree group is doing great work in supporting graphics chipsets.

BIOS settings

Linux gets very little information from the BIOS of the system. The hard disk settings are actually not used, and the information is obtained from the disks and controllers. The one piece of information obtained from the BIOS is the time and date. Before installing Linux, check the time settings in the BIOS to make sure they are correct. Also, note whether the time is set to local time or to UTC/GMT.

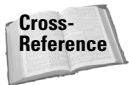
Peripherals and other hardware

Linux supports many other types of hardware including pointing devices, CD-ROMs, modems, and tape backups.

Pointing devices

Mice are usually used in X Window and sometimes in text consoles using such tools as General Purpose Mouse (GPM), which provides copy and paste support. Linux natively supports all standard serial, PS/2, and bus mice. There is even support for the new wheel mice and those with extra buttons. No matter which you choose, it is a good idea to get a mouse with at least three buttons.

Some distributions support USB pointing devices with the version 2.2 kernels. USB is officially being added in the 2.4 version kernels. The new 2.4 kernel was released during the writing of this book, but is still very new and should not show up in new Linux distribution for some time. When it does show up, the main difference users will notice is increased hardware support and new features, but no usage changes will occur.



The Linux kernel is covered in detail in Chapter 13.

CD/DVD drives

Current CD/DVD-ROM drives use either IDE/ATAPI or SCSI. Both of these are widely supported under Linux. Many older proprietary CD-ROM drives are also supported in the kernel. As usual, check the Hardware-HOWTO or your distribution to make sure your proprietary CD-ROM is covered.

Almost all CD-R/RW drives are supported as well. Linux has a very nice setup for the new IDE CD-R drives, because it uses a SCSI emulation layer. This way an application does not need to know which interface a CD-R/RW drive uses nor does the application need to be modified if it already supports SCSI drives.

Printers

Linux supports both local and network printers. Many printers now have Linux drivers; however, there is less support for printers than for other hardware so be sure to check for support before buying a printer. If possible, it is best to buy a printer that supports PostScript.

Tape drives and removable media

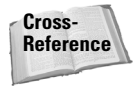
The best tape drive and removable media support are for those devices that use the SCSI bus. Lower end tape drives using floppy controllers are also supported.

Modems

Objective**1.1 Hardware & Architecture**

- **Configure Modem, Sound cards.** Ensure devices meet compatibility requirements (particularly that the modem is NOT a win-modem), verify that both the modem and sound card are using unique and correct IRQ's, I/O, and DMA addresses, if the sound card is PnP install and run `sndconfig` and `isapnp`, configure modem for outbound dial-up, configure modem for outbound PPP | SLIP | CSLIP connection, set serial port for 115.2 Kbps

Linux supports a full range of internal and external modems. However, there is one very large caveat: make sure that your modem is not a Winmodem. Winmodems use software to do the actual modulation instead of hardware. This allows manufacturers to make very inexpensive modems at the cost of some CPU usage. The problem is that there is very little Linux support for these modems. Normally the only modem driver available, if there is one, is a binary driver that may work with only one or two kernel versions.



Configuring the modem to dial out using PPP is covered in Chapter 15.

Network cards

Objective**1.1 Hardware & Architecture**

- **Setup SCSI and NIC Devices.** Manipulate the SCSI BIOS to detect used and available SCSI ID's, set the SCSI ID to the correct ID number for the boot device and any other devices required, format the SCSI drive - low level with manufacturer's installation tools - and properly partition and system format with Linux `fdisk` and `mke2fs`, set up NIC using manufacturer's setup tools setting the I/O and the IRQ as well as the DMA if required

Network support is a definite strong point of Linux. Almost every network card from the new Gigabit Ethernet to "ancient" ARCnet has a driver. The big differences are speed and ease of setup.

If you have the option, get a PCI network card as they normally require no extra configuration other than loading the driver; plus, they use very little CPU overhead. If your network card vendor is not listed as having a driver, be sure to actually check the chipset on the card as many manufacturers use the same chipset.

Configuring the IRQ and DMA settings is covered in the section "Resolving Conflicts and Configuring Plug-and-Play Devices" later in this chapter.

Sound cards

Objective

1.1 Hardware & Architecture

- **Configure Modem, Sound cards.** Ensure devices meet compatibility requirements (particularly that the modem is NOT a win-modem), verify that both the modem and sound card are using unique and correct IRQ's, I/O, and DMA addresses, if the sound card is PnP install and run `sndconfig` and `isapnp`, configure modem for outbound dial-up, configure modem for outbound PPP | SLIP | CSLIP connection, set serial port for 115.2 Kbps

Sound cards can be very easy or very difficult depending on the type of driver required. Some cards have a driver included in the kernel and work right off. Others require you to work with the Plug-and-Play tools in Linux to set them up properly. If a driver is not included in the kernel, several other sound driver libraries are available. Most current distributions include a sound setup tool to ease the configuration.

The most common sound card configuration tool is `sndconfig`, created by Red Hat but used on many distributions. Figure 1-1 shows an example of `sndconfig`. By default, `sndconfig` probes the system and tries to determine the type of sound card installed. If it cannot detect the card automatically, it will prompt you to choose one from a list. The probing can be disabled with the `--noprobe` parameter, and the automatic configuration of the card can be disabled with the `--noautoconfig` parameter.



Figure 1-1: `sndconfig` example



Tip

The ALSA project adds support for many different sound cards. Their URL is <http://www.alsa-project.org/>.

Configuring the IRQ and DMA settings is covered in the following section.

Resolving Conflicts and Configuring Plug-and-Play Hardware

Hardware conflicts happen when two devices share the same configuration address. New PCI devices can share addresses, but older hardware cannot. To resolve a conflict, you may need to change a hardware jumper, or with Plug-and-Play devices, you may need to adjust the software configuration.

Hardware addresses

Hardware devices in a computer use several configuration options. These configuration addresses tell the system how to talk to the device. The usual address settings are the following:

- ♦ DMA
- ♦ IRQ
- ♦ I/O address

Direct memory access (DMA) channels are used when devices need to access memory directly, without going through the CPU. Devices should not share DMA channels. Interrupt requests (IRQs) are used by devices to alert the CPU when the devices need some action taken by the CPU. Most older devices had to have a unique IRQ, while newer PCI devices can share IRQs. An I/O address is a memory address used by the device to communicate with the rest of the system. Each device must have a unique I/O address.

Older hardware had the configuration addresses set manually by jumpers on the physical board or by software configuration tools. Current hardware is configured automatically by the system's BIOS or by the operating system.

Most hardware can be set to any free IRQ or I/O address. The usual exceptions are serial and parallel ports. They need to be at a certain address to be used as a standard port. Serial ports COM1 and COM2 are set to IRQ 4 and 3 respectively. Parallel ports LPT1 and LPT2 are set to IRQ 7 and 5 respectively.

Viewing configuration addresses

The first step in eliminating a hardware conflict is to see which resources are currently being used and which ones are free. Move any conflicting pieces of hardware to a free resource.

Listing interrupts

To see which IRQs are used and free on a system, you can view the `/proc/interrupts` file. For example:

```
debian:~$cat /proc/interrupts
          CPU0
 0:  510082305          XT-PIC  timer
 1:    278104          XT-PIC  keyboard
 2:         0          XT-PIC  cascade
 9: 3014717670          XT-PIC  acpi, EMU10K1, eth0
10:  3253300          XT-PIC  usb-ohci
13:         0          XT-PIC  fpu
14:  387511089          XT-PIC  ide0
15:  3844545          XT-PIC  ide1
NMI:         0
ERR:         0
```

As you can see from the example, the IDE controllers are on IRQ 14 and 15. The sound card, EMU10K1, and the network card, eth0, are both on IRQ 9. The USB controller, usb-ohci, is on IRQ 10. The IRQ numbers not listed are considered free and unused.

Listing I/O ports

To see which I/O ports are being used on a system, you can view the `/proc/ioports` file. For example:

```
debian:~$cat /proc/ioports
0000-001f : dma1
0020-003f : pic1
0040-005f : timer
0060-006f : keyboard
0080-008f : dma page reg
00a0-00bf : pic2
00c0-00df : dma2
00f0-00ff : fpu
0170-0177 : ide1
01f0-01f7 : ide0
0376-0376 : ide1
03c0-03df : vga+
03f6-03f6 : ide0
03f8-03ff : serial(set)
0cf8-0cff : PCI conf1
5000-5003 : acpi
5004-5005 : acpi
5008-500b : acpi
5020-5023 : acpi
cb00-cb0f : Advanced Micro Devices [AMD] AMD-756 [Viper] IDE
          cb00-cb07 : ide0
          cb08-cb0f : ide1
```

```
e000-ffff : PCI Bus #01
f800-f8ff : Lite-On Communications Inc LNE100TX
  f800-f8ff : eth0
ff80-ff9f : Creative Labs SB Live! EMU10000
  ff80-ff9f : EMU10K1
ffe4-ffe7 : Advanced Micro Devices [AMD] AMD-751 [Irongate]
System Controller
fff0-fff7 : Creative Labs SB Live!
```

As with IRQs, any range not listed is considered free, so it can be used by another device.

Listing DMA channels

To view the used and free DMA channels on the system you can view the `/proc/dma` file. For example:

```
debian:~$ cat /proc/dma
4: cascade
```

This system has only one DMA channel being used.

Configuring Plug-and-Play devices

Plug-and-Play devices have their configuration options set via software instead of using hardware jumpers. This makes it easier to adjust them when a hardware conflict occurs. The first step in configuring Plug-and-Play hardware is to probe the system for supported hardware and to create a configuration file. This is done with the following command:

```
debian:~$pnpdump > /etc/isapnp.conf
```

Note that this works only for ISA devices. The default configuration file has all options commented out. You will need to go through the file and uncomment the selected IRQ, DMA, and I/O port addresses that you want to use. Before uncommenting the lines, check the system for free addresses, as shown in the previous section. For example, to set up a sound card you uncomment the IRQ, DMA, and I/O port lines, as in the following:

```
(CONFIGURE CTL0028/1530224 (LD 0
#   ANSI string -->Audio<--

# Multiple choice time, choose one only !

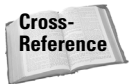
#   Start dependent functions: priority preferred
#   IRQ 5.
#   High true, edge sensitive interrupt (by default)
(INT 0 (IRQ 5 (MODE +E)))
```

```

#       First DMA channel 1.
#       8 bit DMA only
#       Logical device is a bus master
#       DMA may execute in count by byte mode
#       DMA may not execute in count by word mode
#       DMA channel speed in compatible mode
(DMA 0 (CHANNEL 1))
#       Next DMA channel 5.
#       16 bit DMA only
#       Logical device is a bus master
#       DMA may not execute in count by byte mode
#       DMA may execute in count by word mode
#       DMA channel speed in compatible mode
#       Logical device decodes 16 bit IO address lines
#       Minimum IO base address 0x0220
#       Maximum IO base address 0x0220
#       IO base alignment 1 bytes
#       Number of IO addresses required: 16
(IO 0 (SIZE 16) (BASE 0x0220))
(NAME "CTL0028/1530224[0]{Audio"           })
(ACT Y)

```

You must also be sure to uncomment the final (ACT Y) line so that the card is activated. For each device detected by `pnpdump` you need to manually set the configuration options. Once you are satisfied with your settings, you should run the `isapnp` command or reboot the system. Once `isapnp` has been run, you can load the driver for the configured hardware.



Hardware drivers and modules are covered in more detail in Chapter 13.

Partitioning Schemes



2.2 Linux Installation and Package Management

- **Design hard-disk layout.** Design a partitioning scheme for a Linux system, depending on the hardware and system use (number of disks, partition sizes, mount points, kernel location on disk, swap space).

Unlike most other operating system installs you may be familiar with, Linux installs are not always to one large partition. *Partitions* are used to section up disk space in to smaller logical blocks. In the UNIX world, partitions are sometimes known as volumes. Splitting up the drive(s) into multiple partitions gives you flexibility, stability, and easier maintenance.

When building a server, you most likely want to put the following directories, the standard directories that all Linux systems will have, on separate volumes.



You can find most of these directories covered in more detail in Chapter 6.

- ♦ /
- ♦ /home
- ♦ /opt
- ♦ /tmp
- ♦ /usr
- ♦ /usr/local
- ♦ /var

The reasons for the separation are performance, data integrity, backup, and security. Most of the benefits require each volume to be on a separate physical disk. If one disk fails the entire file system is not taken out with it. Restores can be done much faster since only one directory structure is affected. Many directories contain binary files that rarely change, or should not change. By mounting these volumes as read-only, you can provide some system security so that a hacker cannot put a Trojan horse utility on the system and gain more information. Directories such as `/tmp` and `/var` contain temporary files and logs. Either intentionally or accidentally a user can quickly consume space on these volumes. By mounting these as separate volumes you can stop any denial-of-service attacks or accidents so that the entire system does not run out of space. A hard disk can have four partitions. There can be up to four primary partitions or three primary and one extended partition that can be divided up into logical drives.

On a workstation you may not want to go through the effort to have multiple volumes. A good suggestion is to put the `/home` directory on another volume. This lets you upgrade or change your Linux distribution without having to worry about moving your home directories to another place while the system is redone.

Linux supports the use of virtual memory through the use of swap partitions. You can have up to eight swap partitions for a combined total of 4GB on an x86 system. No matter how much RAM your system has, you should always have some swap space for extraordinary circumstances.



Before kernel v2.2, swap partitions on x86 systems were limited to 128MB in size. The limit is now 2GB.

If you have a hard disk with greater than 8GB capacity, you may want to use a separate partition for the `/boot` directory. This directory holds the kernel and other boot files. Some versions of the Linux boot loaders cannot access a kernel that is outside the first 1024 cylinders on a disk. By putting the `/boot` partition at the beginning of the drive you can be assured of not having a problem when accessing the kernel at boot. This problem shows itself most often in cases of dual booting Linux along with another operating system that is on the first partition.

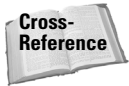


More information on designing your disk layout can be found in Chapter 5.

Using fdisk

The original partitioning tool in Linux is `fdisk`. It is a console-based tool that has been around for years, but is still widely used. Start `fdisk` by typing the following (where *x* is the letter of the disk you want to partition):

```
fdisk /dev/hdx
```



Disk and partition naming are discussed in detail in Chapter 5.

You must specify which disk device to work with when starting `fdisk`. After starting `fdisk` you will see this prompt:

```
The number of cylinders for this disk is set to 8190.
There is nothing wrong with that, but this is larger than
1024, and could in certain setups cause problems with:
1) software that runs at boot time (e.g., LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)
```

Command (m for help):

Typing **m** for help will give you the options shown in Table 1-1.

Table 1-1
fdisk Options

<i>Command</i>	<i>Function</i>
a	Toggle a bootable flag
b	Edit BSD bootlabel
c	Toggle the DOS compatibility flag
d	Delete a partition
l	List known partition types
m	Print the help screen
n	Add a new partition
o	Create a new empty DOS partition table
p	Print the partition table
q	Quit without saving changes

Continued

Table 1-1 (continued)

<i>Command</i>	<i>Function</i>
s	Create a new empty Sun disk label
t	Change a partition's system ID
u	Change display/entry units between sectors and cylinders
v	Verify the partition table
w	Write partition table and exit
x	Expert functions

The *partition type* defines whether the partition is for data or swap space. Type 83 is for data and type 82 is for swap.

STEPS: To create a new partition follow these steps:

1. Enter **n** to create a new partition. You will see the following:

```
Command action
  e   extended
  p   primary partition (1-4)
```

2. Enter **p** to create a primary partition or **e** to create an extended partition. You will be prompted for the partition number as follows:

```
Partition number (1-4):
```

3. Enter the partition number you want to add. You must choose a number that is not already taken.

4. Enter the first cylinder number or accept the default. For example:

```
First cylinder (1101-2343, default 1101):
```

5. Enter the last cylinder or the size in the format of +size, +sizeM (megabytes), or +sizeK (kilobytes).

Optionally, to change the partition's type, follow these steps:

1. Choose **t** to change the type.

2. Enter the partition number to change.

```
Partition number (1-5):
```

3. Enter the hex code of the new type or enter **L** to list them.

```
Hex code (type L to list codes): 82
```

Using Disk Druid

Disk Druid is a graphical tool used during the install of Red Hat. It is very easy to use and lets you set up your partitions quickly. Disk Druid not only lets you create partitions, but it also lets you configure where these partitions will be placed in the file system. With UNIX, a partition is not accessed as a separate drive with a drive letter; it is mounted as a directory. This directory is known as a *mount point*. Figure 1-2 shows the main Disk Druid screen.

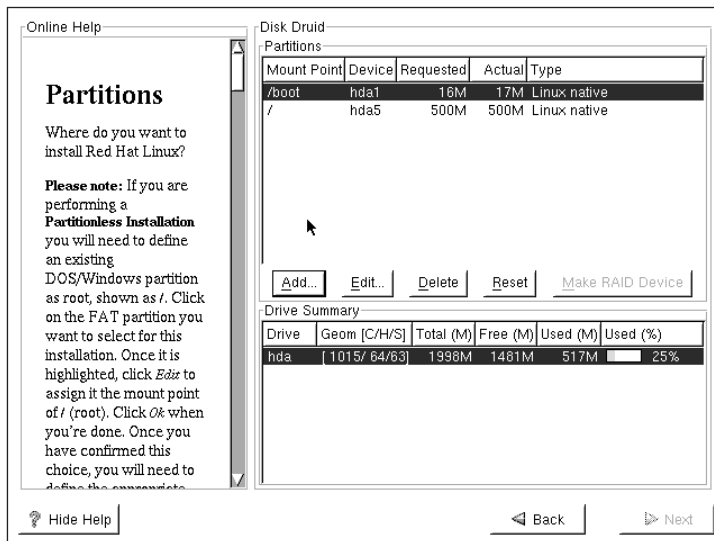


Figure 1-2: Disk Druid

STEPS: To create a partition in Disk Druid follow these steps:

1. Click the Add button. The Add Partition window opens (see Figure 1-3).
2. Enter the mount point of the volume.
3. Enter the size of the new partition.
4. Select the partition type to use.
5. Select the drive to put the new partition on.
6. Click OK.

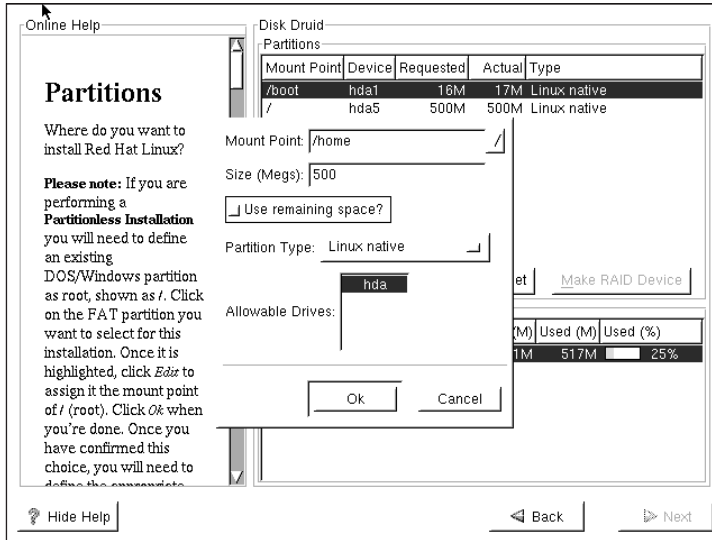


Figure 1-3: Add Partition window

To remove a partition, highlight the unneeded partition and click Delete.

Using cfdisk

`cfdisk` is a newer text based console tool, which is easier to use than the original `fdisk`. Start `cfdisk` by typing the following (where *x* is the letter of the drive you want to partition):

```
cfdisk /dev/hdx
```

The main screen should look similar to Figure 1-4.

```

cfdisk 2.10f

Disk Drive: /dev/hda
Size: 2096962560 bytes
Heads: 64 Sectors per Track: 63 Cylinders: 1015

-----
Name      Flags      Part Type  FS Type      [Label]      Size (MB)
-----
hda1     Boot      Primary   Linux ext2    16.52
hda2     Primary   Linux ext2    999.17
hda3     Primary   Linux swap    200.25
          Pri/Log   Free Space    879.43

[Bootable] [ Delete ] [ Help ] [Maximize] [ Print ]
[ Quit ]  [ Type ]  [ Units ] [ Write ]

Toggle bootable flag of the current partition

```

Figure 1-4: cfdisk menu

Use the vertical arrow keys to choose the partition you want to work with and the horizontal keys to choose the action item at the bottom of the screen.

STEPS: To create a new partition, follow these steps:

1. Highlight the section of free space to be partitioned.
2. Choose New from the menu.
3. You will be given a prompt for the type of partition to create. Choose whether you want to create a primary or extended partition.
4. Enter the size of the partition to be created.
5. Choose whether or not to create the partition and the beginning or end of the free space.
6. Optionally, choose Type from the menu and change the partition type.

To delete a partition, simply highlight the appropriate partition and choose Delete from the menu. Make sure the partition you want to boot to is set correctly. Highlight the boot partition and choose Boot. Finally, no changes are done until the Write option is selected.

Boot Managers

How you will boot your Linux system can be very simple or very complicated. Linux is capable of coexisting with most other operating systems. This is called a *multiboot* setup. With this configuration a boot manager is used to select which operating system is booted.

Most Linux distributions ship with the LILO (Linux LOader). If you do not want to use LILO, you can also use other third-party boot loaders or the Windows NT Boot Manager.



LILO is covered in detail in Chapter 8.

Installing Linux

This section details installing two different Linux distributions. For this we have chosen Red Hat 7.0 and Debian 2.2, because they provide good practice for the exams. You aren't required to know the installations for the exam. However, you will need to know about package management and file locations, and these two installs give you exposure to different types of configurations and packaging systems. We recommend that you install more than one distribution to get a feel for different ways to do things, and you will be asked about these differences on the exam.

Red Hat installation

Red Hat has a very easy to use installation. If your video card is supported by X Window you should be able to use the GUI install process. If not, Red Hat offers a text-based install with all of the power of the GUI, but without the pretty pictures.

During the setup there are several consoles that can be used for troubleshooting or manual tasks. A *console* is another virtual terminal that you can access to see troubleshooting information or to do tasks outside of the setup application. To change consoles from the GUI install you press CTRL-ALT-#, where # is the number of the console you want to switch to. For text mode, use ALT-#. The extra consoles are shown in Table 1-2.

Table 1-2
Red Hat Install Consoles

<i>Console Number</i>	<i>Function Key</i>	<i>Contents</i>
1	F1	Text-based installation procedure
2	F2	Shell prompt

<i>Console Number</i>	<i>Function Key</i>	<i>Contents</i>
3	F3	Messages from the installation program
4	F4	System-related messages
5	F5	Other messages
7	F7	GUI installation

Setting up the installer

The installation begins by booting off either the Red Hat CD-ROM or boot floppies. The easiest and fastest method is the CD-ROM, if your computer has a BIOS new enough to do that. If you put in the CD-ROM and it does not boot off it the first time, check your BIOS setup for an option. Normally this option will be listed as the boot order for the system. If it is set to a hard disk or the floppy first, change it to the CD-ROM option.



To create the boot disks you can use the Raw Write (`rawwrite.exe`) tool, which is available on the Red Hat site under `dosutils`. The available boot disks are `boot.img`, `bootnet.img`, `drivers.img` and `pcmcia.img`. The `boot.img` is for CD-ROM installs, and `bootnet.img` is for network installs. You should make the `drivers.img` disk, which contains network card drivers, unless you are using a notebook, for which the `pcmcia.img` is used.

Figure 1-5 shows the Red Hat 7.0 Welcome screen, which is shown when the system boots. Table 1-3 lists the options available at the Welcome screen.

```

Welcome to Red Hat Linux 7.0!

o To install or upgrade a system running Red Hat Linux 3.0.3
  or later in graphical mode, press the <ENTER> key.

o To install or upgrade a system running Red Hat Linux 3.0.3
  or later in text mode, type: text <ENTER>.

o To enable expert mode, type: expert <ENTER>. Press <F3> for
  more information about expert mode.

o To enable rescue mode, type: linux rescue <ENTER>. Press <F5>
  for more information about rescue mode.

o If you have a driver disk, type: linux dd <ENTER>.

o Use the function keys listed below for more information.

[F1-Main] [F2-General] [F3-Expert] [F4-Kernel] [F5-Rescue]
boot: _

```

Figure 1-5: Red Hat Welcome screen

Table 1-3
Welcome Screen Options

<i>Key</i>	<i>Function</i>
<ENTER>	Start the installation.
text <ENTER>	Start the installation in text mode.
expert <ENTER>	Enter expert mode. This is used to manually configure hardware and kernel modules.
linux rescue	Start rescue mode. Useful to recover a non-booting system.
linux dd	Use if you received a driver disk for hardware not supported during the default setup.
F1	Main screen.
F2	General help.
F3	Expert mode help.
F4	Optional kernel parameters. Most used to specify the amount of RAM in a system if the kernel does not automatically detect it.
F5	Rescue mode help.

For most installations you will just hit the Enter key to begin. Some systems may require a kernel parameter to correctly detect the amount of memory in the system if it is more than 64MB.

Language Selection

The next screen shown is the Language Selection, as shown in Figure 1-6. As you can see, Red Hat supports many different languages. This will be the language used during and after setup.

Keyboard Configuration

Next is the Keyboard Configuration screen, as shown in Figure 1-7. Simply select the type of keyboard you have from the list and test it in the text box at the bottom. Dead keys can also be configured. These are keys that do nothing when pressed, but change the way the next character is formed. These keys are normally used to add accents to characters for different languages.

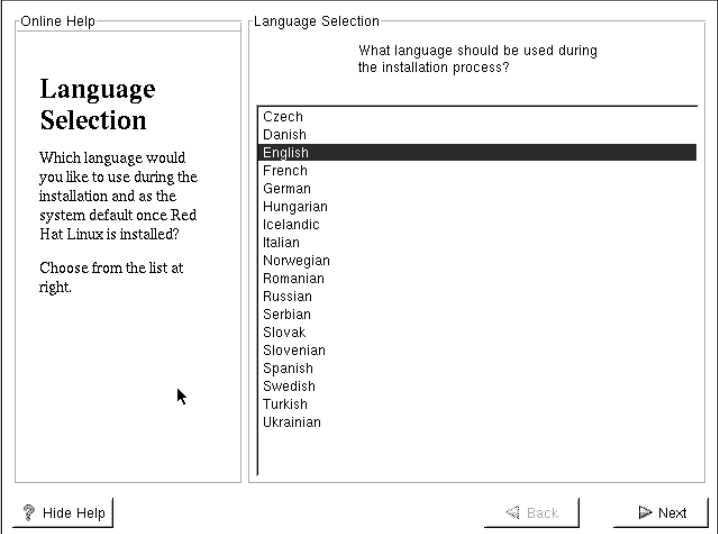


Figure 1-6: Language Selection screen

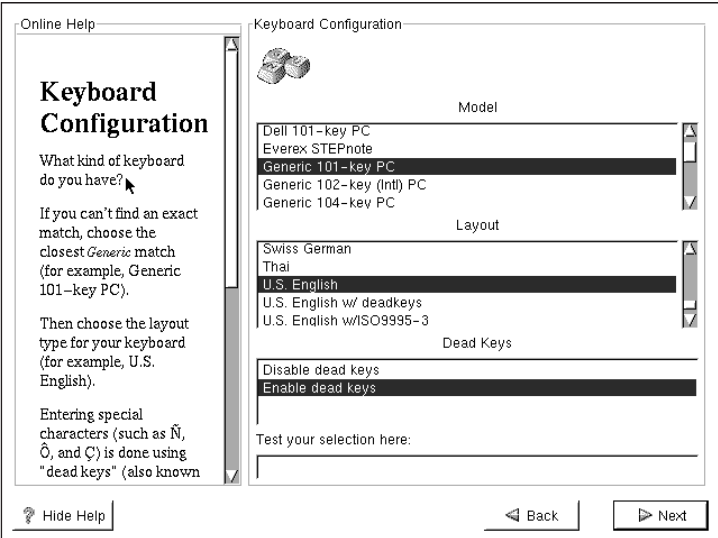


Figure 1-7: Keyboard Configuration screen

Mouse Configuration

The next screen allows you to set the correct mouse, as shown in Figure 1-8. Be sure to set the correct port if necessary. If you have only a two-button mouse, then be sure to enable three-button emulation. This lets you emulate a third button by pressing both buttons at the same time.

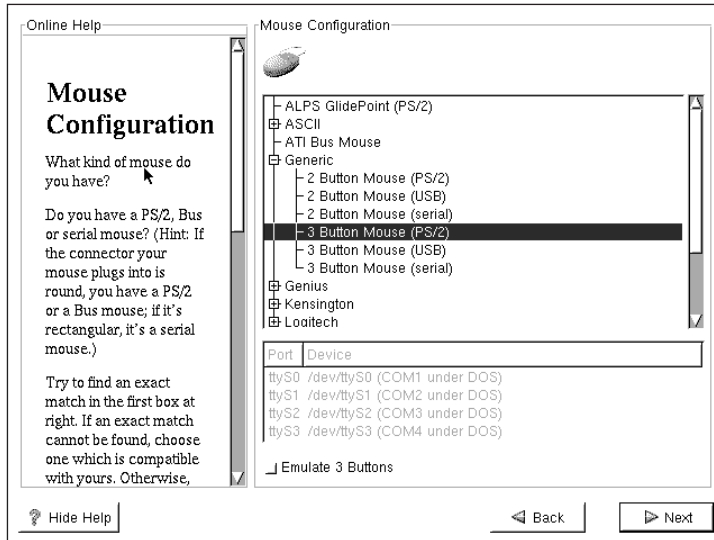


Figure 1-8: Mouse Configuration screen

Welcome to Red Hat Linux

The installer next displays a Welcome screen explaining where to go for more information if needed. This begins the system installation.

Configuring the system

Now the real system configuration and installation begins. The first major choice is whether to perform an upgrade or a new install, as shown in Figure 1-9. If this is a new install you have several options to choose from, as follows:

- ♦ **Workstation** — This option installs workstation-related packages, such as graphic applications, network client tools, and other end user software.
- ♦ **Server System** — This option installs server-related packages such as Apache, FTP, and so on.
- ♦ **Custom System** — The Custom installation option lets you select the packages.



For this lab we will choose the Workstation installation option, since the Server installation does not install the X Window System.

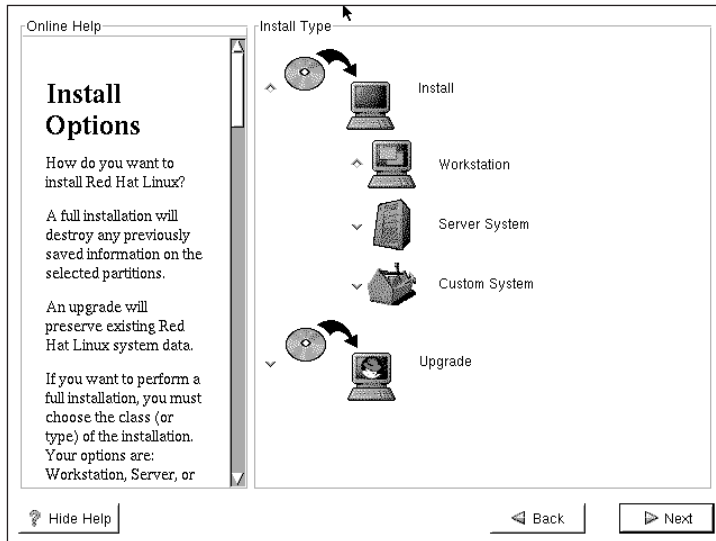


Figure 1-9: Install Type

Disk partitioning



Be careful when partitioning your disk! If you have any data you want to keep, be sure to do a backup before partitioning, just in case.

The next task is to partition the hard disk. If there are no other operating systems on the computer, you can let the installation partition the disks for you. If you want to dual boot, you can manually partition the drive with Disk Druid or `fdisk`.



Disk Druid and `fdisk` are covered earlier in the chapter.

Figure 1-10 shows the Automatic Partitioning screen.

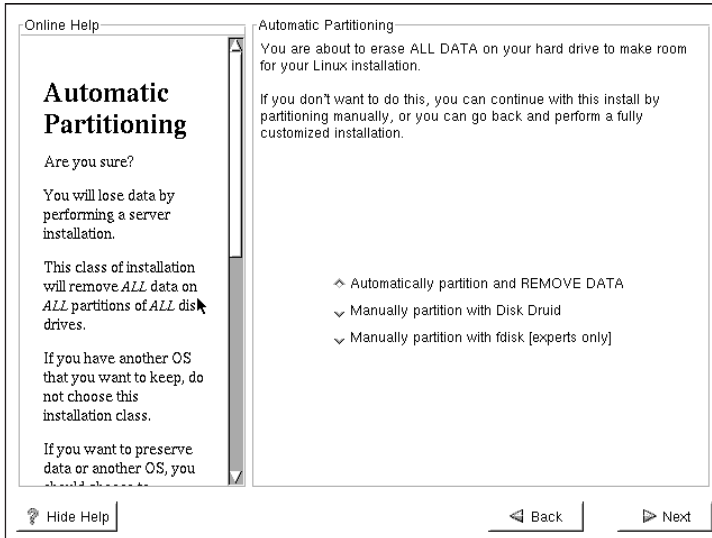


Figure 1-10: Automatic Partitioning screen

Network Configuration

If a network card was detected in the system the installer now prompts you for configuration information, as shown in Figure 1-11. Enter the following configuration information:

- ♦ **Configure using DHCP**—Have the network interface automatically configured from a DHCP server.
- ♦ **Activate on boot**—Have the network interface automatically started on boot up.
- ♦ **IP Address**—Unique IP address for this interface.
- ♦ **Netmask**—Subnet mask for your network.
- ♦ **Network**—Network number of your network.
- ♦ **Broadcast**—Broadcast address for your network.
- ♦ **Hostname**—Name of this computer, not including the domain name. If none is specified, the name “localhost” will be used.
- ♦ **Gateway**—Default network gateway or router.
- ♦ **Primary DNS**—Preferred DNS server.
- ♦ **Secondary DNS**—Backup DNS server.
- ♦ **Tertiary DNS**—Second backup DNS server.

If you are setting up a lab to practice on, you can use the following network configuration:

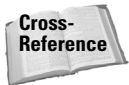
IP address: 192.168.1.1

Netmask: 255.255.255.0

Broadcast address: 192.168.1.255

Network address: 192.168.1.0

The gateway and DNS settings can be left blank.



Network configuration is covered in more detail in Chapter 15.

Figure 1-11 shows the Network Configuration screen.

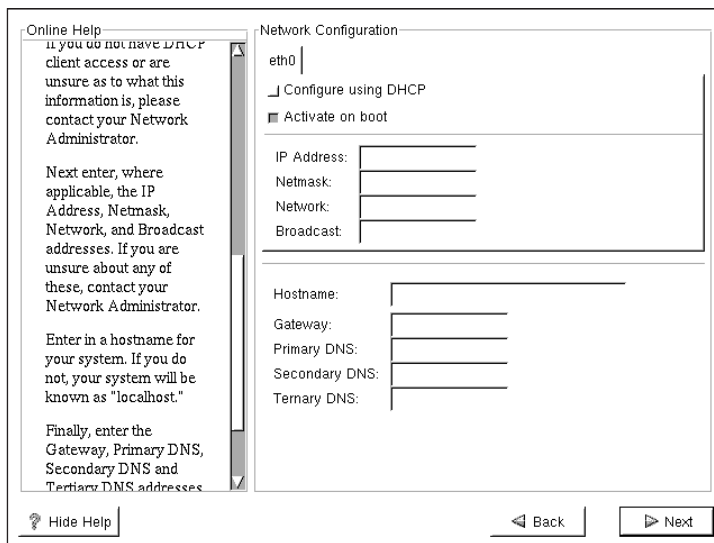


Figure 1-11: Network Configuration screen

Time Zone Selection

The clock configuration is done on the Time Zone Selection screen, shown in Figure 1-12. Choose your correct time zone and choose whether or not your system clock is set to UTC. UTC, or Coordinated Universal Time, is a worldwide standard for denoting time. It was formerly known as Greenwich Mean Time (GMT).

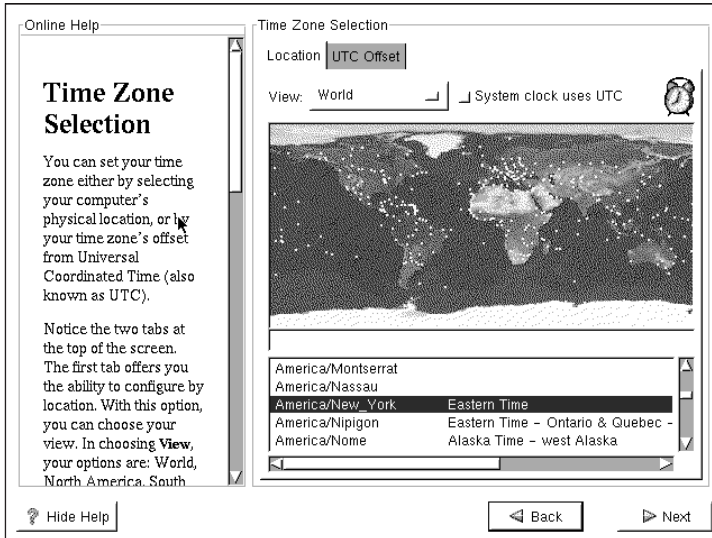


Figure 1-12: Time Zone Selection screen

Account Configuration

The Account Configuration screen lets you specify the root password and create any other normal user accounts you want. It is always a good idea to create a non-root account to use most of the time. This way you will not accidentally delete files or make system changes. For each account you configure the user name, password, and optional full name. Figure 1-13 shows the Account Configuration screen.

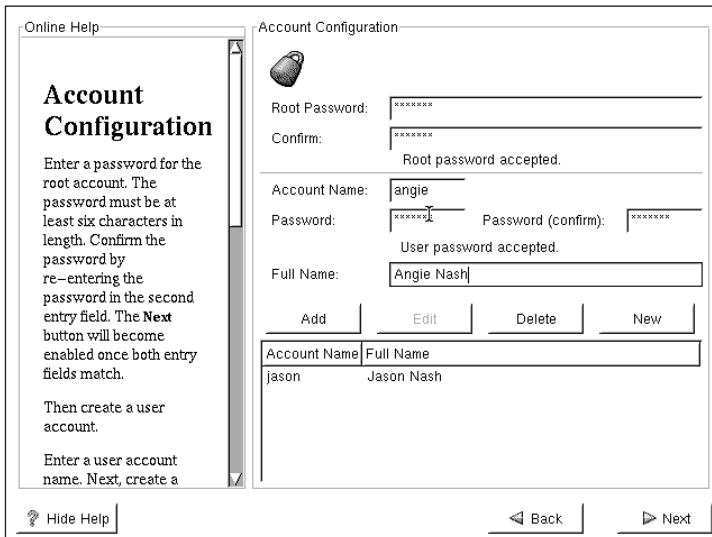


Figure 1-13: Account Configuration screen

Package Group Selection

The screen shown in Figure 1-14 lets you select package groups to install to add functionality to the server. A *package* contains all the files for a software application. You should install only packages you intend to use so that you do not waste disk space or have to worry about maintaining the security of extra software. You can also select the individual package option to tell the installer other packages to add.

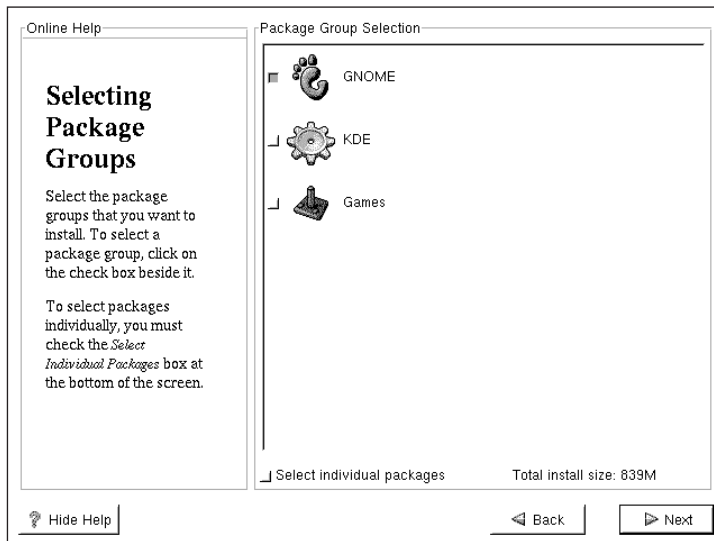


Figure 1-14: Package Group Selection screen

Individual Package Selection

The Individual Package Selection screen lets you select separate packages to install. Clicking on a package will bring up a description. Feel free to look through the available packages to see if any interest you. They are all optional, so none are required. To install the package, select the Select Package For Installation option. Figure 1-15 shows the Individual Package Selection screen.

Unresolved Dependencies

The Unresolved Dependencies screen, shown in Figure 1-16, checks any packages you manually install for needed dependencies. If any are found it gives you the option to install the other packages. This keeps you from accidentally installing any unusable software.

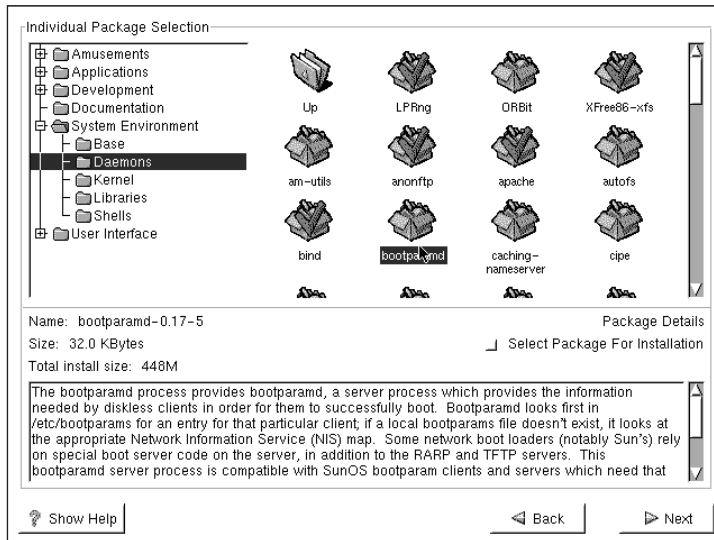


Figure 1-15: Individual Package Selection

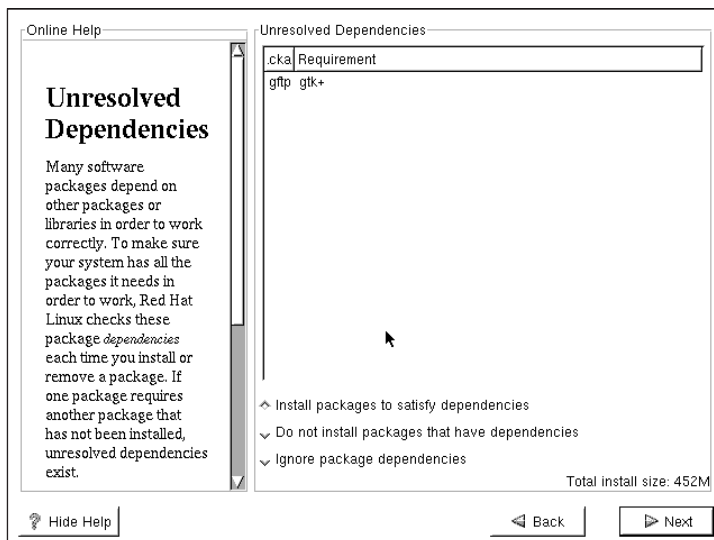


Figure 1-16: Unresolved Dependencies screen

X Configuration

X Window is installed by default whenever you choose to do the Workstation install. The first step in configuring X Window is to select your monitor. If your monitor is not listed you can manually put in the frequency settings available from your manual or vendor's Web site. Figure 1-17 shows the X Monitor Configuration screen.



Configuring the X Window System is discussed in detail in Chapter 9.

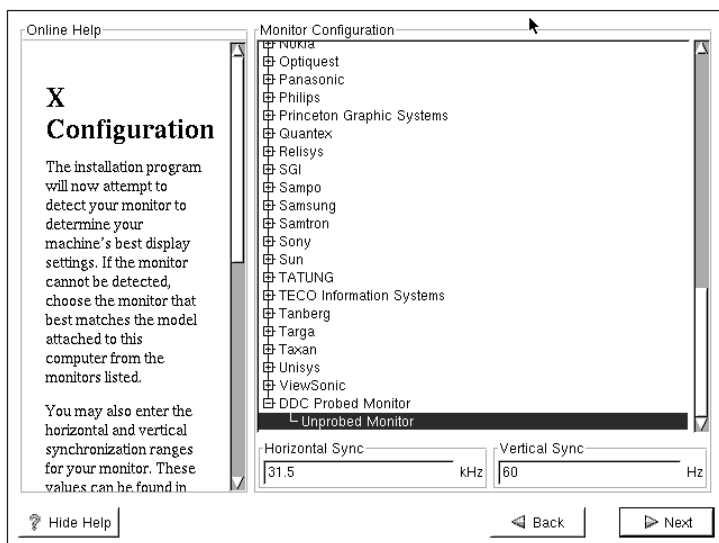


Figure 1-17: X Monitor Configuration screen

The next option allows you to select your video card, as shown in Figure 1-18. In most cases the install program can detect your video card for you. If it is incorrect, you can manually choose the card or options.

An option to Use Graphical Login is also presented on this screen. This has the system boot to a GUI login screen instead of the normal text login. If you make incorrect choices during the X Window setup you may have trouble booting after setup. We recommend that you leave this disabled until you can test X Window and then enable it after setup.

About to Install

The installation program will now show the About to Install screen. This is the warning right before all the changes are made to the system. If you reboot now, you can forget everything done up to this point and the system will still be as it was. By clicking Next you start the installation of Red Hat. Depending on your installation media, number of packages chosen, and computer speed, this may take some time.

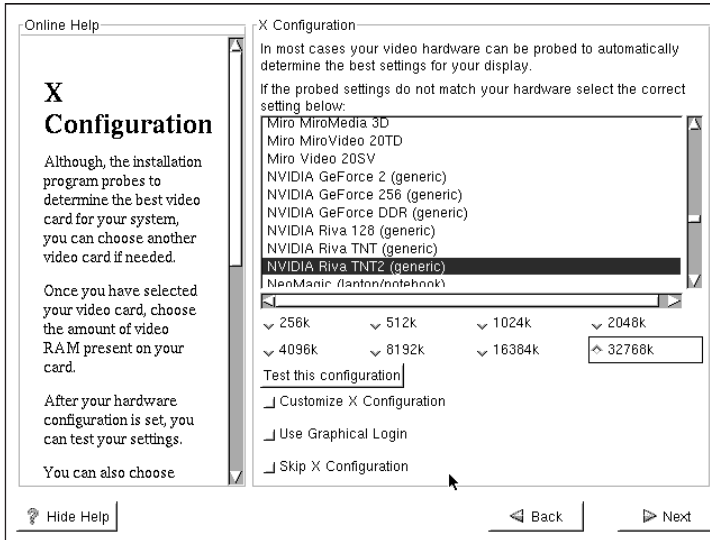


Figure 1-18: X video card configuration

The installation

The next step is for the setup program to copy over all files selected during install. Figure 1-19 shows the copying process.

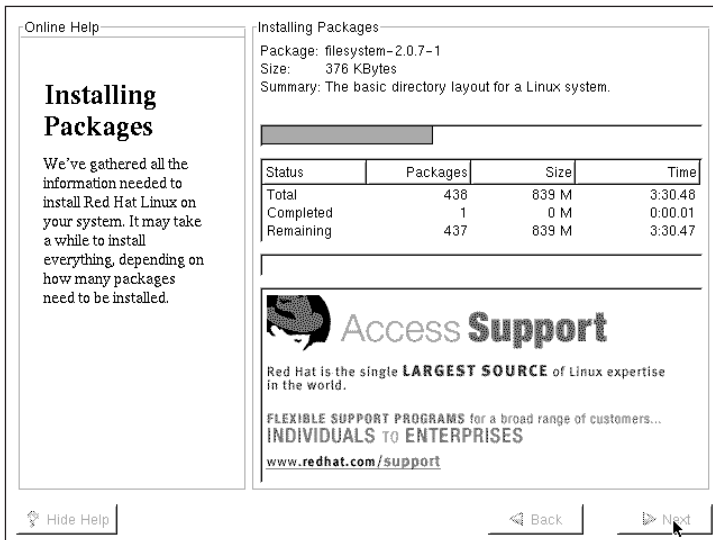


Figure 1-19: Installing packages

After all of the packages have been installed you will be prompted to create a boot disk. The boot disk can save you and your system should Linux not boot due to a kernel or LILO problem. We highly recommended you create the disk.

That's it! Reboot and enjoy the new Red Hat installation.

Debian installation

Debian's installation is not as smooth as Red Hat's, but Debian makes up for it in other ways. For example, once you understand Debian's packaging system, which the setup uses for install, you can customize the installation sources greatly. This way you can do an install from many different media sources at once.

Booting

To start the Debian installation you will boot from either an install CD or several floppy disks. The easiest way is of course the CD-ROM. When you start the installation you will see the Welcome to Debian screen shown in Figure 1-20.



Debian provides a number of boot disks for many different configurations on their FTP site. The same Raw Write (rawwrite.exe) is used to create these as you used to create the Red Hat disks.

```

Welcome to Debian GNU/Linux 2.2!

This is the Debian Rescue disk. Keep it once you have installed your system,
as you can boot from it to repair the system on your hard disk if that ever
becomes necessary (press <F3> for details).

On most systems, you can go ahead and press <ENTER> to begin installation.
You will probably want to try doing that before you try anything else. If
you run into trouble or if you already have questions, press <F1> for
quick installation help.

WARNING: You should completely back up all of your hard disks before
proceeding. The installation procedure can completely and irreversibly
erase them! If you haven't made backups yet, remove the rescue disk
from the drive and press <RESET> or <Control-Alt-Del> to get back to
your old system.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law. For copyright information, press <F10>.

This disk uses Linux 2.2.17
(from kernel-image-2.2.17_2.2.17pre6-1)

Press <F1> for help, or <ENTER> to boot.
boot:
```

Figure 1-20: Welcome to Debian

You may be surprised to see that the screen says this is a rescue disk. After install you can use your boot CDs or disks to repair the system installation. The available options on this screen are as follows:

- ♦ <ENTER> — Starts the installation
- ♦ F1 — Installation help
- ♦ F3 — Boot methods
- ♦ F10 — Copyright information

The F3 function shows the possible boot methods for recovery or installation. The options are as follows:

- ♦ **linux** — Start the default installation.
- ♦ **ramdisk0 & ramdisk1** — Start the installation and read the ramdisk from disk 0 or 1. This is used in recovery and loads a file system into a *ramdisk*, which is a piece of memory that looks like a normal disk.
- ♦ **floppy0 & floppy1** — Start the installation and mount the disk in the first or second floppy drive.
- ♦ **rescue** — Boot and mount any volume as the root file system. You must give an option to specify the root volume to mount (for example, `rescue root=/dev/hda1`).

Base system configuration and installation

Once the Linux kernel has loaded you will be presented with the release notes for this version. During the setup you have access to other consoles for information. Table 1-4 shows the consoles.

Table 1-4
Debian Install Consoles

<i>Console Number</i>	<i>Purpose</i>
Alt-1	Setup program
Alt-2	Shell prompt
Alt-3	Debug messages
Alt-4	Kernel messages

The release notes will tell you which kernel the installation was built using, as well as any other important notes about this installation. The Debian installation follows a certain flow, but gives you the option of doing things out of order if needed in

certain situations. Figure 1-21 shows the configuration options. The next step is given as the default whenever you are brought back to the menu. In this case, it is asking you to configure your keyboard. Choose the type of keyboard that your system uses.

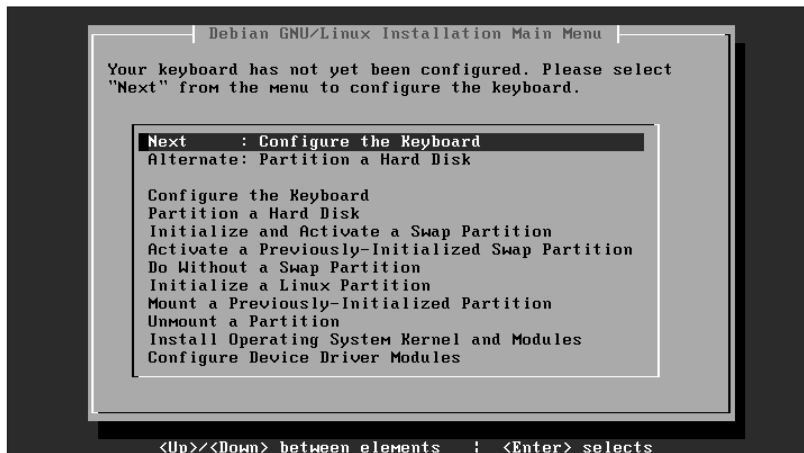


Figure 1-21: Main Menu

Disk setup

The next step is to partition and set up the data volumes and swap partitions, as shown in Figure 1-22.

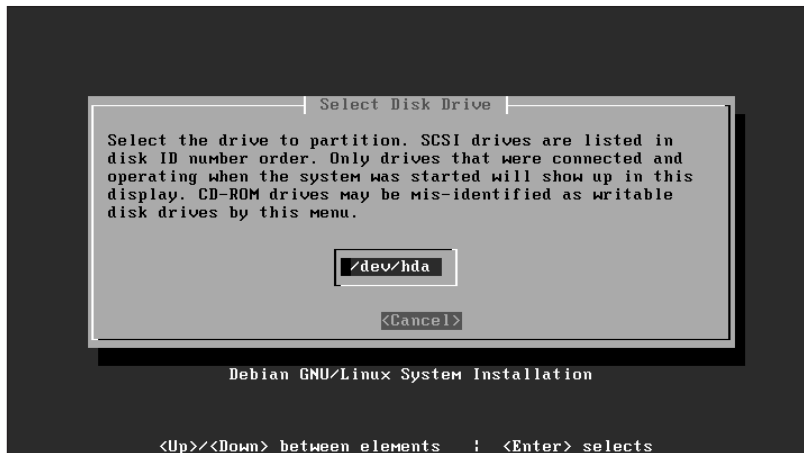


Figure 1-22: Disk drive selection

Choose the hard disk you want to partition. Depending on your system configuration you may need to heed the warning in the LILO Limitations screen, as shown in Figure 1-23. As discussed previously in this chapter, some systems cannot boot if the Linux kernel is stored beyond the 8GB barrier on a hard disk, or past cylinder number 1023. New versions of LILO allow you to boot from any point on the drive, if your system has a BIOS that is new enough. If your system does not support this, or if you are just unsure, make a `/boot` partition below the 8GB barrier.

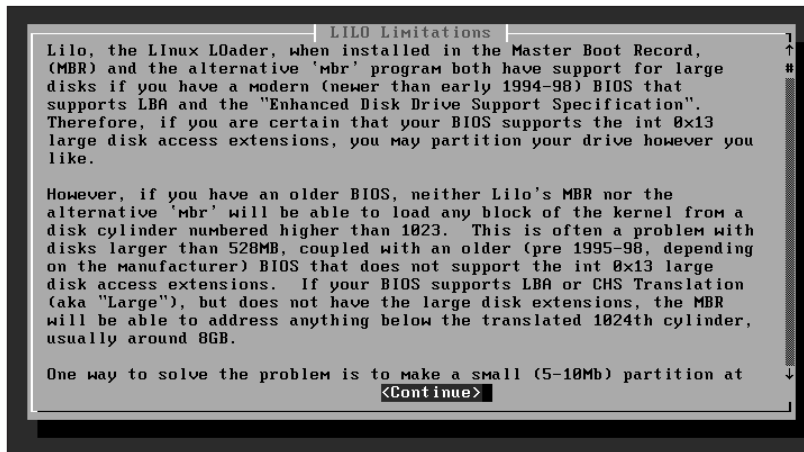


Figure 1-23: LILo Limitations screen

Debian uses `cfdisk` to handle partitioning. Set up your partitions as you see fit. For lab purposes it is probably a good idea to split up your volumes so you get experience with more than just single volume systems. A good method for splitting the disk is to put `/home`, `/tmp`, and `/var` on separate partitions. You can also put `/usr` on a separate partition, but be sure to give it at least a gigabyte of disk space so you have room for software installations later. Don't forget to make a swap partition!

Once `cfdisk` is completed, you will be prompted to initialize the swap partition(s), as shown in Figure 1-24.

Any time you initialize a new partition in the Debian install you will be prompted to perform a bad block check. This writes some data to every sector on the disk and then reads it to make sure it is the same. If the read data is different, then the sector is marked as bad. Modern disk controllers can now handle bad blocks automatically, and as this process can take a very long time, you can normally skip this test. But, if the hardware is older or questionable, it does not do any harm to run the test. After the test you can confirm the disk initialization, which will destroy any data. Figure 1-25 shows the bad block test screen.

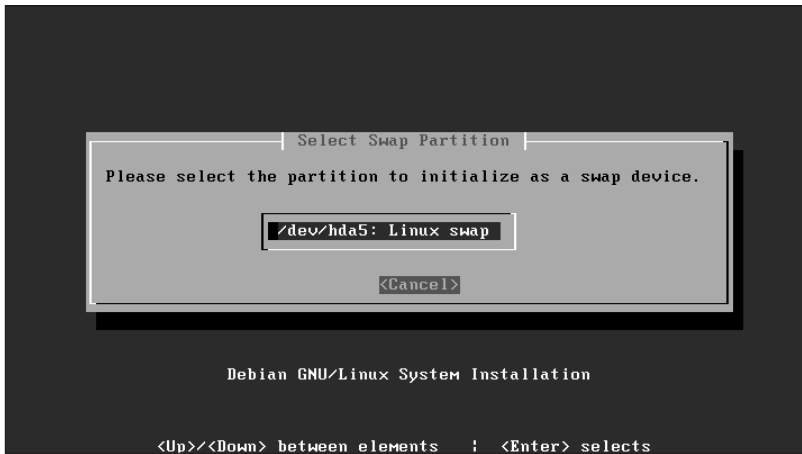


Figure 1-24: Swap selection



Figure 1-25: Bad block test

The next several screens walk you through initializing the data volumes. You have the option of disabling some features of the ext2 file system used, if you need to access them from a Linux kernel older than 2.2. Normally this is not needed. Figure 1-26 shows the Pre-2.2 Kernel Compatibility screen.

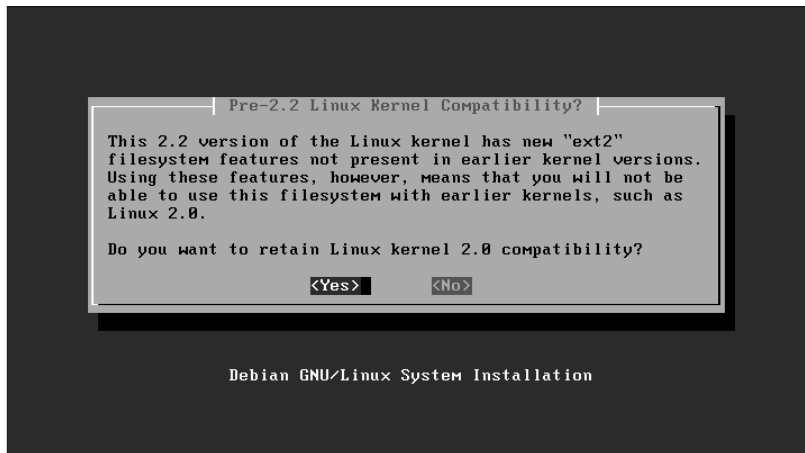


Figure 1-26: Pre-2.2 Kernel Compatibility screen

After initialization you will be prompted to mount this new file system as root ("/"). This screen is shown in Figure 1-27. If this is the correct partition to be root, choose Yes. If not, choose No. Once back at the main menu you can go through and initialize any other partitions you need. Once you have mounted a partition as root, you will be prompted to mount the others in standard locations as you initialize them.



Figure 1-27: Mount point selection

Kernel and module installation

The next phase of the installation is to install the Linux kernel and device driver modules needed by the system. You are prompted for the location from which to install the kernel and modules. This is useful if you need a special kernel or modules not included on the CD-ROM.

Unless you are doing anything special, such as installing from an archive you created yourself that uses nonstandard paths, you should just accept the defaults for the archive path screens. Next, choose the correct modules for your system, as shown in Figure 1-28. It is important to pick the correct drivers so your system can be configured correctly. If you do not load a network driver here, you will have to manually configure the network after the install. You will also be given a chance to add any module options, if needed.

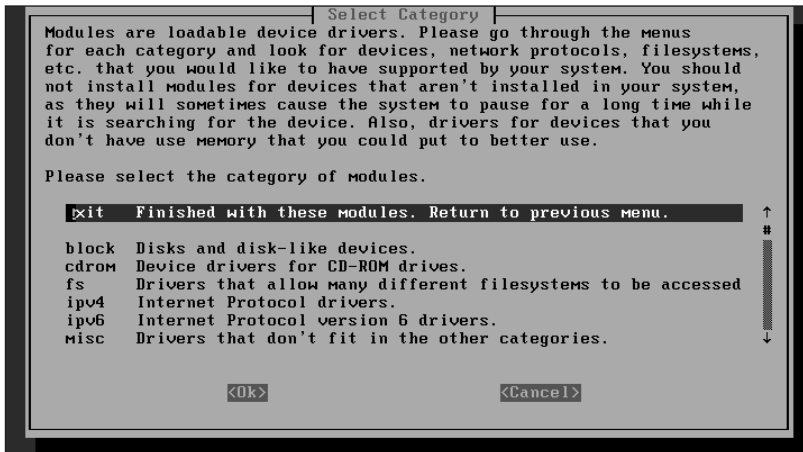


Figure 1-28: Module selection

Network configuration

If you loaded a driver for a network card, the next step in the installation is to configure your network. The first option is your computer's hostname. This is the hostname only, not combined with the domain name. For example, if my computer is known as `norbert.the-nashes.net`, the hostname would just be `norbert`. Figure 1-29 shows the hostname configuration screen.

If your network interface will be configured automatically using DHCP or BOOTP, enable that here. If you choose automatic configuration, you may skip to the next section. Figure 1-30 shows the Automatic Network Configuration screen.

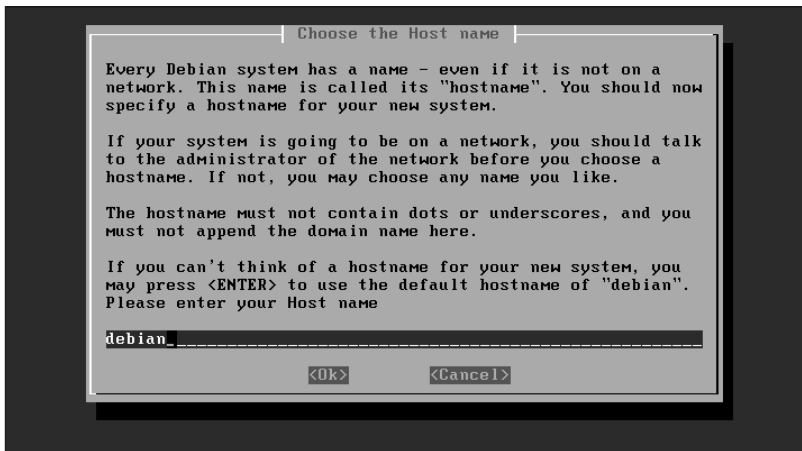


Figure 1-29: Hostname configuration

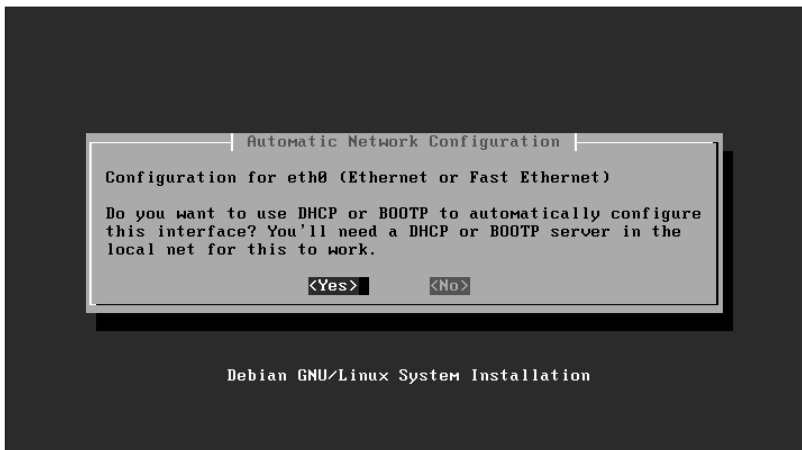


Figure 1-30: Automatic Network Configuration screen

The next several screens prompt you for basic host information such as IP address, netmask, default gateway, domain name, and DNS servers. Multiple DNS servers should be entered in a list with spaces separating them, as shown in Figure 1-31. For example, you would enter them as follows: 1.2.3.4 1.2.3.5 1.2.3.6

If you are setting up a lab to practice on, you can use the following network configuration:

IP address: 192.168.1.1

Netmask: 255.255.255.0

Broadcast address: 192.168.1.255

Network address: 192.168.1.0

The gateway and DNS settings can be left blank.

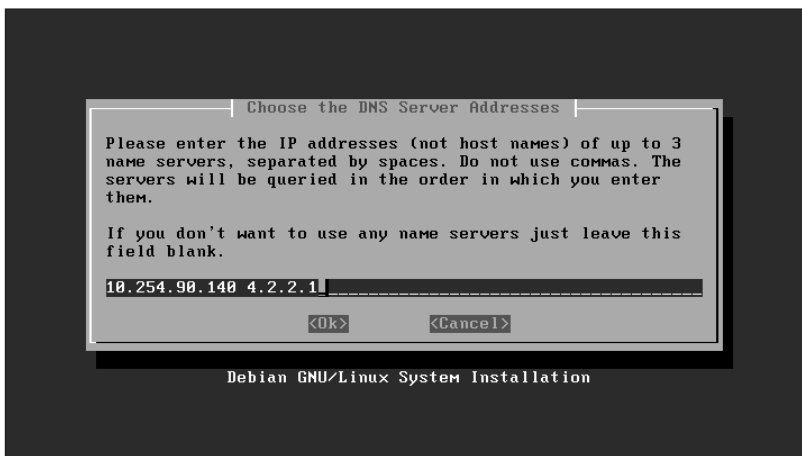


Figure 1-31: DNS server configuration

Base system installation

The next section installs the base system. These are configuration files, basic system tools, directory structures, and anything else that every Linux system requires. You are prompted for the source location since it may be different from the kernel and module source location if you used third-party install disks to install the kernel and modules. Again, in most cases you can take the defaults for the archive paths. Figure 1-32 shows the base system installation.

The base system configuration begins with the time zone selection screen, as shown in Figure 1-33. You will also be prompted as to whether the system clock is set to UTC.

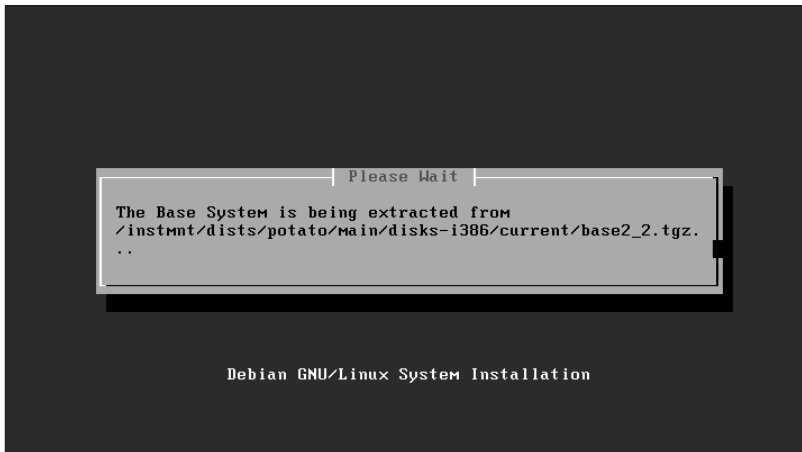


Figure 1-32: Base system installation



Figure 1-33: Time zone selection

By default, Debian installs the LILO boot manager. The installer needs to know where you want LILO installed. In most cases it is installed to the master boot record (MBR) of the primary booting drive, but if you use a third-party boot manager, it may need to be installed to the first sector of the partition. Check your third-party boot manager's documentation for more information. Figure 1-34 shows the LILO installation screen.

The final step before the reboot is to create a floppy disk. This is useful if LILO is not installed correctly and the system cannot boot. A floppy disk will keep you from having to go through all of the above steps again.

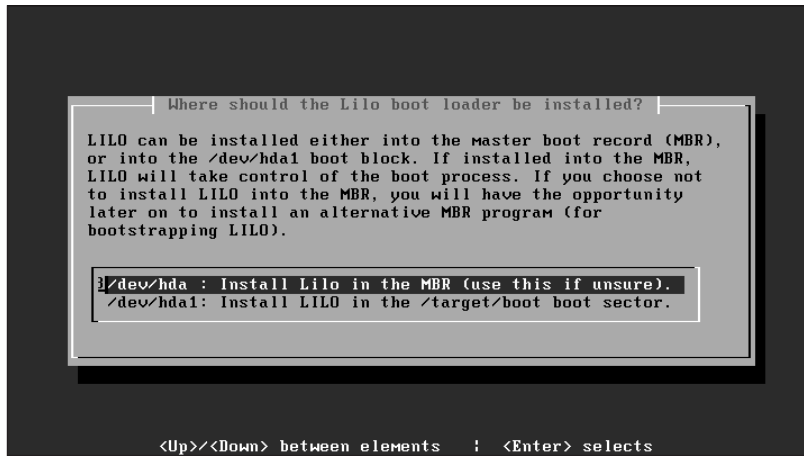


Figure 1-34: LILO installation

If you booted from the CD-ROM to start the installation, remove it now.

System configuration

After the reboot you are dropped in to the system configuration. The first option is whether or not to enable MD5 passwords, as shown in Figure 1-35. Unless you are sharing password files with other systems, you should enable MD5 as it is more secure. Another security option is whether to enable shadow passwords or not. All current software should work with shadow passwords, so there is little reason not to use it.



MD5 and shadow passwords are covered in Chapter 17.

Figure 1-35 shows the MD5 option screen.

The next several screens will prompt you for the root password, as shown in Figure 1-36.

Since it is not a good idea to always log in as root, you may want to create a normal user account to use. Figure 1-37 shows the new user creation screen.

If your system does not have any PCMCIA devices, you can remove the PCMCIA packages. Usually removing the PCMCIA packages is a good idea unless you are using a notebook or desktop PC with a PCMCIA adapter card. The Debian installer can get the packages it needs for installation from the Internet. If you need to configure PPP on this system to let it get on the Internet and get packages, you can configure it here. It is much faster to install from a CD-ROM and update any newer packages after the install.

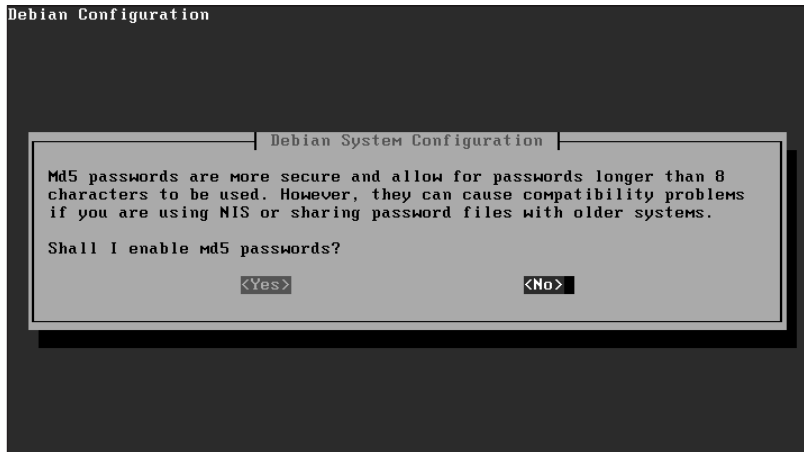


Figure 1-35: MD5 options



Figure 1-36: Root password

Package selection

Debian uses its own advanced package management for installation. This provides for a high level of customization and lets you install packages from several different locations at one time. The first option in this section asks whether or not you have any other Debian package CDs, as shown in Figure 1-38.

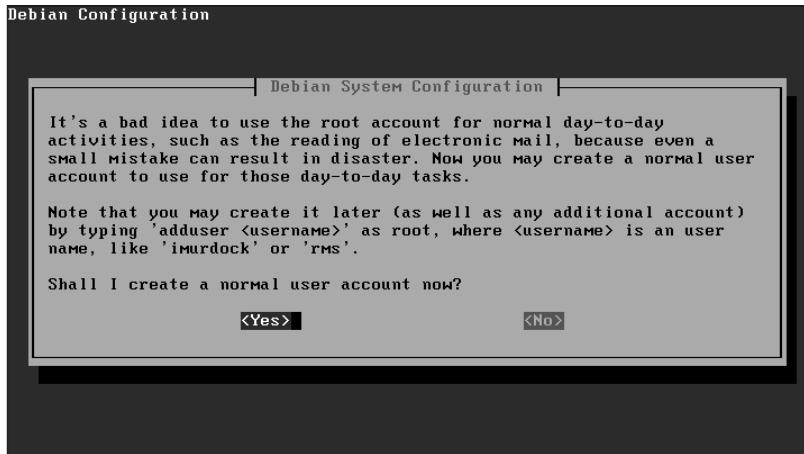


Figure 1-37: New user creation

If you have any other sources of packages, such as FTP or HTTP, you can add other apt sources here. Debian supports two different modes of installation. In most cases you should use the Simple installation method, as shown in Figure 1-39. The simple method lets you install software by groups for certain tasks. If you choose the advanced method you must wade through hundreds of packages selecting them individually.

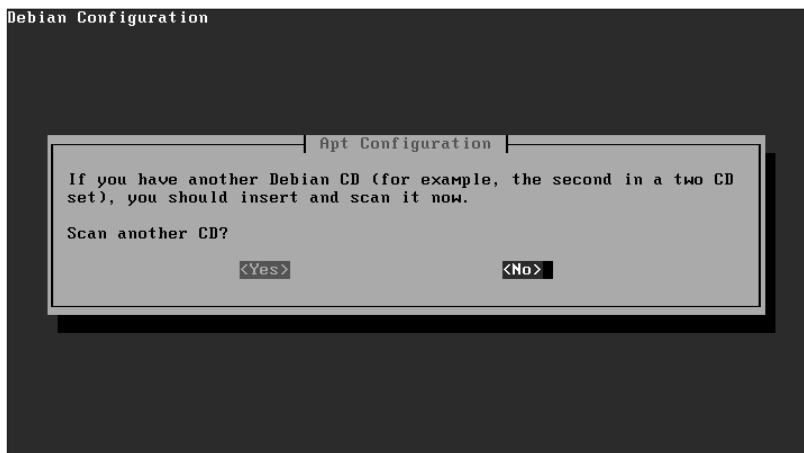


Figure 1-38: Debian CD selection



Configuring apt sources is covered in Chapter 3.

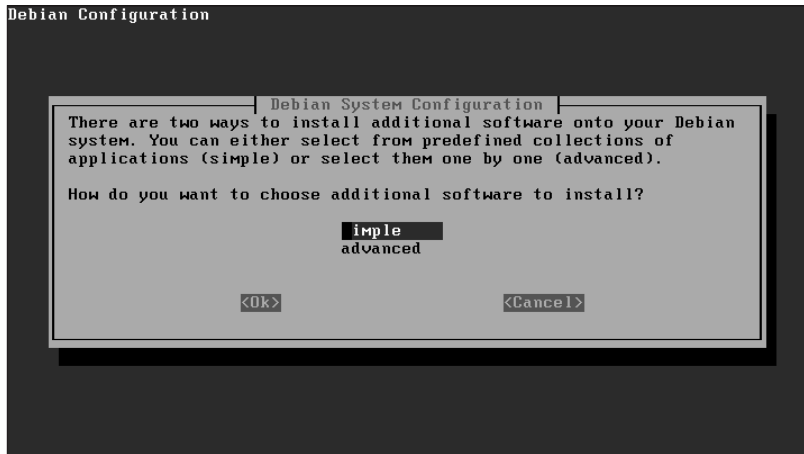


Figure 1-39: Installation type

The next screen presents you with “tasks” that install packages as a group. Choose any groups you will need, as shown in Figure 1-40. For studying, it is recommended to install the GNOME and KDE desktops, X Window, and development packages.

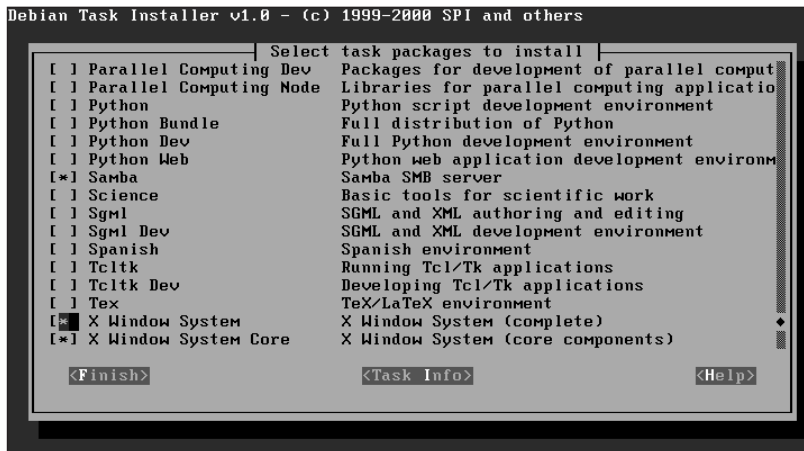


Figure 1-40: Task selection

X Window configuration

Debian uses an X Window configuration tool known as anXious. If you have a PCI or Sbus video card, the installer can detect it and install the correct X Window server for you. In the first screen, select the font sizes you would like installed. In most cases you should go ahead and choose all available sizes. In the next screen, choose the terminal emulators to install. The default should fit most situations.

In the next screen, choose the window manager of your choice. WindowMaker is recommended as it is well supported in the default Debian install.

If you would like to boot to a GUI login when the system starts up, install `xdm`. It is usually recommended to not install `xdm` during the install. This way if there is a problem with X Window it can be fixed without causing the system not to boot properly.

The next steps require you to configure the mouse type that you have, as shown in Figure 1-41. We recommend that you enable three-button emulation if you have only a two-button mouse.

The next several screens prompt you for the mouse device and keyboard mapping, which you should usually just leave as default.

Figure 1-42 shows the monitor configuration screen that is displayed next. Pick the correct monitor; consult your monitor's documentation if needed. The next configuration option is the refresh rates for your monitor. Be sure these are set correctly. Again, these should be in your monitor's documentation. The final step in configuring the monitor is to give it a name, which is referenced in the X Window configuration file.

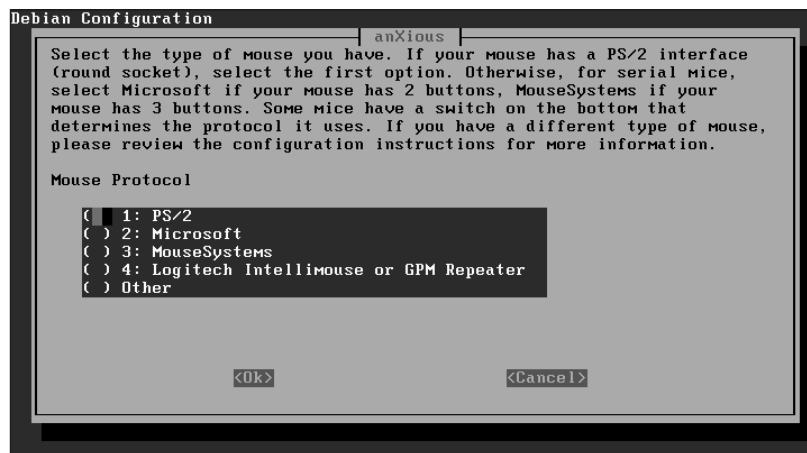


Figure 1-41: Mouse configuration

The next section of questions concerns the configuration of the video card. Select the amount of memory installed on the video card.

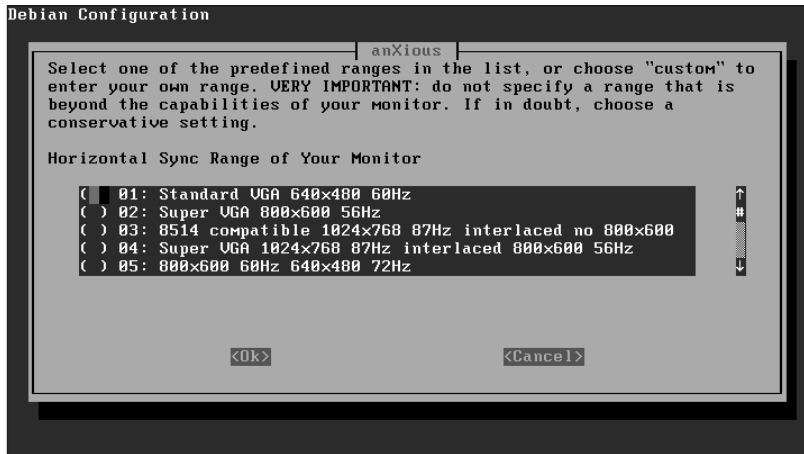


Figure 1-42: Monitor configuration

Enter a name for the video card to be referenced in the X Window configuration file. Some video cards require a *clockchip setting*, which configures the programmable clock generator, as shown in Figure 1-43. If you are unsure select none.

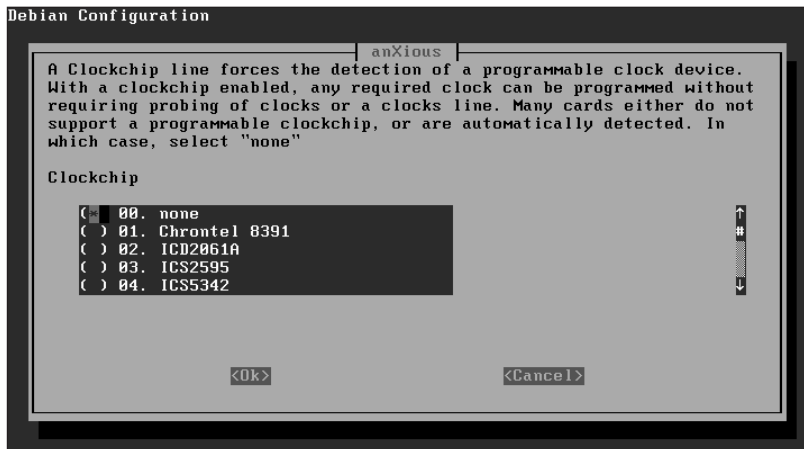


Figure 1-43: Clockchip setting

Some older video card configurations require a clocks line. The setup can probe your system for these lines. If you have an older system you may need these, but in most cases just answer No. The next several screens prompt you for the color depth and screen resolutions to use in X Window. Choose the appropriate settings, which can be changed later if needed.

The final step in the X Window configuration is to save the configuration file, as shown in Figure 1-44. Accept the default location and file name, unless you need this is a special case.

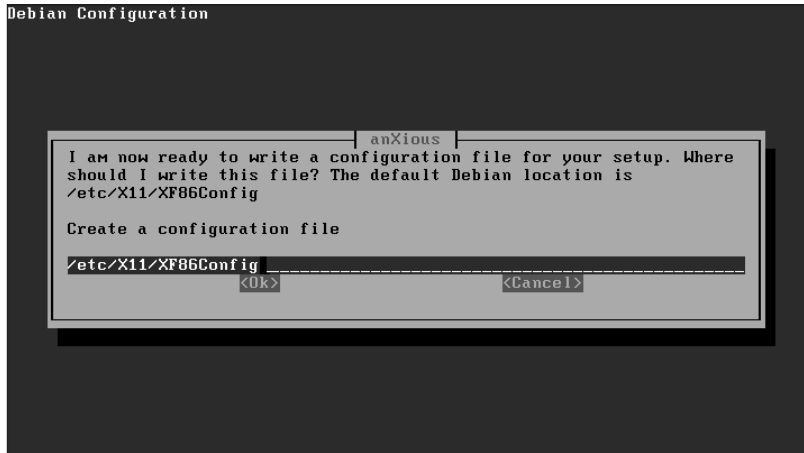


Figure 1-44: X Window configuration file

Package installation

After the X Window configuration the actual package installation process begins. Depending on your installation sources, you may be prompted to enter CD-ROMs or disks. Some of the packages being installed need to be configured, as shown in Figure 1-45. Complete all of the configuration steps as prompted.

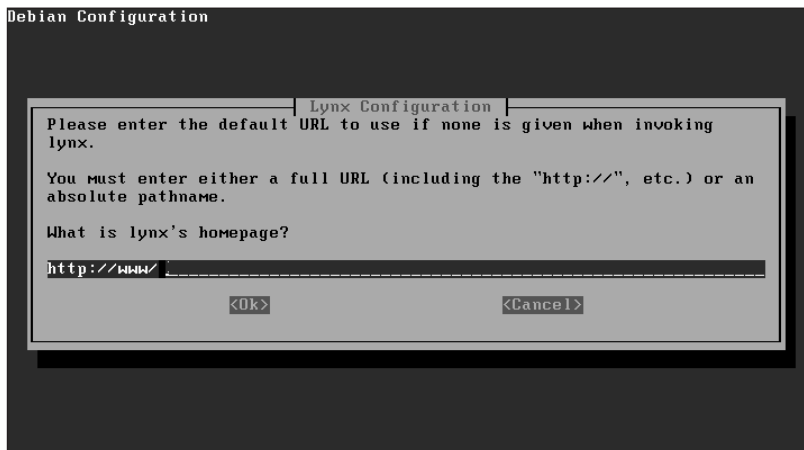


Figure 1-45: Configuration screen

After all of the packages have been installed and configured the congratulations screen is shown. After hitting Enter you are shown the login prompt for your system, no reboot is needed.

Key Point Summary

This chapter gave you an overview of Linux and how it came to be. The important ideas to come away from this chapter with are how Linux is licensed and the general setup issues relevant to all distributions.

- ♦ The Linux kernel and most software is released under the GPL, GNU General Public License.
- ♦ Free software does not mean free of charge, but free as in speech and ideas.
- ♦ Linux is a stable, multiuser, multitasking operating system.
- ♦ The Linux environment is divided up in to kernel space and user space.
- ♦ Linux runs on a variety of different hardware platforms.
- ♦ Virtual memory is provided by means of swap partitions.
- ♦ It is common to section up a disk in to several partitions when using Linux to ease administration.
- ♦ Some boot managers cannot boot a Linux kernel if it is placed past sector 1,023 of a hard disk.



STUDY GUIDE

The following questions and exercises will allow you to review the information covered in this chapter. Take your time when completing this section of the chapter, carefully reviewing any questions that you may have. Answering the question correctly is not as important as understanding the answer, so review any material of which you might still be unsure. Being comfortable with the questions and answers presented here will help you be more prepared for the certification exam questions.

Assessment Questions

1. Which tool is used to configure a sound card device?
 - A. `sndconf`
 - B. `sndconfig`
 - C. `soundconf`
 - D. `soundconfig`
2. Which file holds the Plug-and-Play configuration information?
 - A. `/etc/pci.pnp.conf`
 - B. `/etc/pnp.conf`
 - C. `/etc/rc.d/init.d/isapnp`
 - D. `/etc/isapnp.conf`
3. Drivers reside in which environment?
 - A. Kernel space
 - B. Device space
 - C. Application space
 - D. User space
4. How can you view the used IRQs on a system?
 - A. `cat /etc/interrupts`
 - B. `cat /proc/irq`
 - C. `cat /proc/interrupts`
 - D. `echo 1 > /proc/interrupts`

5. Which type of modems are not traditionally supported under Linux?
 - A. Winmodem
 - B. 3Com
 - C. PCI
 - D. PCMCIA

6. Which platforms does Linux currently support?
 - A. Intel x86
 - B. Sun SPARC
 - C. Compaq Alpha
 - D. All of the above

7. Linux requires a graphical video card with at least 2MB of memory.
 - A. True
 - B. False

8. How many primary partitions can you have on a hard disk?
 - A. 4
 - B. 8
 - C. 16
 - D. 32

9. The Linux kernel cannot be placed beyond what sector, when using some versions of LILO?
 - A. 512
 - B. 2,048
 - C. 999
 - D. 1,023

10. Which tool does the install of Red Hat use for disk partitioning?
 - A. Disk Druid
 - B. YaST
 - C. fdisk
 - D. cfdisk

Scenarios

1. You have just received a new workstation, and you plan to install Linux on it. What information should you gather before installing Linux? How do you prepare the system for installation?
2. You are installing Linux on an older system, but cannot get the network card to function reliably. You think it may be due to a hardware conflict. What steps could you take to resolve the problem?

Answers to Chapter Questions

Chapter Pre-Test

1. GNU is a recursive acronym that stands for “GNU is Not UNIX.”
2. Winmodems are usually not compatible with Linux.
3. Yes, you can charge for distribution.
4. The boot disk on a SCSI system is normally set to 0, but if there is no device with a 0 ID, it can be set to 1.
5. Yes, Linux can exist with another operating system and can enable you to choose which operating system to boot to.
6. Yes, Linux fully supports most new hardware and standards.
7. Current kernels allow swap partitions up to 4GB.
8. Only one partition is required to install Linux, though more may help with system administration.
9. LILO is the default boot loader on most Linux distributions.
10. User space contains user applications. Kernel space contains the kernel and drivers.

Assessment Questions

1. **B.** The `sndconfig` tool by Red Hat is used to configure sound devices. For more information, see the “Sound cards” section of this chapter.
2. **D.** The `isapnp.conf` file is stored in `/etc`. For more information, see the “Configuring Plug-and-Play devices” section of this chapter.
3. **A.** Drivers run from kernel space as they provide an interface between hardware and the kernel. For more information, see the “Overview of the Linux Architecture” section of this chapter.

4. **C.** The `/proc/interrupts` file contains the currently used IRQs. You can view this file using the `cat` command. For more information, see the “Listing interrupts” section of this chapter.
5. **A.** Winmodems require software to handle most operations, instead of hardware. Very few have drivers for Linux. For more information, see the “Peripherals and other hardware” section of this chapter.
6. **D.** Linux supports a variety of hardware platforms. For more information, see the “CPU requirements” section of this chapter.
7. **B.** Linux requires a graphical video card only if you run X Window or other graphical applications. For more information, see the “Video requirements” section of this chapter.
8. **A.** A hard disk can have up to 4 primary partitions, or 3 primary partitions and 1 extended partition. For more information, see the “Partitioning Schemes” section of this chapter.
9. **D.** Due to BIOS limitations, not all versions of LILO can boot a kernel past sector 1,023. For more information, see the “Partitioning Schemes” section of this chapter.
10. **A.** Disk Druid is used during the Red Hat installation. For more information, see the “Using Disk Druid” section of this chapter.

Scenarios

1. As with any Linux installation, the first step is to gather information on which hardware is installed in the system, mainly RAM, CPU type, hard disk size, video card, and any other peripherals such as modem and sound card. Take the list and compare it to the hardware compatibility list from the Linux distribution you plan to install. If any hardware devices are not on the list, check with the manufacturer for a driver or with other popular Linux sites.
2. The first step is to check which addresses the hardware is using. You could view the `/proc/interrupts`, `/proc/dma`, and `/proc/ioports` files to see what addresses are used and free. If the network card is using the same configuration as another device, you can either change it via jumpers on the hardware or do it with software, depending on your options.

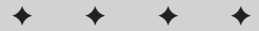
Using the Shell

EXAM OBJECTIVES

Exam 101 ♦ General Linux, Part 1

1.3 GNU and UNIX Commands

- **Work Effectively on the UNIX Command Line.** Interact with shells and commands using the command line. Includes typing valid commands and command sequences, defining, referencing and exporting environment variables, using command history and editing facilities, invoking commands in the path and outside the path, using command substitution, and applying commands recursively through a directory tree.
- **Create, Monitor, and Kill Processes.** Includes running jobs in the foreground and background, bringing a job from the background to the foreground and vice versa, monitoring active processes, sending signals to processes, and killing processes. Includes using commands `ps`, `top`, `kill`, `bg`, `fg`, and `jobs`.
- **Modify Process Execution Priorities.** Run a program with higher or lower priority, determine the priority of a process, change the priority of a running process. Includes the command `nice` and its relatives.



CHAPTER PRE-TEST

1. Which file contains the default shell setting for each user?
2. What is the best way to add to the `PATH` environment variable for all users on a system?
3. How is the size of the bash history list file determined?
4. Which key is used for command completion with the bash shell?
5. The default Readline Library uses which Linux editor?
6. Which command allows you to edit the last command you entered using the default editor?
7. Which file contains the command history?
8. What command would you use to view the setting for the `HOME` environment variable?
9. Programs started from which file operate in the `init PATH`?
10. How do you run a program located in the `pwd`?
11. Which function allows the output from a command to be used instead of the command?
12. Which option is used for recursive operation of a command?
13. Which command is used to change the priority of a currently running process?
14. Which utility provides real-time information on currently running processes?
15. Which utility is used to display jobs running in the background?

This chapter introduces you to many of the functions of the UNIX or Linux shell. You learn about the *bash shell*, which is the default Linux shell, as well as how to change this default shell setting. Using environment variables and command history is covered here, as is command substitution. Knowledge and experience of the shell is crucial when working with Linux systems. You need to understand the material in this chapter before moving on as it helps create a foundation for the rest of the book.

Understanding Shells

Objective

1.3 GNU and UNIX Commands

- **Work effectively on the UNIX Command Line.** Interact with shells and commands using the command line.

The shell allows users to enter commands and then interprets these commands into instructions for the Linux operating system. Linux allows you to use a number of different shells; however, the default shell installed with Linux is the *bash shell*. The bash shell contains characteristics of older shells such as the sh and Korn shells. These shells have been used on other UNIX-based operating systems. This default shell is specified in the `/etc/passwd` file and can be changed there on a per user basis. Allowing each user on the system to specify the shell he or she uses provides a great deal of flexibility for the users and administrators of the system. To change to a different shell, you simply type the full path along with the command name of the new shell. So if you wished to switch to the Berkeley UNIX C shell (csh), you would type the following:

```
/bin/csh
```

To return to the default login shell, simply either type **exit** or press CTRL-D. As you can see by trying the various shells installed on your system, the different shells utilize different prompts. The default login shell is contained in the environment variable `SHELL`. The environment variables are settings used to customize the shell and the user. Each shell has a corresponding resource configuration file, or rc file, located in the `/etc` directory that contains global settings for the shell. For the bash shell the file is `/etc/bashrc`. Each user also has a shell configuration file located in their home directory; for the bash shell this file is `.bashrc`. If you wish to know which default shell you are currently configured to use, type the following command:

```
echo $SHELL
```

The available shells and paths are listed in the `/etc/shells` file and are listed in Table 2-1.

Table 2-1
Linux Shells

<i>Shell</i>	<i>Explanation</i>
/bin/bash	<i>Bourne-Again Shell</i> that is compatible with the <code>sh</code> shell, which includes features of both the Korn and C shells. This may be a link to the <code>/bin/bash2</code> file on some distributions.
/bin/sh	The Bourne Shell. On many systems this file is linked to <code>/bin/bash</code> .
/bin/ash	A System V version of the <code>sh</code> shell.
/bin/bsh	This file is linked to <code>/bin/ash</code> .
/bin/bash2	Bourne-Again Shell version 2.
/bin/csh	The Berkeley UNIX C shell.
/bin/tcsh	An enhanced version of the Berkeley UNIX C shell.

The `chsh` command is used to change the default shell for the user. The options used with the `chsh` command are listed in Table 2-2.

Table 2-2
chsh Options

<i>Option</i>	<i>Alternate</i>	<i>Function</i>
-s	--shells	Specifies the default login shell for this user.
-l	--list	Lists the shells specified in the <code>/etc/shells</code> file.
-u	--help	Displays the options for the <code>chsh</code> command.
-v	--version	Displays version information for the shell.

So to switch the default shell to the Bourne-Again Shell version 2, you would use the following command:

```
chsh -s /bin/bash2
```

After running this command you are prompted for the user's password. Each user, other than root, can change only his or her own shell. Root can change everyone's default login shell.

One important fact to keep in mind is that scripts must be written for the shell in which they run. This is one reason that you may wish to standardize all users on one shell. To do this, simply remove the other shells from the `/etc/shells` file;

this will prevent a user from changing his or her default login shell. If you wish to prevent users from running specific shells other than the default login shell, simply delete the command for these shells.

Using the Command Line

Objective

1.3 GNU and UNIX Commands

- **Work effectively on UNIX Command Line.** Includes typing valid commands and command sequences, using command substitution, and applying commands recursively through a directory tree.

Once you have logged into the Linux system you are presented with a shell. This simply appears as a command-line interface. At the shell prompt you type commands that are interpreted by the shell and then sent to the system. The shell that you are running configures the shell prompt. Most Linux systems default to the bash shell and are usually configured to display the user name, hostname, and present working directory at the prompt. An example of this is the following:

```
[angie@redhat /etc]$
```

In this example `angie` is the user name, `redhat` is the hostname, and `/etc` is the present working directory, called the `pwd`. The shell prompt for the bash shell is the `$` symbol. The shell prompt is configured by the `/etc/bashrc` file. Utilizing this file, you can change the shell prompt display settings.

Commands typed at the command line aren't sent to the system until the Enter key has been pressed. This allows commands to be edited before passing them onto the system. Commands typed at the command line must follow a specific syntax. The first word is the command that is to be run; this can include an absolute path, which is the full path to the command that ends with the command name. Next in the command line are the options that are used for the command. Command arguments follow the command options. Each of these elements is separated on the command line by a space. The format for a command is as follows:

```
$ command options arguments
```



It is important to remember this syntax. Questions on the exam will cover the proper use of commands, options and arguments.

For example, take the `ls` command. This command lists the files and directories in the `pwd`. To simply run the command you would type the following:

```
$ ls
```



The `ls` command, its use, and its options are covered in more detail in Chapter 6.

The output would look something like this:

```
abc123names      list.gz          mergednicksandnames  nicks
abcnames         list1           mycommands           numberednames
alphanames       list2           mydoc                 readmes
alphanicks       list_1          myfile.12.11.00      sortednameslist
alteredervices  list_2          myfiles              status
auto             longerfile.gz   mynewcommands        stuff
doomnicks        longerfile2.gz  names                test
fivenamesaa      longerfile3.gz  nameslist            test2
fivenamesab      longerfile~.gz  nameslisterrors     xaa
fivenamesac      marital         new                  xab
infodoc          mergedalpha     newtest              xac
```

Options are one-letter codes that are preceded by a dash (-), and modify the effect of the command. Multiple options can be combined and used with a single command. Options are case-sensitive, and many times a lowercase letter will signify an option different from its uppercase counterpart. One very important option available with most commands is the `-R` option which specifies that the command should be run recursively through the directory tree.



The option for recursive function of a command is very important and will appear on the exam.

If in the `ls` example you want to list the files and directories with a trailing slash (/) on the directories listed, you can use the `-F` option. An example of this follows:

```
$ ls -F
abc123names      list.gz          mergednicksandnames  nicks
abcnames         list1           mycommands           numberednames
alphanames       list2           mydoc                 readmes
alphanicks       list_1          myfile.12.11.00      sortednameslist
alteredervices  list_2          myfiles/             status@
auto*            longerfile.gz   mynewcommands        stuff/
doomnicks        longerfile2.gz  names                test*
fivenamesaa      longerfile3.gz  nameslist            test2
fivenamesab      longerfile~.gz  nameslisterrors     xaa
fivenamesac      marital*        new*                 xab
infodoc          mergedalpha     newtest/             xac
```

Now, suppose you want to see a listing of all of the files and directories contained in the `/etc` directory from another location on the system. In this listing you want a trailing slash to be displayed with the directory names. The `/etc` directory can be used as an *argument* for the `ls` command. An argument is another option that can be used with a command. In the following example the argument is a directory that will be examined by the command. That command, with the option and the argument, looks like this:

```

$ ls -F /etc
DIR_COLORS      filesystems      issue            nsswitch.conf   sendmail.cw
HOSTNAME        fstab           issue.net        nwserv.conf     sendmail.mc
MACHINE.SID    ftpaccess       krb5.conf        nwserv.stations services
Muttrc         ftpconversions ld.so.cache     pam.d/          shadow
X11/          ftpgroups      ld.so.conf      passwd          shadow-
adjtime        ftphosts       lilo.conf       passwd-         shells
aliases        ftpusers       lmhosts         passwd.0LD      skel/
aliases.db     gettydefs      localtime       pbm2ppa.conf*   smb.conf
anacrontab     gpm-root.conf login.defs       pcmcia/         smb2.conf
at.deny        group          logrotate.conf  pine.conf.rpmsave smbusers
bashrc         group-         logrotate.d/    pnm2ppa.conf    smrsh/
charsets/      gshadow        ltrace.conf     ppp/            snmp/
codepages/     gshadow-       lynx.cfg        printcap        ssh/
conf.linuxconf gtk/           mail/           profile         ssl/
conf.modules   host.conf      mail.rc         profile.d/       sysconfig/
cron.d/        hosts          mailcap         protocols       sysctl.conf
cron.daily/    hosts.allow    mailcap.vga     pwdb.conf       syslog.conf
cron.hourly/   hosts.deny     man.config      rc.d/           termcap
cron.monthly/  httpd/        midi/          redhat-release  tripwire/
cron.weekly/   identd.conf    mime.types     resolv.conf     uucp/
crontab        inetd.conf     minicom.users   rmt@            vga/
csh.cshrc     info-dir       motd            rpc             wgetrc
csh.login     initlog.conf   mtab           rpm/            yp.conf
default/      inittab       named.boot      screenrc        ypserv.conf
dumpdates     inputrc       named.conf      security        security/
exports       ioctl.save     news/           sendmail.cf
fdprm         isapnp.gone    nmh/

```

You can enter multiple commands on the same line. When entered on the same line and separated by a semicolon (;), the first command will complete before the next command is started. You would type the commands like this:

```
$ ls -F /etc ; ls -F /usr
```

This displays the files and directories listed in the `/etc` directory, followed by those in the `/usr` directory. All of the directories contained in the listing will be followed by a slash.

Another special character you can use when entering commands is the backslash (`\`) command. When entered right before pressing the Enter key, the backslash (`\`) command allows commands to span multiple command lines. It causes the system to ignore the Enter key and treat all of the commands as though they are on the same command line. This function can be useful when typing in extremely long commands that span more than one command line. An example of its use is as follows:

```
$ ls -F /etc \
ls -F /usr
```



You'll run most long commands using scripts, instead of typing them at the command line. However, it is important to remember the use of the backslash for the exam.

Command completion

The bash shell includes a feature called *command completion*. This enables you to type the first few letters of the command at the prompt, hit the Tab key, and have the system complete the command for you. If you wish to run the command `dmesg` to display the kernel ring buffer, you could type the following:

```
$ dm
```

If you then hit the Tab key, the system will fill in the following command:

```
$ dmesg
```

If there is more than one possible match to what you typed before pressing the Tab key, the system will simply beep. Pressing the Tab key once again will display all of the possible command matches. Pressing the Esc twice performs the same action as pressing the Tab key.

Editing commands with the Readline Library

The bash shell allows you to edit commands you type at the shell prompt before pressing the Enter key. The editor used to for editing these commands is the *Readline Library*. This editor runs by default in the `emacs` mode using the same keystrokes as the `emacs` editor. The Readline Library enables you to easily fix errors in your command sequence without having to retype the entire sequence. You can easily move your cursor to the spot where you want to make changes and either delete or insert the corrections into the command line. When the error has been fixed you can simply press Enter without moving to the end of the line. The special keys used by the Readline Library and their functions are listed in Table 2-3. While this information is not required for the exam, it can be very useful when working with the command line.

Table 2-3
Readline Library Commands

Key Combination	Function
Ctrl-b	Move back one character.
Ctrl-f	Move forward one character.
Del	Delete the character to the left of the cursor.

Key Combination	Function
Ctrl-d	Delete the character underneath the cursor.
Ctrl-a	Move to the start of the line.
Ctrl-e	Move to the end of the line.
Esc-f	Move forward a word.
Esc-b	Move backward a word.
Ctrl-l	Clear the screen, reprinting the current line at the top.
Ctrl-k	Kill (delete) the text from the current cursor position to the end of the line.
Esc-d	Kill from the cursor to the end of the current word, or if between words, to the end of the next word.
Esc-Del	Kill from the cursor the start of the previous word, or if between words, to the start of the previous word.
Ctrl-w	Kill from the cursor to the previous white space. This differs from Esc-Del because the boundaries separating words differ.

The editor can be configured globally using the `/etc/inputrc` file for global changes. To make changes for a specific user you can edit the `.inputrc` file located in the user's home directory. This file can be used to change the key mappings for the Readline Library editor, and map keys to commonly used commands. To view your keyboard bindings type the following command:

```
bind -v
```

Command substitution

It is possible to replace a command with the output it produces. This is known as command substitution. Command substitution is a valuable tool used with scripts. This allows the output from one command to work as the argument for another. A useful example of this is the `pwd` command. The `echo` command is used to output the `pwd` so that it is available to the system.

```
# echo `pwd`
```

Using the history file

Objective

1.3 GNU and UNIX Commands

- **Work effectively on the UNIX Command Line.** Includes using command history and editing facilities.

The history file contains a list of the commands issued at the command prompt. The number of commands stored here is specified by the `HISTSIZE` environment variable in either the `/etc/profile` or `~/.profile` file. The default setting for this variable is 1,000 entries. The `history` command displays all entries in the history file, which is `~/.bash_history`.

You can cycle through the entries in the history file the arrow keys. The up arrow will cycle to previous entries in the command history while the down arrow cycles down through the command history. This can provide a shortcut to running commands that you entered earlier. When cycling through the commands, you can edit them on the command line before rerunning the commands.

The `HISTCMD` variable is used to provide the history index number of the command currently being run. The `HISTFILE` variable specifies the file used to contain the command history, with the default of `/home/user/.bash_history`. The `HISTFILESIZE` variable specifies the maximum number of lines contained in the `HISTFILE`.

fc

The `fc` utility provides another option for editing commands in the history file before running them. The `fc` utility opens the command into the default editor, where you can edit and save it before rerunning the command. The `fc` utility allows you to specify the number of the history events to edit, which enables you to edit a range of commands at once. The editor to use can also be specified using this command. Once the editor is invoked using the `fc` command the file is processed like any other text file using the editor. For instance, the `-l` option is used to list a range of values. The following example uses the `fc` utility to list the history events 1020-1025.

```
# fc -l 1020 1025
1020 cd /etc
1021 ls -al
1022 vi services
1023 vi hosts
1024 man ls
1025 ls -al s*
```

Environment Variables and Settings

Objective

1.3 GNU and UNIX Commands

- Work effectively on the UNIX Command Line. Includes defining, referencing and exporting environment variables.

Any command that is executed is a *process*. Processes can be initiated by the user, or by the operating system itself. Further, processes can create other processes as they execute. If process A is executed and creates process B as part of its execution, B is called a *child process* of A, and A is the *parent process* of B. A process can have many child processes, but only one parent process.

Each process has an associated *environment*, a series of variables that define certain parameters for the process. The environment is expressed as a list of variables and their values. Both the variable's name and value are represented as strings that can be made up of most standard characters. When a child process is created, it inherits the environment of its parent process. A child process can then modify its environment before passing it to its own child processes.

For example, the `PATH` variable indicates which directories the system should look in to execute a command, if no directory path is specified in the command. In Linux, even the processes run by the operating system search through directories in the `PATH` variable: you can use the path mechanism anywhere you try to execute a command. Even if there is no variable `PATH` in the environment, the system always checks the directories `/bin` and `/usr/bin` for suitable commands.

The variables can be configured and made available to the system using the `export` command. This is done as shown in the following example:

```
# PATH=/bin
# export PATH
```

To view the setting for an environment variable, the `echo` command is used. When specifying a variable to display the variable name is preceded by the `$` symbol. An example of this is shown here:

```
# echo $PATH
/bin:/usr/bin:/usr/local/bin
```



Pay close attention to the files listed in the following sections. It is very important to understand that the shell configuration files are used to configure the shell environment for users.



These configuration files are covered in more detail in Chapter 10.

Editing the `PATH` variable



1.3 GNU and UNIX Commands

- **Work effectively on the UNIX Command Line.** Includes invoking commands in the path and outside the path.

When you run a command, it is important to understand how the command is located. The `PATH` environment variable is used by the shell to determine where to look for commands. One common location to be contained within this variable is `/bin`. This directory contains many of the commands used by the users of the system including the `ls` command which is used to provide file and directory listings.

For example, your `PATH` variable may be set to the following:

```
/bin:/usr/bin:/usr/local/bin
```

This is a list of directories for the shell to search, with each directory separated by a colon. When you type the command `ls`, the shell first looks for `/bin/ls`, then `/usr/bin/ls`, and so on.

This variable is very useful because it allows commands to be executed without knowing and specifying the full path to the command. This is especially helpful because on many systems executables are scattered about in many places, such as `/usr/bin`, `/bin`, or `/usr/local/bin`. Note that the `PATH` has nothing to do with finding regular files.

As you see here, the `home` directory is not included in the path. When running a command from the `home` directory, or any directory not in the path, you can supply the full path and command name, such as `/home/angie/myfile`. Or, if you wish to run a file from the `pwd`, you can type the following:

```
./myfile
```

The `./` specifies that you are running a command from the `pwd`. This helps ensure that you are running the version of the file that you intend to run.



Some users add the `pwd` to their path statement using a period (`.`); however, this practice isn't recommended for safety concerns. Because the `ls` command is frequently used, someone could place a malicious file by this name throughout the system. If the user executes the `ls` command from one of these locations, great damage could be done to the system.

Typically, you don't have to change your `PATH`, but it very useful to understand what your current `PATH` is.

You can check your `PATH` using the following command:

```
# echo $PATH
/opt/kde/bin:/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin:/home/angie/bin
```

Because the colon is a separator this `PATH` represents the following list of directories:

```
/opt/kde/bin
/usr/local/bin
/bin
/usr/bin
/usr/X11R6/bin
/home/angie/bin
```

Here is the output from the command `echo $PATH` run on my system on the account `root`:

```
/opt/kde/bin:/sbin:/bin:/usr/sbin:/usr/bin:/usr/X11R6/bin:/root
/bin
```

You can change the `PATH` for all users on the system by logging in as `root`, editing the file `/etc/profile`, and adjusting the line starting with `PATH=`. To edit this file using the `vi` editor type the following:

```
vi /etc/profile
```

Re-login for the change to take effect. To set up the `PATH` for an individual user only, edit the file `/home/user_login_name/.bash_profile` (note the dot in front of the filename—files starting with a dot are normally invisible; you have to use `ls -a` to see them).



Using text editors is discussed in Chapter 4.

The specification for the path in `/home/username/.bash_profile` may then look like this:

```
PATH="$PATH:$HOME/bin
export PATH
```

This command takes the contents of the `PATH` variable (as set for all users in `/etc/profile`) and appends to it the name of your home directory as set by the variable `HOME` with an attached `/bin`. Finally, the command assigns the resulting string back to the variable called `PATH`. You need to use the `export` command after modifying `PATH` or any other user-environment variable, so that the variable is visible outside of the script that sets it.

When you wish to run a command that is not located within the directories specified in the `PATH` variable you must specify the absolute path of the command. This is the full path starting at the system root directory and ending with the command name. The absolute path to the `ls` command on most systems is `/bin/ls`. Being forced to specify the absolute path can provide a layer of protection for commands that are powerful and should be run with care because requiring the absolute path helps ensure that the commands aren't run accidentally.

The `init` process and the `PATH` variable

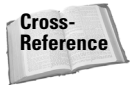
`init` is a parent process for all the other processes of the system. Other processes inherit the environment of the `init` process, and the `PATH` is the `init` path in the rare case that no other path is set.

The `init` path is fixed in the source of the `init` program and is as follows:

```
/usr/local/sbin:/sbin:/bin:/usr/sbin:/usr/bin
```

Note that `init` path does not contain `/usr/local/bin`.

All the programs that are started from `/etc/inittab` work in the `init` environment, especially system initialization scripts in `/etc/init.d`. Some examples of these programs are FTP and font servers.



The `/etc/inittab` file and the processes started in this file are covered in greater detail in Chapter 8.

Prompt

The shell prompt can be configured by the user to display a variety of information. For example, on my machine, the prompt may look like this:

```
[angie@deedee angie]$ _
```

Here `angie` is my login name, `deedee` is the name of the computer, the second `angie` is the name of my current working directory, and `_` represents the cursor.

The prompt is set by the environmental variable called `PS1`. To display the current setting, use this command:

```
echo $PS1
```

The system-wide setting of the prompt (for all users on the system) is in the file `/etc/bashrc`, which on my system contains the following line:

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

To customize the prompt, edit the file `/etc/bashrc` (as root) and insert almost any text inside the quotes. Table 2-4 lists the meaning of some special codes you can use in the `PS1` variable.

Table 2-4
Settings Used To Configure the Prompt

Setting	Meaning
<code>\u</code>	User name of the current user.
<code>\h</code>	The name of the computer running the shell (hostname).
<code>\W</code>	The base of the name of the current working directory.
<code>\w</code>	The full name of the current working directory.
<code>\\$</code>	Displays “\$” for normal users and “#” for the root.
<code>\!</code>	History number of the current command.
<code>\#</code>	Number of the current command.
<code>\d</code>	Current date.
<code>\t</code>	Current time.
<code>\s</code>	Name of the shell.
<code>\n</code>	New line.
<code>\\</code>	Backslash.
<code>\[</code>	Begins a sequence of nonprintable characters.
<code>\]</code>	Ends a sequence of nonprintable characters.
<code>\nnn</code>	The ASCII character corresponding to the octal number <i>nnn</i> .
<code>\$(date)</code>	Output from the <code>date</code> command (or any other command for that matter).

HOME

The `HOME` variable is used to represent the user’s home directory. This directory is usually located at `/home/username`. So for a user with the user name `angie`, the home directory is located at `/home/angie`. This variable is useful with scripts that require the path to the user’s home directory. The home directory setting for each user is contained in the `/etc/passwd` file.

Managing Processes

Objective

1.3 GNU and UNIX Commands

- **Create, Monitor, and Kill Processes.** Includes running jobs in the foreground and background, bringing a job from the background to the foreground and vice versa, monitoring active processes, sending signals to processes, and killing processes. Includes using commands `ps`, `top`, `kill`, `bg`, `fg`, and `jobs`.

Whenever a command or job is executed in a shell, a process is created. Each process started is assigned a unique process id number or PID. These processes and PIDs can be viewed using the `ps` command. This command provides process status by presenting a snapshot of the current system processes. The information presented by the `ps` command is configured using a variety of switches. The most commonly used of these switches are shown in Table 2-5.

Table 2-5
Switches Used with `ps`

Switch	Use
<code>a</code>	Displays processes for all users.
<code>txx</code>	Displays processes within controlling terminal <code>txx</code> .
<code>u</code>	Displays user information for the process.
<code>l</code>	Displays in long format with detailed information.
<code>s</code>	Displays information in signal format.
<code>m</code>	Displays memory information.
<code>x</code>	Displays processes without a controlling terminal.
<code>S</code>	Displays CPU time and page faults of child processes.

In order to view all processes run on the system, the following command is used.

```
# ps -aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.2  1120  372 ?        S    18:26   0:05 init [3]
root         2  0.0  0.0      0     0 ?        SW   18:26   0:00 [kflushd]
root         3  0.0  0.0      0     0 ?        SW   18:26   0:00 [kupdate]
root         4  0.0  0.0      0     0 ?        SW   18:26   0:00 [kpiod]
root         5  0.0  0.0      0     0 ?        SW   18:26   0:00 [kswapd]
root         6  0.0  0.0      0     0 ?        SW<  18:26   0:00 [mdrecoveryd]
root       272  0.0  0.5  1432  692 ?        S    18:28   0:00 /sbin/pump -i eth
bin        316  0.0  0.3  1208  408 ?        S    18:28   0:00 portmap
root       367  0.0  0.4  1172  528 ?        S    18:28   0:00 syslogd -m 0
root       376  0.0  0.5  1436  760 ?        S    18:28   0:00 klogd
nobody    390  0.0  0.4  1292  628 ?        S    18:28   0:00 identd -e -o
nobody    394  0.0  0.4  1292  628 ?        S    18:28   0:00 identd -e -o
nobody    395  0.0  0.4  1292  628 ?        S    18:28   0:00 identd -e -o
nobody    396  0.0  0.4  1292  628 ?        S    18:28   0:00 identd -e -o
nobody    397  0.0  0.4  1292  628 ?        S    18:28   0:00 identd -e -o
```

The `ps` command is useful for providing a snapshot of the process status. However, there are times that you may want real-time information on the system processes. The `top` command is used for this purpose. Because this process itself can be CPU intensive, it should be used only when necessary. The information presented by `top` can be arranged according to CPU usage, memory usage, and runtime. The most common way to run `top` is simply to type the command name at the command line. From within the `top` utility type the `s` command and then specify the option that you wish to use. The options used with `top` are displayed in Table 2-6.

Table 2-6
Options Used with `top`

<i>Option</i>	<i>Use</i>
d	Specifies the seconds between updates.
q	Specifies that there should be no delay between updates.
S	Specifies that child process information should be included in the information displayed.
s	Runs the process in secure mode.
i	Specifies that idle processes are to be ignored.
c	Displays the full command line of processes.

Following is an example of the output produced by the `top` command. The `top` display is exited using the `q` key.

```
# top

11:51pm up 5:25, 3 users, load average: 0.00, 0.00, 0.00
43 processes: 42 sleeping, 1 running, 0 zombie, 0 stopped
CPU states: 0.1% user, 0.3% system, 0.0% nice, 99.4% idle
Mem: 127976K av, 124328K used, 3648K free, 47936K shrd, 88108K buff
Swap: 264056K av, 336K used, 263720K free, 14080K cached
```

```

PID USER      PRI  NI  SIZE  RSS  SHARE STAT   LIB  %CPU  %MEM  TIME  COMMAND
 902 root        18   0   856   856   668 R    0    0.5  0.6   0:00  top
    1 root         0   0   380   372   324 S    0    0.0  0.2   0:05  init
    2 root         0   0     0     0     0 SW   0    0.0  0.0   0:00  kflushd
    3 root         0   0     0     0     0 SW   0    0.0  0.0   0:00  kupdate
    4 root         0   0     0     0     0 SW   0    0.0  0.0   0:00  kpiod
    5 root         0   0     0     0     0 SW   0    0.0  0.0   0:00  kswapd
    6 root        -20  -20     0     0     0 SW<  0    0.0  0.0   0:00  mdrecoveryd
 272 root         0   0   692   692   568 S    0    0.0  0.5   0:00  pump
 316 bin         0   0   408   408   328 S    0    0.0  0.3   0:00  portmap
```



```

367 root      0  0  528  528  428 S      0  0.0  0.4  0:00 syslogd
376 root      0  0  760  760  388 S      0  0.0  0.5  0:00 klogd
390 nobody    0  0  628  628  520 S      0  0.0  0.4  0:00 identd
394 nobody    0  0  628  628  520 S      0  0.0  0.4  0:00 identd
395 nobody    0  0  628  628  520 S      0  0.0  0.4  0:00 identd
396 nobody    0  0  628  628  520 S      0  0.0  0.4  0:00 identd
397 nobody    0  0  628  628  520 S      0  0.0  0.4  0:00 identd
408 root      10 0  620  620  512 S      0  0.0  0.4  0:00 crond
426 root      0  0  488  488  412 S      0  0.0  0.3  0:00 inetd
436 root      0  0 1096 1096  968 S      0  0.0  0.8  0:00 sshd
451 root      0  0  532  532  448 S      0  0.0  0.4  0:00 lpd

```

There may be times that a process stops responding and must be terminated. This is done using the `kill` command. The `kill` command is used with the following syntax:

```
kill -s signal process
```

The process can be specified using the PID or the process name. Only the root user has the ability to kill processes belonging to another user. The default signal sent to a process with the `kill` command is the TERM signal. It is also possible to specify other signals with this command. The most common signal levels and their meanings are shown in Table 2-7.

Table 2-7
Common Signal Levels and Meanings

<i>Signal</i>	<i>Meaning</i>
1	SIGHUP. Stops and restarts the process.
9	SIGKILL. Stops a process without allowing it to exit gracefully.
15	SIGTERM. Stops a process by allowing it to exit.

A list of all available signal levels can be viewed using the `-l` option, as shown in the following example:

```

# kill -l
1) SIGHUP          2) SIGINT          3) SIGQUIT        4) SIGILL
5) SIGTRAP        6) SIGIOT         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

```

An example of using the `kill` command to restart a process is shown below.

```
# kill -1 408
```

Some processes occupy the shell until they are completed. These processes are said to run in the foreground. No other processes can be started at the shell until these processes are completed. Other processes are said to run in the background. When a process runs in the background, the shell is available so that new processes can be started while the background process is running. Whether a process runs in the foreground or background depends on the process and how it is started. Typically, commands run from the command line are run in the foreground. To start a command-line process in the background the `&` symbol is placed at the end of the command. The following is an example of starting the `netscape` process as a background process.

```
# netscape &
```

Running the process this way frees the shell to receive other commands while the `netscape` process runs. If you wish to move a current process from the foreground to the background another method is used. The `^Z` command will stop a process running in the foreground. Once the process has been stopped, it can be moved into the background using the `bg jobnumber` command. The `jobnumber` can be discovered using the `jobs` command. This command will display the processes currently running in the background as well as stopped jobs. An example of the correct use of these commands is as follows:

```
# netscape
^Z
# jobs
[2] Stopped netscape
# bg 2
```

In the above example the `netscape` process is started and then stopped using the `^Z` command. The `jobs` command is then used to view the jobs currently running in the background as well as those that are stopped. Then the `bg` command is used to move the `netscape` process to the background.

It is also possible to move a process from the background to the foreground. This is done using the `fg jobnumber` command. To move the `netscape` process in the example above back to the foreground the following command would be used:

```
# fg 2
```

Modifying Process Priorities

Objective

1.3 GNU and UNIX Commands

- **Modify Process Execution Priorities.** Run a program with higher or lower priority, determine the priority of a process, change the priority of a running process. Includes the command `nice` and its relatives.

Whenever a process runs, it has an assigned priority level on the system. The default priority level assigned to a process when it is run is the priority level of 0 (zero). The priority numbers exist on a scale from -20 to 19. The larger priority numbers are used to make a process run slower while lower numbers make it run faster. Only the root user has the ability to assign negative priority levels. The priority levels can be viewed with the `top` command, which is covered earlier in this chapter.

The priority level can be assigned when a process is executed using the `nice` command. This is done using the following syntax:

```
nice priority command
```

An example of the correct use of this command is shown below. The `netscape` process is executed with a priority level of 10.

```
# nice 10 netscape
```

If you wish to change the priority of a process already running or to set user process priorities, you use the `renice` command. The `renice` command uses the options shown in Table 2-8.

Table 2-8
Options Used with `renice`

Option	Use
-g	Specifies the priority for processes executed by members of the specified group.
-u	Specifies the priority level for processes executed by the specified user.
-p	Specifies the priority level for the specified process.

An example of the use of this command to set the priority level for a process currently running on the system is as follows.

```
# renice 10 -p 408
408: old priority 0, new priority
```

Key Point Summary

This chapter introduces you to many of the functions and configuration options for the shell environment. Many key points here serve as a basis for the rest of the material in the book.

- ♦ The default shell used on Linux systems is the bash shell (Bourne-Again Shell).
- ♦ This shell can be changed by the user. However, scripts must be written for the specific shell that is being run, so changing shells should be done with caution.
- ♦ The bash shell allows multiple commands to be entered on one line when separated with a semicolon (;). Commands can span multiple lines when using the backslash key followed by Enter.
- ♦ Commands can be automatically completed using the Tab key.
- ♦ The Readline Library allows commands to be edited at the command line using the `emacs` editor. This capability can be changed to use the `vi` editor.
- ♦ Command substitution is used to replace a command with its output.
- ♦ The command history is stored in the `~/.bash_history` file. The number of entries stored here is configured using the `HISTSIZE` variable. This file allows you to view and edit previously entered commands.
- ♦ The `fc` command allows you to edit entries in the history file from the default editor. Multiple entries can be edited at once.
- ♦ Environment variables and settings in the `/etc/profile` and `/etc/bashrc` file set configuration parameters for the entire system. If a user wishes to overwrite these settings with specific user settings, the files `~/.profile` and `~/.bashrc` are used.
- ♦ The `PATH` variable contains a listing of directories to search for commands that are entered at the command line.
- ♦ The `init path` is used by programs started from the `/etc/inittab` file.
- ♦ The shell prompt can be configured to display a variety of information such as `pwd`, `hostname`, and user name.
- ♦ The `HOME` variable stores the user's home directory. This variable is useful for scripts which work with this directory.
- ♦ The `ps` command provides a snapshot of the process status on the system.
- ♦ The `top` command provides real-time information on processes.
- ♦ The `kill` command is used to terminate processes.
- ♦ Processes can be run in the background using the `&` symbol and `bg` command.

- ♦ Processes in the foreground can be halted using the `^Z` command.
- ♦ Processes can be moved to the foreground using the `fg` command.
- ♦ The `nice` command is used to set a process priority when the process is started.
- ♦ The `renice` command is used to change the priority of processes already running on the system.



STUDY GUIDE

The following questions and exercises enable you to review the information covered in this chapter. Take your time when completing this section of the chapter, carefully reviewing any questions that you may have. Answering the question correctly is not as important as understanding the answer, so review any material that you might still be unsure of. Being comfortable with the questions and answers presented here will help you be more prepared for the certification exam questions.

Assessment Questions

1. Which shell is the default shell used on Linux systems?
 - A. `csh`
 - B. `rsh`
 - C. `bash`
 - D. `tcsh`
2. Which file contains the available shells for the system?
 - A. `/etc/passwd`
 - B. `/etc/command`
 - C. `/etc/bash`
 - D. `/etc/shells`
3. Which command is used to change the default shell to the Bourne-Again Shell version 2?
 - A. `chng -s /bin/bash2`
 - B. `chsh -s /bin/bash2`
 - C. `shell -c /bin/bash2`
 - D. `default -shell /bin/bash2`
4. What is not a correct format for entering commands at the command line?
 - A. *command*
 - B. *command options*
 - C. *command arguments options*
 - D. *command options arguments*

5. Multiple commands can be entered on the same command line when separated by which character?
 - A. .
 - B. ;
 - C. ,
 - D. \

6. Which key combination is used to allow a command to span multiple lines?
 - A. ""
 - B. \ Enter
 - C. / Enter
 - D. Tab-Enter

7. Which key, when pressed once, is used to provide command completion at the command line?
 - A. Tab
 - B. Esc
 - C. Enter
 - D. Ctrl

8. Which key, when pressed twice, is used to provide command completion at the command line?
 - A. Tab
 - B. Esc
 - C. Enter
 - D. Ctrl

9. The Readline Library uses which editor to provide command-line editing?
 - A. pico
 - B. vi
 - C. emacs
 - D. edit

10. Keyboard bindings can be viewed using which command?
- A. `bind -keys`
 - B. `bind -v`
 - C. `display -keys`
 - D. `display -v`
11. Which file contains the system-wide variables on a system with a bash shell?
- A. `/bash`
 - B. `/bin/bash`
 - C. `/etc/bash`
 - D. `/etc/profile`
12. _____ would run a script called `update_mozilla` stored in your home directory, from that directory.
13. When making changes to an environment variable what command must you use to ensure that those changes are available to the shell?
- A. `save`
 - B. `remember`
 - C. `export`
 - D. `echo`
14. Which environment variable is used to customize the shell prompt?
- A. `PS1`
 - B. `prompt`
 - C. `shell`
 - D. `display`
15. Which file contains the assignment of user's home directories?
- A. `/etc/home`
 - B. `/etc/profile`
 - C. `/etc/passwd`
 - D. `/etc/users`

16. Which command displays a listing of previously entered commands?
- A. `commands`
 - B. `review`
 - C. `history`
 - D. `export`
17. Which key allows you to view the last command entered?
- A. down arrow
 - B. up arrow
 - C. right arrow
 - D. left arrow
18. Which command allows you to use the default editor to edit several commands from the history file?
- A. `history`
 - B. `edit`
 - C. `fc`
 - D. `view`
19. It is possible to run commands that are not located in a directory listed in the `PATH` if you know the full path and command name.
- A. True
 - B. False
20. The _____ command is used to view a snapshot of the processes running on a system.
21. Which of the following is used to end the 408 process without allowing it to exit gracefully?
- A. `kill 408`
 - B. `kill -15 408`
 - C. `kill -1 408`
 - D. `kill -9 408`
22. The _____ command runs the `netscape` process in the background.

23. Which command is used to set the priority of a process when it is started?
- A. jobs
 - B. renice
 - C. nice
 - D. top

Scenarios

1. You have several scripts stored in a directory named `maintenance` located inside of your home directory. You wish to run these scripts from various locations on the system without entering the full path. How can you accomplish this?
2. You've noticed that several of your users have changed their default shells and your scripts no longer work. What can you do to prevent this from happening in the future?

Lab Exercises

Lab 2-1 Changing the Prompt

This lab will change the default prompt for users on your system to display the user name, hostname, and full working path of the `pwd`.

1. Log into the system as root.
2. Run the command `vi /etc/bashrc`.
3. Scroll to the entry `PS1=` using the arrow keys.
4. Use the `x` key to remove the current settings.
5. Insert text by typing the `I` key.
6. Type the following:

```
PS1="[ \u@\h \w] \\\$ "
```
7. Save and exit by hitting the `Esc` key followed by `:wq`
8. Log out and then log back in to verify your changes.

Answers to Chapter Questions

Chapter Pre-Test

1. The `/etc/passwd` file contains the default shell for each user.
2. Adding to the `PATH` variable by editing the `/etc/profile` file will add to the `PATH` of all users on the system.
3. The `HISTSIZE` variable sets the number of entries to store in the bash history file.
4. The `Tab` key pressed once and the `Esc` key pressed twice provide command completion, if there is only one possible command match.
5. The `emacs` editor is used as the default Readline Library editor.
6. The `fc` command allows you to edit entries from the `~/bash_history` file using the default system editor.
7. The `.bash_history` file, located in each user's home directory, contains the command history.
8. Variables can be viewed using the `echo` command, such as `echo $HOME`.
9. Programs started from `/etc/inittab` file operate using the `init PATH`.
10. To run a command from the `pwd`, you type the command preceded by `./`, as in `./commandname`.
11. Command substitution allows output from a command to be used instead of the command name.
12. The `-R` option allows recursive function of a command.
13. The `renice` command is used to change the priority of a process that is currently running on the system.
14. The `top` command is used to view real-time information on processes.
15. The `jobs` command is used to view the background jobs running on a system.

Assessment Questions

1. C. The default shell used on Linux systems is `bash`. See the "Understanding Shells" section for more information.
2. D. The `/etc/shells` file contains a listing of available shells. See the "Understanding Shells" section.
3. B. The `chsh` command is used to view and change shell settings. See the "Understanding Shells" section for more information.
4. C. Command options precede arguments when entered at the command line. See the "Using the Command Line" section for more information.

5. **B.** A semicolon (;) is used to separate multiple commands on a single command line. See the “Using the Command Line” section for more information.
6. **B.** The backslash (\) key causes whichever key follows it to be ignored. This allows the Enter key to be ignored and the command to be continued on the next line. See the “Using the Command Line” section for more information.
7. **A.** The Tab key provides command completion when pressed once. See the “Command completion” section for more information.
8. **B.** The Esc key provides command completion when pressed twice. See the “Command completion” section for more information.
9. **C.** By default, the `emacs` editor is used with the Readline Library. See the “Editing commands with the Readline Library” section for more information.
10. **B.** The `bind -v` command allows keyboard bindings to be viewed. See the “Editing commands with the Readline Library” section for more information.
11. **D.** The `/etc/profile` file stores the system-wide variable statements using the bash shell. See the “Understanding Shells” section for more information.
12. `./update_mozilla`. When running a command from the `pwd`, the command name is preceded by a period and slash (./), which specifies that the command is in the current directory. See the “Editing the PATH variable” section for more information.
13. **C.** The `export` command causes changes to an environment variable to be made available to the user’s shell. See the “Environment Variables and Settings” section for more information.
14. **A.** The `PS1` variable is used to make changes to the prompt. See the “Prompt” section for more information.
15. **C.** The `/etc/passwd` file contains the assignment of the user’s home directories. See the “Environment Variables and Settings” section for more information.
16. **C.** The `history` command displays the previously entered commands that are stored in the user’s `.bash_history` file. See the “Using the history file” section for more information.
17. **B.** The up arrow is used to cycle through previously entered commands. See the “Using the history file” section for more information.
18. **C.** The `fc` command allows you to edit previously entered commands using the default editor. See the “fc” section for more information.
19. **A.** Commands can be run using the full path as long as the user has the necessary file permissions. See the “Editing the PATH variable” section for more information.
20. `ps`. The `ps` command is used to view a snapshot of processes running on the system. See the “Managing Processes” section for more information.
21. **D.** The `-9` signal kills the process without allowing it to exit gracefully. See the “Managing Processes” section for more information.

22. **net** **scape** **&**. The **&** symbol is used to start a process as a background job. See the “Managing Processes” section for more information.
23. **C**. The **nice** command is used to change the priority level of a process when it is started. See the “Modifying Process Priorities” section for more information.

Scenarios

1. When logged in as the user, edit the `.profile` file, which is in the user's home directory. Add the following lines to the file:

```
PATH="$PATH:/home/username/maintenance"  
export PATH
```

Save the file, export, and then log out.

2. Change the users' shells back to the default by editing the `/etc/passwd` file. Then remove all entries from the `/etc/shells` file except for the default shell entry. Users will then be unable to change their shells. However, it is a better practice to specify the correct shell in the script. This is done at the first line in the script.

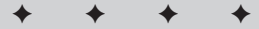
Installing Software

EXAM OBJECTIVES

Exam 102 ♦ General Linux, Part 2

2.2 Installation and Package Management

- **Make and install programs from source.** Manage (compressed) archives of files (unpack “tarballs”), specifically GNU source packages. Install and configure these on your systems. Do simple manual customization of the `Makefile` if necessary (like paths, extra include dirs) and make and install the executable. Involves using the commands: `gunzip`, `tar`, `./configure`, `make`, `make install`. Involves editing the files: `./Makefile`
- **Manage shared libraries.** Determine the dependencies of executable programs on shared libraries, and install these when necessary. Involves using the commands: `ldd`, `ldconfig`. Involves editing the files: `/etc/ld.so.conf`
- **Use Debian package management.** Use the Debian package management system, from the command line (`dpkg`) and with interactive tools (`dselect`). Be able to find a package containing specific files or software; select and retrieve them from archives; install, upgrade or uninstall them; obtain status information like version, content, dependencies, integrity, installation status; and determine which packages are installed and from which package a specific file has been installed. Be able to install a non-Debian package on a Debian system. Involves using the commands and programs: `dpkg`, `dselect`, `apt`, `apt-get`, `alien`. Involves reviewing or editing the files and directories: `/var/lib/dpkg/*`.



EXAM OBJECTIVES (CONTINUED)

- **Use Red Hat Package Manager (rpm).** Use `rpm`, from the command line. Familiarize yourself with these tasks: Install a package, uninstall a package, determine the version of the package and the version of the software it contains, list the files in a package, list documentation files in a package, list configuration files or installation or uninstallation scripts in a package, find out for a certain file from which package it was installed, find out which packages have been installed on the system (all packages, or from a subset of packages), find out in which package a certain program or file can be found, verify the integrity of a package, verify the PGP or GPG signature of a package, upgrade a package. Involves using the commands and programs: `rpm`, `grep`

CHAPTER PRE-TEST

1. Which script do you run to generate a `Makefile` for your system?
2. Which command is used to update the `ld.so.cache` file?
3. What is the tool that is used to convert package formats?
4. Name three sources of packages that the `apt-get` tool can use?
5. Which command option is used to upgrade a package using `rpm`?
6. Where is the RPM package database stored?
7. How do you clear the `apt-get` package cache?
8. Which file do you edit to add sources to `apt-get`?
9. Which tool is used to authenticate packages in RPM format?
10. Which Debian tool is used to query a package's status?

Chapter 3 covers the process of managing software on your Linux systems. Adding new software to your system is something you will probably do a lot of. In the beginning, the standard way of distributing software was via source code, which the user had to compile, or precompiled binary packages with no standard installation method. While this is still a common method, a couple of package management systems now exist that make this job much easier. They resolve file conflicts and alert the user to missing dependencies. The three major types of packages are tarballs, Debian `.deb` packages, and Red Hat RPM packages. For the exam you will need to be familiar with all three. The key to these is found by remembering which tools to use in a situation and the appropriate options to go with those tools.

The three distribution methods covered in this chapter are as follows:

- ♦ Tarball of source code
- ♦ Red Hat Package Manager (RPM)
- ♦ Debian's `.deb` format

Installing Software from Source Code

Objective

2.2 Installation and Package Management

- **Make and install programs from source.** Manage (compressed) archives of files (unpack “tarballs”), specifically GNU source packages. Install and configure these on your systems. Do simple manual customization of the `Makefile` if necessary (like paths, extra include dirs) and make and install the executable. Involves using the commands: `gunzip`, `tar`, `./configure`, `make`, `make install`. Involves editing the files: `./Makefile`

Installing software via compiling the source code is still popular today in Linux and other UNIX operating systems. All major Linux distributions now support a package management system in one form or another. The benefit of distributing software in source form is that it can be compiled on many different platforms.

Why would you get source code when you can download a *binary package*, which includes the software precompiled? Many people feel that the source code is more secure since they can read it and look for backdoors or other security flaws. Of course, some people don't have the knowledge to check the code or the time to do so. In many cases binary packages come out some time later than the source code. On major applications it may be as long as a few days, but on more obscure packages it may be weeks until the newest version is packaged up.

Source code for applications is available at sites all over the Internet. They can be found at many of the Linux software sites such as `www.freshmeat.net`. The software is normally distributed in a `.tar.gz` file, which is a compressed file that must be decompressed and compiled. Normally, files that tell you how to install and configure the software, usually called `README` and `INSTALL`, are included in the package. Some source code includes a preconfigured `Makefile`, while others use a script to generate it for each system. The `Makefile` is a large text file that has the instructions to tell the compiler how to compile the application. The normal steps to install software using this method are as follows:

1. Obtain the source code from an Internet site.
2. Decompress the software to a temporary directory.
3. Read the included `INSTALL` and `README` files.
4. Run the `./configure` script, if included.
5. Make any changes to the `Makefile`, if needed.
6. Compile the software using the `make` command.
7. Install the software using the `make install` command, if included.

The following example walks through the steps to install the software and provides more details.

Obtaining the source code

Normally source code is transferred via FTP from a site on the Internet or downloaded via the Web. The following example shows the download of `httperf`, a Web benchmarking tool that follows the standard method of compiling software. Some information is edited for brevity.

```
[root@redhat /root]# ftp ftp.hp1.hp.com
Connected to butler.hp1.external.hp.com.
220
220-
220- All logins and file transfers on this system are actively
monitored
220- and logged.
220-
220- Please contact <root@butler.hp1.hp.com> with any technical
problems
220- accessing this archive.
220-
220- The current time is Sat Nov 18 12:46:51 2000.
220-
220 butler.hp1.external.hp.com FTP server (Version 1.1.214.4
Mon Feb 15 08:48:46
GMT 1999) ready.
```

```
Name (ftp.hpl.hp.com:jason): anonymous
331 Guest login ok, send your complete e-mail address as
password.
Password: <Email Address>
230
230- You are currently user 7 out of a maximum 40 users.
230-
230 Guest login ok, access restrictions apply.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd pub/httpperf
250- it was last modified on Tue Oct 31 15:53:06 2000 - 18
days ago
250 CWD command successful.
ftp> get httpperf-0.8.tar.gz
local: httpperf-0.8.tar.gz remote: httpperf-0.8.tar.gz
200 PORT command successful.
150 Opening BINARY mode data connection for httpperf-0.8.tar.gz
(146107 bytes).
226 Transfer complete.
146107 bytes received in 1.42 secs (1e+02 Kbytes/sec)
ftp> quit
221 Goodbye.
```

Decompressing the tarball

The file you download is normally a *tarball*, a collection of files grouped together with the `tar` utility and sometimes compressed with `gzip`. In this example you can use either the `gunzip` or `gzip -d` commands to uncompress them:

```
[root@redhat /root]# gunzip httpperf-0.8.tar.gz
```

Once the package is uncompressed, it needs to be `untar'd` to expand the individual files.

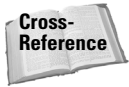
```
[root@redhat /root]# tar xvf httpperf-0.8.tar
httpperf-0.8/
httpperf-0.8/timer.h
httpperf-0.8/core.c
httpperf-0.8/httpperf.c
httpperf-0.8/core.h
httpperf-0.8/event.h
httpperf-0.8/httpperf.h
...
httpperf-0.8/object.c
httpperf-0.8/sess.h
httpperf-0.8/sess.c
```

```

httpperf-0.8/Makefile.in
httpperf-0.8/configure
httpperf-0.8/install-sh
httpperf-0.8/mkinstalldirs
httpperf-0.8/config.sub
httpperf-0.8/configure.in
httpperf-0.8/config.h.in
httpperf-0.8/config.guess
httpperf-0.8/aclocal.m4
httpperf-0.8/idleconn.c
[root@redhat /root]# cd httpperf-0.8

```

Since most Linux distributions include a version of `tar` that can handle gzip compression, you can also use the `tar zxvf` command to uncompress and untar the files in one step.



The compression utilities are covered in detail in Chapter 6.

Next, you should change to the directory that was just created when the files were untar'd. Be sure and read any documentation that comes with the source code. The important files are usually named `README` and `INSTALL`. They explain how to compile and configure the software.

Running the configure script

Compiling software today is much easier than it used to be, thanks to the `./configure` script distributed with most applications. This script automatically configures the `Makefile` for the current system. In the days before this script, the administrator had to manually edit the `Makefile` to point to the correct files and libraries installed on the system and to make any adjustments for the compiler. Along with the automatic configuration, the `./configure` script usually gives much better feedback of a problem than a long cryptic compiler error would. The following is an example showing the `configure` script from the `httpperf` application decompressed in the previous section.

Tip

Some source code uses a script named `Configure` instead of `configure`.



```

[root@redhat httpperf-0.8]# ./configure
creating cache ./config.cache
checking for gcc... gcc
checking whether the C compiler (gcc ) works... yes
checking whether the C compiler (gcc ) is a cross-compiler...
no
checking whether we are using GNU C... yes
checking whether gcc accepts -g... yes
checking for a BSD compatible install... /usr/bin/install -c

```

```
checking whether make sets ${MAKE}... yes
checking for ranlib... ranlib
checking how to run the C preprocessor... gcc -E
checking whether gcc needs -traditional... no
checking host system type... i686-pc-linux-gnu
checking target system type... i686-pc-linux-gnu
checking build system type... i686-pc-linux-gnu
checking for main in -lm... yes
checking for main in -lcrypto... yes
checking for SSL_version in -lssl... yes
checking for ANSI C header files... yes
checking for fcntl.h... yes
checking for sys/ioctl.h... yes
checking for sys/time.h... yes
checking for unistd.h... yes
checking for openssl/ssl.h... no
checking for working const... yes
checking for size_t... yes
checking for long long type... yes
checking whether time.h and sys/time.h may both be included...
yes
checking size of long... 4
checking for u_char... yes
checking for u_short... yes
checking for u_int... yes
checking for u_long... yes
checking for working alloca.h... yes
checking for alloca... yes
checking for unistd.h... (cached) yes
checking for getpagesize... yes
checking for working mmap... yes
checking return type of signal handlers... void
checking for vprintf... yes
checking for gettimeofday... yes
checking for select... yes
checking for socket... yes
checking for strdup... yes
checking for strerror... yes
checking for strtod... yes
checking for strtol... yes
checking for strtoul... yes
checking for getopt_long... yes
updating cache ./config.cache
creating ./config.status
creating stat/Makefile
creating lib/Makefile
creating Makefile
creating gen/Makefile
creating config.h
```

As you can see, the script goes through many checks to make sure the system is capable of compiling the application. Even with all these tests you may still encounter problems during the compile, but far fewer than you used to encounter. The `./configure` script usually takes several options to pass any special information for the tests it performs or to change the configuration of the software. Usually these options are included in the `README` and `INSTALL` files and vary from one application to the next. For all the supported options with a package you can run `./configure -help`. In the example in this section you could have enabled debug mode for the application using the following:

```
./configure --enable-debug
```

Also, many packages support the `--prefix=` `configure` option that is used to change the path to where the software is installed. Note that these are compile time options and usually require a recompile to change.

After you have run the `configure` script, you'll have at least one file called `Makefile` in the root directory of the source code tree. Others may be created in subdirectories off this tree, depending on the application. This is the file that the software compiler uses to know what to do. It also dictates how and where the software is installed.

Making changes to the Makefile

In some cases you may need to make changes to the created `Makefile`, though this is needed less often now that the `configure` script is used. The following is an example of the beginning of a `Makefile`.

```
# Generated automatically from Makefile.in by configure.
SHELL=/bin/sh

srcdir = .
top_srcdir = .
top_builddir = .

prefix = /usr/local
bindir = ${exec_prefix}/bin
mandir = ${prefix}/man

#DEFS += -DTIME_SYSCALLS
#DEFS += -DDONT_POLL

SUBDIRS = lib gen stat

CC = gcc
RANLIB = ranlib
MKDIR = $(top_srcdir)/mkinstalldirs
INSTALL = /usr/bin/install -c
INSTALL_PROGRAM = ${INSTALL}
INSTALL_DATA = ${INSTALL} -m 644
```

All of the user definable options are at the beginning of the `Makefile` and have the syntax of `Option = Value`. The options shown will vary in different applications, but they are usually self-explanatory. Some example options from this file are as follows:

```
prefix = /usr/local
```

This defines the directory in which the software will be installed.

```
bindir = ${exec_prefix}/bin
```

This defines where the binary executable file will be installed. In this case it shows that the `/bin` directory will be added to the `prefix` option or `/usr/local/bin`.

```
mandir = ${prefix}/man
```

This line specifies which directory the man pages, which provide documentation for the application, will be installed to. Again, this shows the `/man` directory appended to the `prefix` value or `/usr/local/man`.



Caution

After you change the `Makefile` do not run the `configure` script again or your changes will be overwritten.

Compiling the software

The next step is to actually compile the software. The compile process is started with the simple command `make`. The `make` command uses the `Makefile` to instruct the compiler on how to operate. The following is an example:

```
[root@redhat httpperf-0.8]# make
making all in lib
make[1]: Entering directory `/root/httpperf-0.8/lib'
gcc -c -DHAVE_CONFIG_H -I.. -I.. -I../lib -DNDEBUG -
D_GNU_SOURCE -D_XOPEN_SOURC
E -g -O2 -Wall getopt.c
gcc -c -DHAVE_CONFIG_H -I.. -I.. -I../lib -DNDEBUG -
D_GNU_SOURCE -D_XOPEN_SOURC
E -g -O2 -Wall getopt1.c
gcc -c -DHAVE_CONFIG_H -I.. -I.. -I../lib -DNDEBUG -
D_GNU_SOURCE -D_XOPEN_SOURC
E -g -O2 -Wall ssl_writev.c
ar r libutil.a getopt.o getopt1.o ssl_writev.o
ranlib libutil.a
make[1]: Leaving directory `/root/httpperf-0.8/lib'
making all in gen
make[1]: Entering directory `/root/httpperf-0.8/gen'
gcc -c -DHAVE_CONFIG_H -I.. -I. -I.. -I../lib -DNDEBUG -
D_GNU_SOURCE -D_XOPEN_S
OURCE -g -O2 -Wall call_seq.c
```

```

...
-02 -Wall timer.c
gcc -o httpperf httpperf.o object.o call.o conn.o sess.o core.o
event.o http.o ti
mer.o \
    gen/libgen.a stat/libstat.a lib/libutil.a -lssl -
lcrypto -lm
gcc -c -DHAVE_CONFIG_H -I. -I. -I./lib -DNDEBUG -D_GNU_SOURCE -
D_XOPEN_SOURCE -g
-02 -Wall idleconn.c
gcc -o idleconn idleconn.o -lssl -lcrypto -lm

```

Once this command completes you will be left with a compiled application, but one that is not yet installed. The next step installs the application.

Installing the software

The final step is to install the software. This is done using the `make install` command that most Makefiles include. The destination directory can usually be set with a `./configure` option. The `make install` command will install the appropriate binary and documentation files, including man pages.



Most of the `install` commands can be run as a normal user and do not require root. Only when installing software to a secured directory is root required.

```

[root@redhat httpperf-0.8]# make install
making install in lib
make[1]: Entering directory `/root/httpperf-0.8/lib'
make[1]: Nothing to be done for `install'.
make[1]: Leaving directory `/root/httpperf-0.8/lib'
making install in gen
make[1]: Entering directory `/root/httpperf-0.8/gen'
make[1]: Nothing to be done for `install'.
make[1]: Leaving directory `/root/httpperf-0.8/gen'
making install in stat
make[1]: Entering directory `/root/httpperf-0.8/stat'
make[1]: Nothing to be done for `install'.
make[1]: Leaving directory `/root/httpperf-0.8/stat'
./mkinstalldirs /bin /usr/local/man/man1
/usr/bin/install -c httpperf /bin/httpperf
/usr/bin/install -c -m 644 ./httpperf.man
/usr/local/man/man1/httpperf.1
[root@redhat httpperf-0.8]#

```

You're done! Now you can use your new application. The `README` file should specify the name of the executable file to run and where it is installed to by default.

Managing Shared Libraries

Objective

2.2 Installation and Package Management

- **Manage shared libraries.** Determine the dependencies of executable programs on shared libraries, and install these when necessary. Involves using the commands: `ldd`, `ldconfig`. Involves editing the files: `/etc/ld.so.conf`

When source code is written, most programmers do not rewrite code for common operations. These operations are written once and used again by many other programmers. These common operations are stored in what are called *libraries*. For an application to successfully compile and run, it needs access to the libraries it was written to use. Some source code compiles the libraries into the final executable so the executables do not need the libraries to run. These are known as *statically compiled* applications. Other applications are linked only to the libraries during compile and must have access to the library files later. These are known as *dynamically compiled* applications. While it would seem advantageous to statically compile every application, statically compiling makes the application much larger than a dynamically compiled version.

Linux's shared libraries are stored in several paths, such as the following:

- ♦ `/lib` — Main shared libraries
- ♦ `/usr/lib` — Supplemental libraries
- ♦ `/usr/X11R6/lib` — Shared libraries for X Window

Shared libraries usually follow the naming convention of:

libraryname-major-minor-patch.so

The `so` extension stands for “shared object.” In the example of `libcrypt-2-1-3.so`, you can tell that the following:

- ♦ The name of the library is `libcrypt`.
- ♦ The major version is 2.
- ♦ The minor version is 1.
- ♦ The patch level is 3.

In many cases symbolic links to the shared library file will exist. These are named as follows:

- ♦ *libraryname.so*
- ♦ *libraryname.so.major*

Using our example above, two links may exist with the names of:

- ♦ `libcrypt.so`
- ♦ `libcrypt.so.2`

These links allow software to point to them instead of the actual file. This way the minor version and patch level can change without breaking the association.

Viewing required shared libraries

If you want to see which libraries an application uses, use the following command:

```
ldd file_name
```

For example, to see which libraries `wget` uses:

```
norbert:/usr/bin# ldd wget
      libc.so.6 => /lib/libc.so.6 (0x40020000)
      /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
```

This means that `wget` uses the libraries `libc.so`, version 6, and `ld-linux.so`, version 2.

Setting library paths

If an application cannot find a shared library it requires, it will give an error and exit. If the library is not in a standard path, you can add the nonstandard path to the `LD_LIBRARY_PATH` environment variable to have it searched. For example:

```
Export LD_LIBRARY_PATH=/usr/newpath
```

would add the `/usr/newpath` to the list of searched directories for libraries.

Configuring shared libraries

If a shared library is installed manually, not by a package service such as RPM, you must manually inform the system to use the new libraries. The library configuration file is named `/etc/ld.so.conf` and contains a list of directories to be searched. If you manually install a library into a new directory not listed in the `ld.so.conf` file, you should add the new directory to the file. An example of this file is as follows:

```
/usr/lib
/usr/X11R6/lib/Xaw3d
/usr/X11R6/lib
```

Notice that the `/lib` directory is not listed; it is included by default since system essential libraries reside there. To enhance system performance a cache file is

created that holds all of the libraries in these directories. Any time that the `ld.so.conf` file is changed, this cache file needs to be updated. Updating the cache file, `/etc/ld.so.cache`, is done using this command:

```
ldconfig
```

An optional `-v` parameter lists all libraries found in the directories as they are scanned.



The `ld.so.cache` file is created from the `ld.so.conf` file by using `ldconfig`.

Red Hat Package Manager

Objective

2.2 Installation and Package Management

- **Use Red Hat Package Manager (rpm).** Use `rpm`, from the command line. Familiarize yourself with these tasks: Install a package, uninstall a package, determine the version of the package and the version of the software it contains, list the files in a package, list documentation files in a package, list configuration files or installation or uninstallation scripts in a package, find out for a certain file from which package it was installed, find out which packages have been installed on the system (all packages, or from a subset of packages), find out in which package a certain program or file can be found, verify the integrity of a package, verify the PGP or GPG signature of a package, upgrade a package. Involves using the commands and programs: `rpm`, `grep`

The example from the previous section showing how to compile a package distributed as source code worked well, but that is not always the case. It sometimes takes a lot of effort to get a piece of code to compile on a system, and unless you need to make changes or want to read the code, there really is no advantage to doing so. Most software is now also distributed in binary form, already compiled.

The most popular package management system in use with Linux today is RPM, or Red Hat Package Manager. Even though Red Hat created it, it is used on all major distributions by default except for Slackware and Debian. RPM is one of Red Hat's most well-known contributions to the open source community and one that has saved many administrators time and effort.

A package management system enhances a binary distribution by managing version control, dependencies on other packages, and administration. By using the package tools, you can check which version was installed, which files were included in the package, and more. RPM is made up of several parts, which are as follows:

- ♦ Package files (`*.rpm`)
- ♦ The RPM database
- ♦ The `rpm` tool

Package files

RPM files are distributed for most applications now. An RPM file includes several pieces, including the following:

- ♦ Compressed application files
- ♦ Name and version of the package
- ♦ Build date and the build host
- ♦ Description of the package and application
- ♦ Information on who built the package
- ♦ MD5 checksum to verify the package integrity
- ♦ Other required packages (dependencies)

As you can see an RPM package contains a lot more information than a tarball. Along with the required files, it includes all information needed to install and maintain the package. RPM packages have a standard naming scheme:

```
package-version-patch.architecture.rpm
```

where:

- ♦ *package* — Name of the application installed by the package.
- ♦ *version* — Version number of the application.
- ♦ *patch* — Patch number of build of this package. If a small change is made or the package maintainer makes a change to the package, this number is incremented.
- ♦ *architecture* — The computer architecture that the package was compiled for. This is very important now that Linux runs on so many different types of computers. Some architecture examples are as follows: *i386*, *i586*, and *i686* for Intel x86 and compatible; *sparc* for Sun SPARC; and *alpha* for Digital/Compaq Alpha.

As an example, take this package:

```
ethereal-0.8.9-1.i386.rpm
```

This package contains version 0.8.9 of Ethereal, a packet sniffer used to troubleshoot a network. It is the first build of this package, and it is for the i386 (Intel PC) platform.



A popular site to find RPM packages of most applications is www.rpmfind.net.

The RPM database

Information is kept about every package installed on the system in a database. The database is kept in the `/var/lib/rpm` directory. This data is used to match up dependencies, to check for files that already exist, and to verify installed packages. Almost every time that the `rpm` command is used the database is consulted.

In normal operations you do not need to worry about the RPM database. Occasionally the database will have inconsistencies and will need to be rebuilt using the `rpm` command with `-rebuilddb`:

```
rpm --rebuilddb
```

When inconsistencies or corruption occurs, you may receive strange errors when adding or removing packages.

The rpm tool

The `rpm` tool is only one command-line tool to worry about with RPM, so no chance of confusion exists. The downside is that there are many options to use with it to cover all possible circumstances. The `rpm` command is used to:

- ♦ Install packages
- ♦ Upgrade packages
- ♦ Remove and uninstall packages
- ♦ Query the RPM database for information
- ♦ Verify the package file
- ♦ Check installed files
- ♦ Build a binary package from source

The `rpm` tool performs several sanity checks when doing an install, uninstall, or upgrade. This helps to protect against installing an unusable package or damaging another application. These checks include checking the following:

- ♦ That enough free disk space exists for the package
- ♦ That existing files will not be overwritten
- ♦ That all dependencies are met

Validating package integrity



2.2 Installation and Package Management

- Use Red Hat Package Manager (`rpm`). Familiarize yourself with these tasks:
 - Verify the integrity of a package, verify the PGP or GPG signature of a package.

RPM includes functions to let you check the integrity of a package to make sure it was downloaded correctly and not tampered with. This is done by using the MD5 algorithm and the GnuPG tool. MD5 is a mathematical algorithm that is used to make sure a file has not been modified. When a file is checked, the algorithm will output a checksum number, and if this number matches the one generated by the file after you download it, you know the file was not corrupted. The GnuPG tool is a public key encryption package that can be used to check the authenticity of the source of a file or document and to encrypt communications. GnuPG is installed by default on Red Hat systems.

The `-K` or `--checksig` options validate the integrity of a package using MD5 and/or GnuPG.

```
rpm -K package_file.rpm
```

For this to work, you must perform the following steps first:

1. Install the GnuPG application if it is not currently installed. It is available from www.gnupg.org.
2. Retrieve the public key for the package maintainer of the application you want to check. This is usually available on the application's Web or FTP sites. For example, Red Hat's is available on its FTP site and is named RPM-GPG-KEY.
3. Add the public key for the appropriate package maintainers to your key ring using the `gpg -import` command. For example:

```
[root@redhat /root]# gpg --import RPM-GPG-KEY
gpg: key DB42A60E: public key imported
gpg: /root/.gnupg/trustdb.gpg: trustdb created
gpg: Total number processed: 1
gpg:          imported: 1
```

If the package validates correctly, `rpm` will output a message similar to the following:

```
[root@redhat /root]# rpm -K wget-1.5.3-6.src.rpm
wget-1.5.3-6.src.rpm: md5 gpg OK
```

If the package isn't valid, you'll get a message like this:

```
[root@redhat /root]# rpm -K wget-1.5.3-10.i386.rpm
wget-1.5.3-10.i386.rpm: rpmReadSignature failed
```



Some packages use PGP to check integrity while others use GnuPG. PGP is available from www.pgpi.com.

Installing packages

Objective

2.2 Installation and Package Management

- Use Red Hat Package Manager (rpm). Familiarize yourself with these tasks:
Install a package.

Red Hat Package Manager makes installing new software very easy. The command to add a new package to the system can be as simple as the following:

```
rpm -i package_file.rpm
```

or:

```
rpm --install package_file.rpm
```

**Exam Tip**

Many options contain a long and short version, for example `-i` and `--install`. Be sure to know both versions for the exam, and remember they are case-sensitive.

The rpm tool can also install multiple packages at one time. For example:

```
rpm -i first_package_file.rpm second_package_file.rpm  
third_package_file.rpm
```

or:

```
rpm -i *.rpm
```

**Tip**

You can use wildcards when installing or upgrading packages, but not when removing packages.

Installing multiple packages at one time is useful when a package is installed along with any needed dependencies. All of the standard sanity checks are done for each package that is installed. Be warned that if one package fails, it can stop the entire command from executing, and no packages will be installed.

The rpm tool can also get packages for you from the Internet, which may save you a step or two. The location is given with the same address format as with a Web browser. Both anonymous and login FTP are supported. For example:

```
rpm -i ftp://rpmfind.net/linux/redhat/redhat-  
7.0/i386/en/RedHat/RPMS//libpcap-0.4-29.i386.rpm
```

or:

```
rpm -i http://rpmfind.net/linux/redhat/redhat-  
7.0/i386/en/RedHat/RPMS//libpcap-0.4-29.i386.rpm
```

Several useful options to add when installing packages are `-v` and `-h` (or `--hash`). The `-v` option shows the name of the package being installed.

```
[root@redhat /root]# rpm -iv libpcap-0.4-19.i386.rpm
libpcap-0.4-19
```

The `-h` option shows hash marks as the package is installed to show status. This is useful with very large packages.

```
[root@redhat /root]# rpm -ivh libpcap-0.4-19.i386.rpm
libpcap
#####
```

As previously stated, `rpm` checks for needed dependencies before installing or removing any software. If a problem is encountered, you will see an error message, such as the following:

```
[root@redhat /root]# rpm -ivh ethereal-0.8.9-1.i386.rpm
error: failed dependencies:
        libpcap >= 0.4 is needed by ethereal-0.8.9-1
[root@redhat /root]#
```

The package `ethereal-0.8.9-1.i386.rpm` requires `libpcap` version 0.4 or greater.

If a package tries to overwrite an existing file, you will receive an error similar to this:

```
[root@redhat /root]# rpm -ivh libpcap.rpm
file /usr/lib/libpcap.a from install of libpcap-0.4a6-35
conflicts with file from package libpcap-0.4-19
file /usr/man/man3/pcap.3.gz from install of libpcap-0.4a6-35
conflicts with file from package libpcap-0.4-19
[root@redhat /root]#
```

By default, `rpm` will not let you overwrite a file from another package.

Suppose that you have a very good reason to go ahead with an operation even though `rpm` gives you a warning. The `rpm` tool provides several overwrite options that you can manually specify. These are as follows:

- ♦ `--force` — Forces `rpm` to overwrite existing packages or files.
- ♦ `--nodeps` — Bypasses dependency checking. Useful if you have installed a dependency by other means, such as compiling from source.
- ♦ `--replacefiles` — Overwrite files owned by other packages.

Using the previous example, assume that you have compiled the `libpcap` library from source code and not from an RPM. To install the `Ethereal` package you would use the following:

```
rpm -ivh --nodeps ethereal-0.8.9-1.i386.rpm
```

The package will now be installed without producing any errors. For the application itself to run correctly the needed dependency must be installed correctly. Overriding warnings can be a very risky measure, so be careful and consider the consequences on important packages.

Upgrading packages

Objective

2.2 Installation and Package Management

- **Use Red Hat Package Manager (rpm).** Familiarize yourself with these tasks: Upgrade a package.

One constant in the world of software is that there will always be updates and new versions. RPM makes upgrading easy by handling the removal of the old version and the install of the new version in one step. To upgrade a package simply use the `-U` option:

```
rpm -U package_file.rpm
```

or:

```
Rpm --upgrade package_file.rpm
```

As with the install options it is recommended that you use the `-v` and `-h` parameters. Since the upgrade option first removes the old version, it will save any modified `config` files with the `.rpmsave` extension. The second phase then installs the new version of the package and performs the standard checks.

If an older version of the package is not already installed, `rpm` will go ahead and install the new version. Many administrators simply use the `-U` option to do any install or upgrade. This way they are always covered since `rpm` tries to upgrade first.

Another useful option is `-F` (or `--freshen`), which upgrades packages only if there is an older version installed. This can be a fast way to update a large number of packages on a system at once.

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

This command attempts to upgrade any installed package with a newer package in the current directory. Combine this with the fact that most distributions maintain a directory on their site with packages that have been updated since the initial release, and you have a good way to keep your system current.

Removing packages

Objective

2.2 Installation and Package Management

- Use Red Hat Package Manager (rpm). Familiarize yourself with these tasks: Uninstall a package.

You remove a package with the `-e` or `--uninstall` options:

```
rpm -e package_name
```

or:

```
rpm --uninstall package_name
```



Tip

When removing a package, remember to use the package name, not the filename. Wildcards also do not work when removing packages.

Both options perform the same functions; some people just find the `-uninstall` easier to remember. To help memorize `-e`, consider its standing for *erase*. When a package is removed, rpm does the normal dependency checks and will not let you remove a package if another depends on it, by default.

```
[root@redhat jason]# rpm -e libpcap
error: removing these packages would break dependencies:
libpcap >= 0.4 is needed by ethereal-0.8.9-1
```

The `--nodeps` option can be used to override these warnings. Remember that when removing a package you need to specify the package name, not the filename that was installed. Also, wildcards will not work. When a package is removed, rpm saves any configuration files that were changed from the default. This way you can reinstall the package later without needing to reconfigure it.

```
[root@redhat jason]# ls /etc/pine*
/etc/pine.conf
[root@redhat jason]# rpm -e pine
[root@redhat jason]# ls /etc/pine*
/etc/pine.conf.rpmsave
[root@redhat jason]#
```

Querying the RPM database

Objective

2.2 Installation and Package Management

- Use Red Hat Package Manager (rpm). Familiarize yourself with these tasks: Determine the version of the package and the version of the software it contains, list the files in a package, list documentation files in a package, list configuration files or installation or uninstallation scripts in a package, find out for a certain file from which package it was installed, find out which packages have been installed on the system (all packages, or from a subset of packages), find out in which package a certain program or file can be found.

The RPM database stored in `/var/lib/rpm` holds information about every package installed on the system. You can expose this data to gather information to help maintain your systems. All of the query options are done using the `rpm` command with the `-q` option.

Listing installed packages

The most basic query is to check the version of an installed package:

```
rpm -q package_name
```

or:

```
rpm --query package_name
```

For example:

```
[root@redhat /root]# rpm -q kernel
kernel-2.2.14-5.0
```

To list every package installed on your system use the `-a` option:

```
rpm -qa
```

Combine this with `grep` to see which groups of packages you have installed. The `grep` command searches for specific text patterns in a file, and the pipe symbol (`|`) redirects the output from one command to another command. The following example uses `rpm -qa` to find all the packages, and `grep` searches that list for the packages that contain the word *kernel*:

```
[root@redhat /root]# rpm -qa | grep kernel
kernel-headers-2.2.14-5.0
kernel-2.2.14-5.0
kernel-pcmcia-cs-2.2.14-5.0
kernel-utils-2.2.14-5.0
```



The `grep` command and pipes are discussed in detail in Chapter 4.

Checking the package that installed a file

There may be a time when you are not sure which package installed a file and you need to know. The `-f` option provides this information.

```
rpm -qf filename
```

For example:

```
[root@redhat /root]# rpm -qf /etc/bashrc
bash-1.14.7-22
```

This tells you that the `bashrc` file was installed by the package `bash-1.14.7-22`.

Listing files in a package

To list all the files installed by a package use the `-l` option.

```
rpm -ql package_name
```

For example, to list all files in the `openssh-clients` package you would use the following:

```
[root@redhat /root]# rpm -ql openssh-clients
/etc/ssh/ssh_config
/usr/bin/slogin
/usr/bin/ssh
/usr/bin/ssh-add
/usr/bin/ssh-agent
/usr/man/man1/slogin.1.gz
/usr/man/man1/ssh-add.1.gz
/usr/man/man1/ssh-agent.1.gz
/usr/man/man1/ssh.1.gz
```

To list the files that will be installed with a package use the additional `-p` option.

```
rpm -qpl package_file.rpm
```

Displaying package information

To print the description and other information about a package use the `-i` option.

```
rpm -qi package_name
```

For example, to display the information about the installed Linux kernel you would do the following:

```
[root@redhat /etc]# rpm -qi kernel
Name           : kernel                      Relocations: (not relocateable)
Version        : 2.2.14                  Vendor: Red Hat, Inc.
Release        : 5.0                     Build Date: Tue 07 Mar 2000 09:13:08
             PM EST
Install date: Wed 01 Nov 2000 06:58:30 PM EST      Build Host: porky.devel.redhat.com
Group          : System Environment/Kernel      Source RPM: kernel-
2.2.14-5.0.src.rpm
Size           : 11973135                 License: GPL
Packager       : Red Hat, Inc. <http://bugzilla.redhat.com/bugzilla>
Summary        : The Linux kernel (the core of the Linux operating system).
Description    :
The kernel package contains the Linux kernel (vmlinuz), the core of your
Red Hat Linux operating system. The kernel handles the basic functions
of the operating system: memory allocation, process allocation, device
input and output, etc.
```

If you want to display this information about a package you have not yet installed, use the `-p` option.

```
rpm -qip package_file.rpm
```

Printing package scripts

Some packages include scripts that run before or after they are installed. To display these scripts use the following command:

```
rpm -qp --scripts package_file.rpm
```

For example, to show the scripts that run when the package `kernel-2.2.14-5.0.i686.rpm` is installed, you would execute the following:

```
[root@redhat /root]# rpm -qp --scripts
/mnt/cdrom/RedHat/RPMS/kernel-2.2.14-5.0.i386.rpm
preinstall script (through /bin/sh):
/sbin/modprobe loop 2> /dev/null > /dev/null
exit 0
postinstall script (through /bin/sh):
cd /boot
ln -sf vmlinuz-2.2.14-5.0 vmlinuz
ln -sf System.map-2.2.14-5.0 System.map
ln -sf module-info-2.2.14-5.0 module-info

# Allow clean removal of modules directory
preuninstall script (through /bin/sh):
rm -f /lib/modules/2.2.14-5.0/modules.dep
```

Verifying package files

You sometimes need to check if a file has changed since a package was installed. You may have made a change to some configuration file and now need to know which was modified. The `rpm` tool provides this functionality with the `-V` verification option.

```
rpm -V package_name
```

When a verify is performed, `rpm` checks the files that were installed for a number of characteristics and displays those that have changed, and the items that have changed. Each file has an entry in the RPM database that stores its MD5 checksum, file size, symbolic link pointer, modification time, user and group owners, and permissions and file types. If the file is a device, the major and minor device numbers are checked as well. Table 3-1 shows the output given by the verification for any file that has changed.

Table 3-1
Package Verification Characteristics

<i>Item</i>	<i>Meaning</i>
.	No change for this characteristic
5	The MD5 checksum has changed
S	File size has changed
L	Symbolic link has changed
T	Modification time has changed
D	The device major and/or minor number has changed
U	User owner has changed
G	group owner has changed
M	Permission and/or file type has changed

For example, to see which configuration files installed by the `setup` package have changed you would enter the following:

```
[root@redhat /etc]# rpm -V setup
S.5....T c /etc/hosts.allow
.....G. c /etc/profile
S.5....T c /etc/services
```

This output shows that for the `hosts.allow` file and the `services` file, the file size, MD5 checksum, and modification time have changed. For the `profile` file, the group owner has changed, but nothing else.

To verify a package by using the package filename, use the `-p` option.

```
rpm -Vp package_file.rpm
```

To verify every package installed on the system use the `-a` option. This is a fast way to see which files have changed since the system was installed.

```
rpm -Va
```

Creating binary packages from source packages

Not all RPM packages distribute binary files. Some distribute the source code and install scripts that let you build your own binary RPMs. These are useful when you want to optimize or change a package. These packages have a slightly different naming scheme, since they are not dependent on a particular system architecture.

```
package-version-patch.src.rpm
```

Source RPMs are installed just like binary packages, with the `rpm -i` command. This places the different pieces into the `/usr/src/redhat` hierarchy of directories. Table 3-2 shows the purpose of the directories in this tree.



For the exam you should know how to build a binary package from a source package, including how to modify the `spec` file.

Table 3-2
Subdirectories in the `/usr/src/redhat` Tree

<i>Directory</i>	<i>Purpose</i>
<code>/usr/src/redhat/SOURCES</code>	Contains the application source code
<code>/usr/src/redhat/SPECS</code>	Contains the RPM <code>spec</code> file
<code>/usr/src/redhat/BUILD</code>	Where source code is built
<code>/usr/src/redhat/RPMS</code>	Contains the final binary RPM
<code>/usr/src/redhat/SRPMS</code>	Contains the source RPM built by the process

The `spec` file for a package controls how the package is compiled and the scripts that run when it is installed or removed. This file is named `/usr/src/redhat/SPECS/package_name.spec`. The `spec` file has eight sections, as shown in Table 3-3.

Table 3-3
spec File Sections

<i>spec File Section</i>	<i>Description</i>
Header	General information such as summary, name, version, and so on
Prep	Shell scripts that do any work needed before the compile process
Build	Commands to build the <code>spec</code> file and compile the source code
Install	Commands needed to install the software on a system
Clean	Optional commands to clean up the build environment, in case this package is rebuilt again
Optional Install and Uninstall Scripts	Scripts that may be run during install or uninstall of the package
File List	List of files in the package
Changelog	Log of any changes you make to the package

Here is a sample build section from the spec file for the `wget` application:

```
%build
#./configure --prefix=/usr --sysconfdir=/etc

%configure --sysconfdir=/etc
make
```

If you wanted to make any changes to the compile process they would be made here. This is what allows you to customize the package to suit your needs. After all of the appropriate modifications have been made, the binary package needs to be built. This is done with the `-ba` option for `rpm`.

```
rpm -ba package.spec
```

Once the process is complete, the binary package will be put in the `/usr/src/redhat/RPMS` directory.

Debian Package Management

Objective

2.2 Installation and Package Management

- **Use Debian package management.** Use the Debian package management system, from the command line (`dpkg`) and with interactive tools (`dselect`). Be able to find a package containing specific files or software; select and retrieve them from archives; install, upgrade or uninstall them; obtain status information like version, content, dependencies, integrity, installation status; and determine which packages are installed and from which package a specific file has been installed. Be able to install a non-Debian package on a Debian system. Involves using the commands and programs: `dpkg`, `dselect`, `apt`, `apt-get`, `alien`. Involves reviewing or editing the files and directories: `/var/lib/dpkg/*`.

The Debian distribution and those based on it use a different packaging system than Red Hat uses. The Debian system is often considered to be more powerful and robust. The four pieces of the system used most often are the following:

- ♦ `dpkg`
- ♦ `dselect`
- ♦ `apt-get`
- ♦ `alien`

Debian packages, or *debs*, usually contain the binary files to be installed along with other information known as *metadata*. This metadata holds information on the package, scripts that will be executed, and a list of any dependencies, conflicts, or suggestions. Some Debian packages contain the source code and can be compiled manually. The Debian package naming scheme is as follows:

```
package_version-build_architecture.deb
```

where:

- ♦ `package` is the name of the application being installed.
- ♦ `version` is the version number of the application.
- ♦ `build` is the build number of this package. Each time the package is redone this number is incremented, usually when it is tweaked in some way.
- ♦ `architecture` is the platform the package was compiled for.

A special type of package is available, known as a *task package*. These packages are actually empty packages with no software, but have a large number of dependencies. They are used to install a large “task” on the system, such as X Window workstation, GNOME desktop, or KDE. They are installed the same way as normal packages and have names such as `helix-gnome-task`.

Using dpkg

The core package tool in Debian is `dpkg`. Most of the other tools are just overlays that make `dpkg` easier to use. But, some functions can be performed only with `dpkg`, and it is sometimes faster to use `dpkg` directly than going through another interface.

Installing packages

Objective

2.2 Installation and Package Management

- **Use Debian package management.** Be able to install, upgrade or uninstall packages.

Once you have obtained a package you want to install, you use the following command:

```
dpkg --install package_file.deb
```

or:

```
dpkg -i package_file.deb
```


During the install process `dpkg` will check the package for needed dependencies and display an error similar to the following if they are not installed.

```
debian:~# dpkg --install ethereal_0.8.13-2_i386.deb
Selecting previously deselected package ethereal.
(Reading database ... 54478 files and directories currently
installed.)
Unpacking ethereal (from ethereal_0.8.13-2_i386.deb) ...
dpkg: dependency problems prevent configuration of ethereal:
 ethereal depends on libpcap0 (>= 0.4-1); however:
  Package libpcap0 is not installed.
dpkg: error processing ethereal (--install):
 dependency problems - leaving unconfigured
Errors were encountered while processing:
 Ethereal
```

The package `ethereal_0.8.13-2_i386.deb` requires the `libpcap0` package to be installed already, and it is not. Either install the required package separately or at the same time, as follows:

```
debian:~# dpkg --install ethereal_0.8.13-2_i386.deb
libpcap0_0.4a6-3_i386.deb
(Reading database ... 54499 files and directories currently
installed.)
Preparing to replace ethereal 0.8.13-2 (using
ethereal_0.8.13-2_i386.deb) ...
Unpacking replacement ethereal ...
Selecting previously deselected package libpcap0.
Unpacking libpcap0 (from libpcap0_0.4a6-3_i386.deb) ...
Setting up libpcap0 (0.4a6-3) ...

Setting up ethereal (0.8.13-2) ...
```

Force options

In some cases you may need to override an error when installing or removing a package. The `dpkg` tool provides several layers of forcing so you can set which errors are ignored and which are not. Table 3-4 shows the most common options.

Table 3-4
Common Forcing Options for `dpkg`

<i>Option</i>	<i>Purpose</i>
<code>configure-any</code>	Configure any other package that may help to install this package.
<code>hold</code>	Process another package, even if it is on hold.
<code>bad-path</code>	Force even when missing files.
<code>not-root</code>	Try to add or remove packages, even if not root.

Option	Purpose
<code>overwrite</code>	Overwrite a file from a new package, even if it belongs to another package.
<code>depends-version</code>	Normally an error is given if the correct version of a dependency is not installed. This option makes that a warning instead.
<code>depends</code>	Turn all dependency errors into warnings.
<code>confnew</code>	Always use the newly installed configuration file.
<code>confold</code>	Always use the old configuration file.
<code>conflicts</code>	Allow packages that conflict to be installed.
<code>overwrite-dir</code>	Overwrite another package's directory with a new one.
<code>remove-essential</code>	Remove essential system packages, dangerous.

For example, if you had to install a package that conflicted with an existing package, you would use the following:

```
dpkg -install new_package.deb --force-conflicts
```

Removing packages

Objective

2.2 Installation and Package Management

- Use Debian package management. Be able to install, upgrade or uninstall packages.

To remove a package, use the following commands:

```
dpkg --remove package_name
```

or:

```
dpkg -r package_name
```

These commands remove all files of the package, except for configuration files that may be needed for a later reinstall. To remove all files including configuration files, use the following commands:

```
dpkg --purge package_name
```

or:

```
dpkg -P package_name
```



Removing a package does not remove the configuration files. To remove all files, including the configuration files, use the `purge` option.

As with when installing a package, `dpkg` performs dependency checks.

```
debian:~# dpkg --remove libpcap0
dpkg: dependency problems prevent removal of libpcap0:
  ethereal depends on libpcap0 (>= 0.4-1).
dpkg: error processing libpcap0 (--remove):
  dependency problems - not removing
Errors were encountered while processing:
  libpcap0
```

Querying the package database

Objective

2.2 Installation and Package Management

- **Use Debian package management.** Be able to find a package containing specific files or software; obtain status information like version, content, dependencies, integrity, installation status; and determine which packages are installed and from which package a specific file has been installed.

As with RPM, the Debian packaging system maintains a robust database about every package installed. The `dpkg` tool allows you to access information in this database by performing queries.

To print general information about an installed package, use the following command:

```
dpkg --print-avail package_name
```

or:

```
dpkg -p package_name
```

For example, to show the information about the Ethereal package:

```
debian:~# dpkg --print-avail ethereal
Package: ethereal
Priority: optional
Section: net
Installed-Size: 2996
Maintainer: Frederic Peters <fpeters@debian.org>
Architecture: i386
Version: 0.8.13-2
Depends: libc6 (>= 2.1.94), libglib1.2 (>= 1.2.0), libgtk1.2
(>= 1.2.8-1), libpc
ap0 (>= 0.4-1), libsnmp4.1, xlibs (>= 4.0.1-1), zlib1g (>=
1:1.1.3)
Filename: dists/potato/main/binary-
i386/net/ethereal_0.8.0-1.deb
Size: 1201932
```

```
MD5sum: 78928ca734086acd72b441967bf24bc0
Description: Network traffic analyzer
  Ethereal is a network traffic analyzer, or "sniffer", for Unix
  and
  Unix-like operating systems. It uses GTK+, a graphical user
  interface
  library, and libpcap, a packet capture and filtering library.
```

As you can see some very good information is kept about each package showing the maintainer, size, version, dependencies, and description. This is useful if you are not sure what a package is used for or who to contact for information.

Listing packages

To get a concise listing of packages installed on the system use the following:

```
dpkg --list <pattern>
```

or:

```
dpkg -l <pattern>
```

For this command, *<pattern>* is an optional search pattern. Without this pattern `dpkg` will display every package available to the system. For example, to display all Apache packages you would use the following:

```
debian:/usr/doc/dpkg# dpkg --list apache*
Desired=Unknown/Install/Remove/Purge/Hold
| Status=Not/Installed/Config-files/Unpacked/Failed-
|/ Err?=(none)/Hold/Reinst-required/X=both-problems
(Status,Err: uppercase=bad)
||/ Name          Version          Description
+++-----+-----+-----+
=====
pn  apache          <none>           (no description available)
pn  apache-common  <none>           (no description available)
pn  apache-dev     <none>           (no description available)
pn  apache-doc     <none>           (no description available)
un  apache-modules <none>           (no description available)
pn  apache-perl    <none>           (no description available)
pn  apache-ssl     <none>           (no description available)
```

There are several Apache packages listed, but none are currently installed on this system. However, at one time several were installed but have been purged and are now not installed. The three columns on the left signify the following information: `p` means the file has been purged, `u` means the file is unpacked, and `n` means the file is not installed. The left columns are documented by the lines above the file list. The letters correspond to the first letter of the information. For example, `i` would mean installed, and `h` would mean half-installed.

Selection status

Selection status is used with the `dselect` tool. The possible status modes are as follows:

- ♦ **Unknown** — The current status is not known.
- ♦ **Install** — The package is marked for installation.
- ♦ **Remove** — The package has been selected for removal.
- ♦ **Purge** — The package has been selected to be purged.
- ♦ **Hold** — The package is being held and may not be upgraded.

Current status

The second column shows what state the package is currently in. The possible values are as follows:

- ♦ **Not installed** — The package is currently not installed.
- ♦ **Installed** — The package is currently installed on the system.
- ♦ **Config-files** — The package is not installed, but the config files are present. This is common after a `-remove`.
- ♦ **Unpacked** — The package was unpacked and is ready for installation.
- ♦ **Failed-config** — During installation the configuration script failed.
- ♦ **Half-installed** — During installation a problem occurred, and the installation did not complete.

Error conditions

If there is a problem with a package, you will get an error code in the third column. The values are as follows:

- ♦ **None** — There is no problem with the package.
- ♦ **Hold** — The package has been placed on hold, and it can not be installed or removed.
- ♦ **Reinstallation Required** — The package needs to be reinstalled.

Displaying the status of a package

To individually display the status of a package with full details use the following commands:

```
dpkg --status package_name
```

or:

```
dpkg -s package_name
```

For example:

```
debian:~# dpkg -s ethereal
Package: ethereal
Status: install ok installed
Priority: optional
Section: net
Installed-Size: 2996
Maintainer: Frederic Peters <fpeters@debian.org>
Version: 0.8.13-2
Depends: libc6 (>= 2.1.94), libglib1.2 (>= 1.2.0), libgtk1.2
(>= 1.2.8-1), libpc
ap0 (>= 0.4-1), libsnmp4.1, xlibs (>= 4.0.1-1), zlib1g (>=
1:1.1.3)
Description: Network traffic analyzer
Ethereal is a network traffic analyzer, or "sniffer", for Unix
and
Unix-like operating systems. It uses GTK+, a graphical user
interface
library, and libpcap, a packet capture and filtering library.
```

Listing package files

To list all the files installed by a package use the following commands:

```
dpkg --listfiles package_name
```

or:

```
dpkg -L package_name
```

For example:

```
debian:~# dpkg --listfiles wget
/.
/usr
/usr/bin
/usr/bin/wget
...
/usr/share/doc/wget/copyright
/usr/share/doc/wget/changelog.gz
/usr/share/doc/wget/NEWS.gz
/usr/share/doc/wget/changelog.Debian.gz
/usr/share/doc-base
/usr/share/doc-base/wget
/etc
/etc/wgetrc
```

Displaying file ownership

To find out which package installed a certain file, use the following commands:

```
dpkg --search package_name
```

or:

```
dpkg -S package_name
```

For example:

```
debian:/etc# dpkg --search /etc/issue.net
base-files: /etc/issue.net
```

This indicates that the `/etc/issue.net` file was installed by the `base-files` package.

Checking available packages

Most of the information displayed and used by the `dpkg` command is stored in the `/var/lib/dpkg` directory. Two files of interest in this directory are the following:

- ♦ `available`
- ♦ `status`

The `/var/lib/dpkg/available` file lists the available packages to choose from. The following is an example of this file:

```
debian:/var/lib/dpkg# more available
Package: telnet
Priority: standard
Section: net
Installed-Size: 176
Maintainer: Herbert Xu <herbert@debian.org>
Architecture: i386
Source: netkit-telnet
Version: 0.16-4
Replaces: netstd
Depends: libc6 (>= 2.1.2), libncurses5
Filename: dists/potato/main/binary-i386/net/telnet_0.16-4.deb
Size: 58826
MD5sum: 9f3a865499f80b3b4975a67f0c65f2cd
Description: The telnet client.
The telnet command is used for interactive communication with another host
using the TELNET protocol.
```

The `/var/lib/dpkg/status` file shows the status of installed packages. The following is an example of this file:

```
Package: telnet
Status: install ok installed
Priority: standard
Section: net
Installed-Size: 176
Maintainer: Herbert Xu <herbert@debian.org>
Source: netkit-telnet
Version: 0.16-4
Replaces: netstd
Depends: libc6 (>= 2.1.2), libncurses5
Description: The telnet client.
The telnet command is used for interactive communication with another host
using the TELNET protocol.
```

Normally these files are used only by `dpkg`, but they can be viewed or searched for information on available and installed packages.



Even though the exam objectives require you to know how to edit these files, you will rarely do it. However, they can contain useful information about the packages installed and available on your system.

Using dselect

Objective

2.2 Installation and Package Management

- **Use Debian package management.** Use the Debian package management system, from the command line (`dpkg`) and with interactive tools (`dselect`).

The `dselect` tool is an overlay to `dpkg` that provides a character-based graphical interface to enhance ease of use. When you run `dselect`, it displays the screen shown in Figure 3-1.

```
Debian GNU/Linux 'dselect' package handling frontend.
* 0. [A]ccess      Choose the access method to use.
1. [U]pdate        Update list of available packages, if possible.
2. [S]elect        Request which packages you want on your system.
3. [I]ninstall     Install and upgrade wanted packages.
4. [C]onfig        Configure any packages that are unconfigured.
5. [R]emove        Remove unwanted software.
6. [Q]uit          Quit dselect.

Move around with ^P and ^N, cursor keys, initial letters, or digits;
Press <enter> to confirm selection.  ^L redraws screen.

Version 1.6.14 (i386). Copyright (C) 1994-1996 Ian Jackson. This is
free software; see the GNU General Public Licence version 2 or later for
copying conditions. There is NO warranty. See dselect --licence for details.
```

Figure 3-1: Main `dselect` menu

The following options are given on the main menu and are discussed in the following sections:

- ♦ Access
- ♦ Update
- ♦ Select
- ♦ Install
- ♦ Config
- ♦ Remove
- ♦ Quit

Access

The `dselect` tool allows you to access packages from a variety of sources. These sources are configured from the Access menu option. Figure 3-2 shows a sample Access screen.

```
dselect - list of access methods
Abbrev.      Description
nfs          Install from an NFS server (not yet mounted).
floppy       Install from a pile of floppy disks.
* apt        APT Acquisition [file,http,ftp]

Access Method 'apt'.
apt - APT Acquisition [file,http,ftp]

The APT installation method encompasses most other installation methods
under the umbrella of the new Package Acquisition code. This method allows
installation from locations in the filesystem, ftp and http URLs, supports
full installation ordering and dependency checking as well as multiple
sources. See the man pages apt-get(8) and sources.list(5)

HTTP proxies can be used by setting http_proxy="http://proxy:port/" before
running DSelect. FTP proxies require special configuration detailed in the
explanation of apt -- 83%, press d for more.
```

Figure 3-2: The `dselect` Access screen

Table 3-5 shows the available source types. In most cases only the `apt` source is used as it can access many different source types and is very configurable. See the section on `apt-get` for configuration information on the `sources.list` file that configures the `apt` access method.

Table 3-5
Dselect Access Methods

<i>Method</i>	<i>Description</i>
cdrom	Install from a local CD-ROM drive. Does not have to be mounted.
nfs	Install from an unmounted NFS server.
harddisk	Install from a disk that is not already mounted.
mounted	Install from any file system that is already mounted.
floppy	Install from multiple floppy disks.
ftp	Install from an FTP site.
apt	Allows access from several different sources. See the section on <code>apt-get</code> for more information.

Update

Any time that an access method is changed or packages on them have changed, you need to run the Update function so that the package database is current. This process goes through each source and compiles the complete list of packages, as well as needed dependencies.

Select

After the package database has been updated, you are ready to select the packages for install. Choosing Select from the main `dselect` screen shows you the introduction screen in Figure 3-3.

```

Help: Introduction to package list
Welcome to the main package listing. Please read the help that is available!

You will be presented with a list of packages which are installed or available
for installation. You can navigate around the list using the cursor keys,
mark packages for installation (using '+') or deinstallation (using '-').

Packages can be marked either singly or in groups; initially you will see that
the line 'All packages' is selected. '+', '-' and so on will affect all the
packages described by the highlighted line. Use 'o' to change the order of the
list (this also changes which kinds of group selections are possible).

(Mainly for new installations:) Standard packages will be requested by default.
Use capital 'D' or 'R' key to override this - see the keybindings help screen.

Some of your choices will cause conflicts or dependency problems; you will be
given a sub-list of the relevant packages, so that you can solve the problems.

When you are satisfied with your choices you should press Return to confirm
your changes and leave the package listing. A final check on conflicts and
dependencies will be done - here too you may see a sublist.

Press Space to leave help and enter the list; press '?' at any time for help.

? = help menu   Space = exit help   . = next help   or a help page key

```

Figure 3-3: Introduction to package list

The introduction screen provides an overview of the selection process and explains some of the keys to use. Press the space key to move to the package selection screen, shown in Figure 3-4.

```
dselect - main package listing (avail., priority)  mark:+/=/- verbose:v help:?
E10M Pri Section Package           Inst.ver  Avail.ver  Description
-----
- All packages -
  Newly available packages
  New Standard packages
  New Standard packages in section admin
n* Std admin  ncurses-term <none>    5.0-6      Additional terminal type
  New Standard packages in section base
n* Std base   dpkg-ftp       <none>    1.6.7      Ftp method for dselect.
  New Standard packages in section devel
n* Std devel  bin86         <none>    0.14.9-3  16-bit assembler and load
n* Std devel  bison         <none>    1.28-5    A parser generator that i
n* Std devel  dpkg-perl     <none>    0.1-3.0   Perl interface modules fo
All packages
The line you have highlighted represents many packages; if you ask to
install, remove, hold, &c it you will affect all the packages which match
the criterion shown.

If you move the highlight to a line for a particular package you will see
information about that package displayed here. You can use 'o' and 'O' to
change the sort order and give yourself the opportunity to mark packages in
different kinds of groups.

description
```

Figure 3-4: Package listing

Use the arrow keys to move up and down the list of available packages. Table 3-6 shows the common commands to use in the selection environment.

Table 3-6
Selection Keys

Key	Function
+	Install selected package.
-	Remove selected package.
=	Place package on hold.
:	Remove hold status on package.
/ <i><pattern></i>	Search using <i><pattern></i> .
\	Repeat last search.
O	Cycle through sort options.
V	Change status display.
X	Exit abandoning changes.
Q	Exit saving changes.

When a package is selected for installation or removal, `dselect` also checks for dependencies that are not already installed or that may be broken by a removal. If a dependency problem occurs you will be presented a screen similar to Figure 3-5. This screen will also be displayed if there are any recommended or suggested packages for the package being installed.

```

Help: Introduction to conflict/dependency resolution sub-list
Dependency/conflict resolution - introduction.

One or more of your choices have raised a conflict or dependency problem -
some packages should only be installed in conjunction with certain others, and
some combinations of packages may not be installed together.

You will see a sub-list containing the packages involved. The bottom half of
the display shows relevant conflicts and dependencies; use 'i' to cycle between
that, the package descriptions and the internal control information.

A set of 'suggested' packages has been calculated, and the initial markings in
this sub-list have been set to match those, so you can just hit Return to
accept the suggestions if you wish. You may abort the change(s) which caused
the problem(s), and go back to the main list, by pressing capital 'X'.

You can also move around the list and change the markings so that they are more
like what you want, and you can 'reject' my suggestions by using the capital
'D' or 'R' keys (see the keybindings help screen). You can use capital 'Q' to
force me to accept the situation currently displayed, in case you want to
override a recommendation or think that the program is mistaken.

Press Space to leave help and enter the sub-list; remember: press '?' for help.

? = help menu      Space = exit help      . = next help      or a help page key

```

Figure 3-5: Dependency/conflict resolution introduction

After reading the dependency resolution introduction screen and hitting the spacebar, the resolution screen is displayed, as shown in Figure 3-6. By highlighting a package at the top of the screen the dependencies are displayed at the bottom and can be selected for installation. `dselect` makes suggestions that should resolve any conflicts or dependencies by default. Pressing the spacebar will exit this screen after any changes have been made.

To exit the selection screen hit Q to exit and keep your changes or X to exit and undo any changes.

Install

Once you have chosen the packages for installation, the next step is to select the Install option from the main menu. This option gets the packages and installs them.



In some cases, not all packages will be installed in one step, so it is always a good idea to reselect the Install option and make sure that `dselect` is finished working.

```
dselect - recursive package listing                               Mark:+/=- verbose:0 help:?
E10M Pri Section Package Description
** Std tex tetex-bin teTeX binary files
- * Opt admin debconf Debian configuration management system
- * Opt devel cvs Concurrent Versions System
- * Opt devel cvs-buildpac Debian package scripts for CVS source trees.
** - Opt admin debconf-tiny
** Opt devel task-debian- Debian package development

tetex-bin not installed ; install (was: install). Standard
tetex-bin recommends dialog
dialog does not appear to be available
tetex-bin suggests tetex-extra

interrelationships affecting tetex-bin
```

Figure 3-6: Dependency/conflict resolution

Config

Some packages require user intervention to finish configuration. This usually involves answering one or more questions to customize the package for varying situations. To run the configuration script for any new packages that have them, select the Config option from the main menu. When finished, the configuration scripts will return you to `dselect`.

Remove

If any packages were selected for removal, this process is started by selecting Remove from the main menu. When the removal process is finished you are returned to `dselect`.

Quit

To exit `dselect` choose Quit from the main menu.

Using apt-get

Objective

2.2 Installation and Package Management

- Use Debian package management. Involves using the commands and programs: `dpkg`, `dselect`, `apt`, `apt-get`, `alien`.

`apt-get` is a command-line tool with a lot of the functionality of `dselect`, without the overhead of the sometimes unneeded interface. The `apt-get` tool will automatically go get a package like `dselect`, as well as any needed dependencies.

Editing the `sources.list` file

Before `apt-get` can get packages to install, it has to know where to get them from. The `/etc/apt/sources.list` file has the possible sources for packages. This is the same file used by the `apt source` in `dselect`. The file is laid out with a simple list of resources using the following format:

```
deb uri distribution component
```

for binary packages and:

```
deb-src uri distribution component
```

for source packages.

URI stands for Uniform Resource Identifier, which is a superset of the familiar URL format that most people are familiar with. It uses the following syntax:

```
protocol://host/path
```

The `//host` section of the URI is used only with HTTP and FTP methods. The four types of protocols supported in the `sources.list` file are as follows:

- ♦ `cd-rom`—Local CD-ROM drive
- ♦ `file`—Local directory
- ♦ `http`—Web site
- ♦ `ftp`—FTP site

Debian normally has two distributions current at one time. These are as follows:

- ♦ `stable`—This is the latest official release. The only updates made to this distribution are security updates.
- ♦ `unstable`—This version is in a constant state of change. It will eventually be the next version of `stable`. It is not recommended for production systems.

Not all sites carry all of the components, so they must be individually specified. The standard components are as follows:

- ♦ `main`—The main set of packages.
- ♦ `contrib`—Secondary packages.
- ♦ `non-free`—Debian is a totally free distribution, but some useful packages are not considered free. These packages live in this component and are not considered part of the Debian distribution.

We recommend that you put the fastest sources at the top of the `sources.list` file. Comments can be added to the list using the `#` sign, which can be helpful when adding other sources. It is common to add sources for a specific application or package. An example of a `source.list` file is the following:

```
# See sources.list(5) for more information, especially
# Remember that you can only use http, ftp or file URIs
# CDRoms are managed through the apt-cdrom tool.
deb http://http.us.debian.org/debian unstable main contrib non-free
deb http://non-us.debian.org/debian-non-US unstable/non-US main contrib non-free
deb http://security.debian.org stable/updates main contrib non-free

#HelixCode
deb http://spidermonkey.helixcode.com/distributions/debian unstable main

# Uncomment if you want the apt-get source function to work
#deb-src http://http.us.debian.org/debian stable main contrib non-free
#deb-src http://non-us.debian.org/debian-non-US stable non-US
```

Updating the available packages

The Debian package database maintains a list of available packages. It is useful to update this list periodically and any time you make a change to the `sources.list` file. To update the list run the following command:

```
apt-get update
```

`apt-get` will go through the list of sources and update the package database.

Installing a package

When `apt-get` is instructed to install a package, it first checks to see if the package has already been downloaded. If not, `apt-get` goes to the first source in the list with the newest version. If any dependencies are needed they will be retrieved as well. To install a package, use the following command:

```
apt-get install package_name
```

For example:

```
norbert:~# apt-get install ethereal
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
  libpcap0
The following NEW packages will be installed:
  ethereal libpcap0
0 packages upgraded, 2 newly installed, 0 to remove and 202 not
upgraded.
```

```
Need to get 1240kB of archives. After unpacking 3153kB will be
used.
Do you want to continue? [Y/n]
Get:1 http://http.us.debian.org unstable/main ethereal 0.8.13-2
[1202kB]
Get:2 http://http.us.debian.org unstable/main libpcap0 0.4a6-3
[38.6kB]
Fetched 1240kB in 16s (74.0kB/s)
Selecting previously deselected package libpcap0.
(Reading database ... 53531 files and directories currently
installed.)
Unpacking libpcap0 (from ../libpcap0_0.4a6-3_i386.deb) ...
Selecting previously deselected package ethereal.
Unpacking ethereal (from ../ethereal_0.8.13-2_i386.deb) ...
Setting up libpcap0 (0.4a6-3) ...

Setting up ethereal (0.8.13-2) ...
```

When the Ethereal package is installed, it also requires `libpcap0`. The `apt-get` tool prompts the user to get the dependencies. If you answer *Yes* at the prompt, the extra packages are automatically retrieved and installed in one step.

Upgrading packages

One of the most powerful features of the `apt` system is the ability to upgrade all installed packages to their latest version in one step. To do this use the following command:

```
apt-get upgrade
```

Be sure to issue an `apt-get update` before doing an upgrade, so the package list is up-to-date. Depending on the Internet connection of the system, the upgrade may take a while.

Removing packages

Packages can be removed with `apt-get` as well as `dpkg`. The command for this is as follows:

```
apt-get remove package_name
```

Upgrading the distribution

While Debian's install may not be as smooth as other distributions, it makes up for it in that you may only run it once, even when moving to new versions of the distribution. The entire distribution can be upgraded to a new version, when one is released, via the `apt-get` tool.

```
apt-get dist-upgrade
```


The difference between this and the normal upgrade procedure is that `apt` uses smarter dependency checking and may upgrade more important packages at the expense of other packages.

Clearing the package archives

When `apt-get` installs a package, it keeps a copy of the `deb` file in an archive in the `/var/cache/apt/archives` and `/var/cache/apt/archives/partial` directories. Over time these can take up a large amount of disk space. To clear all of the files in these directories use the following:

```
apt-get clean
```

At times it may be beneficial to keep some archives. To clear only packages that are no longer current and cannot be downloaded again, use the following:

```
apt-get autoclean
```

apt-get options

Along with the action commands above, `apt-get` provides several other options to change its behavior. Table 3-7 shows these options.

Table 3-7
apt-get Options

<i>Option</i>	<i>Purpose</i>
-h	Display the <code>apt-get</code> help.
-q	Display output for logging.
-qq	No output except for errors.
-d	Only download the packages to the archives directory; do not install them.
-s	Simulate the action only. This will display the information as if you have actually executed the process. This is useful for testing purposes before you accidentally break anything.
-y	Answer Yes to all queries.
-f	Continue even if integrity check fails. This is sometimes used to fix dependency issues.
-m	Continue even if the packages can not be located.
-u	Show a list of upgraded packages.
-b	Build a source package after retrieving it.
-c= <i>filename</i>	Read the specified configuration file.
-o= <i>option</i>	Set a special configuration option.

Using alien

Objective

2.2 Installation and Package Management

- **Use Debian package management.** Be able to install a non-Debian package on a Debian system. Involves using the commands and programs: `dpkg`, `dselect`, `apt`, `apt-get`, `alien`.

Most popular applications are already packaged in `deb` files, either by the author or another maintainer, although some are not. The `alien` tool converts packages between several different formats. It can be used on any distribution. The following formats are supported:

- ♦ Debian `.deb`
- ♦ Red Hat `.rpm`
- ♦ Slackware `.tgz`
- ♦ Stampede `.slp`

The syntax for `alien` is as follows:

```
alien [options] package
```

The options for `alien` are shown in Table 3-8.

Table 3-8
alien Options

<i>Option</i>	<i>Alternate</i>	<i>Purpose</i>
<code>-d</code>	<code>--to-deb</code>	The default. Tells <code>alien</code> to create a <code>.deb</code> package
<code>--patch=<filename></code>		Only used with <code>-d</code> . Specifies the patch file to use.
<code>--nopatch</code>		Only used with <code>-d</code> . Specifies that no patch file should be used.
<code>-r</code>	<code>--to-rpm</code>	Creates an RPM package.
<code>-t</code>	<code>--to-tgz</code>	Creates a Slackware <code>.tgz</code> package.
<code>--to-slp</code>		Creates a Stampede package.
<code>-i</code>	<code>--install</code>	Installs the package after creation.
<code>-g</code>	<code>--generate</code>	Unpacks the package, but does not generate a new one.

Continued

Table 3-8 (continued)

Option	Alternate	Purpose
-s	--single	Does the same as -g, but does not create the .orig directory.
-c	--scripts	Includes scripts in the new package.
-k	--keep-version	Does not change the version of the generated package.
--description=		Sets the package's description.
-h	--help	Shows the help options.
-v	--version	Shows the version of alien.

For example, to convert the `wget` package to a `.deb` file, you would use the following command:

```

debian:~# alien wget.rpm
-- Examining wget.rpm
-- Unpacking wget.rpm
1010 blocks
----
-- Automatic package debianization
-- Building the package wget_1.5.3-164_i386.deb
dh_testdir
# Nothing to do.
dh_testdir
dh_testroot
dh_clean -k
dh_installdirs
cp -a `ls |grep -v debian` debian/tmp
dh_installdocs
dh_installexamples
dh_installmenu
dh_installcron
dh_installchangelogs
dh_compress
dh_suidregister
dh_installdeb
dh_shlibdeps
dh_gencontrol
dh_makeshlibs
dh_md5sums
dh_builddeb
dpkg-deb: building package `wget' in
`../wget_1.5.3-164_i386.deb'.

Generation of wget_1.5.3-164_i386.deb complete.
-- Successfully finished

```

Now the new `.deb` file is available for install.



`alien` is used to convert packages from one type to another.

Key Point Summary

Packaging systems have come a long way in Linux and are a great timesaver for users and administrators. The different packaging systems may seem very different at first but share many similar processes and ideas. The key for the exam is remembering which system uses which tools and how those tools relate to one another.

- ♦ Tarball files contain only the source code or files needed for an application.
- ♦ The `./configure` script generates a `Makefile` tuned for the system it was run on.
- ♦ `Makefiles` may need to be edited to add additional `Include` directories.
- ♦ Source code is compiled using the `make` and `make install` commands.
- ♦ RPM is used on many different Linux distributions and was created by Red Hat.
- ♦ The main tool for use with RPM is `rpm`.
- ♦ The common `rpm` commands are as follows:
 - To install a package use `rpm -I` or `rpm --install`
 - To remove a package use `rpm -e` or `rpm --uninstall`
 - To upgrade a package use `rpm -U` or `rpm --upgrade`
 - To query the database use `rpm -q` or `rpm --query`
 - To rebuild the RPM database use `rpm --rebuilddb`
- ♦ The Debian distribution and its variants use the `dpkg` tool for low-level processing.
- ♦ The common `dpkg` commands are as follows:
 - To install a package use `dpkg -I` or `dpkg --install`
 - To remove a package use `dpkg -r` or `dpkg --remove`
 - To purge a package use `dpkg -P` or `dpkg --purge`
- ♦ `dselect` provides an easy to use interface for `dpkg`.

- ♦ `apt-get` automatically retrieves requested packages and their dependencies.
- ♦ `apt-get`'s sources are configured via the `/etc/apt/sources.list` file.
- ♦ `alien` is used to convert packages to different types.



STUDY GUIDE

The following questions and exercises will allow you to review the information covered in this chapter. Take your time when completing this section of the chapter, carefully reviewing any questions that you may have. Answering the question correctly is not as important as understanding the answer, so review any material that you might still be unsure of. Being comfortable with the questions and answers presented here will help you be more prepared for the certification exam questions.

Assessment Questions

1. Which script generates a `Makefile` for your specific system?
 - A. `./gen`
 - B. `./genmake`
 - C. `./configure`
 - D. `./config`
2. Which command installs compiled software?
 - A. `make`
 - B. `./install`
 - C. `make setup`
 - D. `make install`
3. Which packaging system does Red Hat use?
 - A. `rpm`
 - B. `deb`
 - C. `tgz`
 - D. `rhp`
4. Which `rpm` option is used when you receive strange errors when installing packages, suggesting `rpm` database corruption?
 - A. `rpm --fixdb`
 - B. `rpm --rebuilddb`
 - C. `rpm --updatedb`
 - D. `rpm --regendb`

5. Which methods does RPM support to check package integrity? (Select all that apply.)
- A. MD5
 - B. 3DES
 - C. PGP
 - D. GnuPG
6. The command _____ would be entered to install the package named `processor-4.2.i386.rpm`.
7. Which tool is used to update the `ld.so.cache` file?
- A. `ldcache`
 - B. `ldupdate`
 - C. `ldconf`
 - D. `ldconfig`
8. Which linking method creates smaller executable files?
- A. Dynamic
 - B. Unlinked
 - C. Variable
 - D. Static
9. Which command(s) is/are used to remove an RPM package?
- A. `rpm --uninstall <packagename>`
 - B. `rpm --remove <packagename>`
 - C. `rpm -e <packagename>`
 - D. `rpm -u <packagename>`
10. Which file specifies how an RPM source package is compiled?
- A. Makefile
 - B. spec file
 - C. config file
 - D. .conf file

11. Which tool is used to convert packages from one system to another?
- A. alien
 - B. dpkg
 - C. apt
 - D. Pconvert
12. Which command is used to install a Debian package?
- A. apt --install <packagename>
 - B. dpkg --install <packagename>
 - C. apt-get -I <packagename>
 - D. rpm -i <packagename>
13. Which command removes a Debian package, including its configuration files?
- A. dpkg --remove <packagename>
 - B. dpkg -e <packagename>
 - C. apt-get purge <packagename>
 - D. dpkg -P <packagename>
14. Which tool provides an easy-to-use interface to access Debian package management?
- A. dselect
 - B. apt-get
 - C. dpkg
 - D. gnorpm
15. To change the sources for apt-get, the _____ file is edited.
- A. sources.list
 - B. apt.sources
 - C. sources.apt
 - D. dpkg.sources
16. Which apt-get parameter updates the available package database?
- A. upgrade
 - B. refresh
 - C. reload
 - D. update

17. The `apt-get` tool supports which of the following sources? (Select all that apply.)
- A. FTP
 - B. HTTP
 - C. NFS
 - D. CD-ROM
18. Which command clears old packages from the Debian archives?
- A. `dpkg -clean`
 - B. `apt-get autoclean`
 - C. `dpkg -autoclean`
 - D. `dselect`
19. `alien` supports which of the following package formats? (Select all that apply.)
- A. RPM
 - B. `.deb`
 - C. BSD
 - D. `.tgz`
20. Which command converts an RPM package to Debian format?
- A. `alien -r package.rpm`
 - B. `alien -t package.rpm`
 - C. `alien -d package.deb`
 - D. `alien -d package.rpm`

Scenarios

1. You have found a new software package you want to install on your Debian system. The problem is that it is distributed only in source code and RPM format. What is the best option to install this package on your system?
2. Red Hat has released a new version of the RPM system. You install the new version on your system, but now when you try to install a package, you receive an error stating that dependency packages are not installed. You know that you installed these packages before the upgrade, but they no longer show up. What steps could you go through to fix this problem?

Lab Exercises

The following labs allow you to exercise some of the commands learned in this chapter. These are only a few examples. The key to learning all of the commands and tools is experience. Work through installing several packages and create dependency problems to work through.

Red Hat labs

These labs should be done on a Red Hat or other RPM-based system.

Lab 3-1 Listing installed packages

1. List the packages installed on your system by using the `rpm -qa` command.

Lab 3-2 Checking file ownership

1. Check to see which package owns the following system files with the `rpm -qf <filename>` command.

- `/bin/lis`
- `/bin/login`
- `/etc/inetd.conf`
- `/etc/hosts`

Notice that the `/etc/hosts` file is not owned by a package. It is generated during the system installation. You may occasionally run into other files that are not owned by a package.

Lab 3-3 Using GnuPG

1. Confirm that GnuPG is installed by using the following command:

```
[root@redhat /root]# rpm -q gnupg
gnupg-1.0.1-1
```

2. Retrieve several RPM files and the Red Hat GnuPG public key from either Red Hat's FTP site or the Red Hat CD-ROM. An example of FTP is shown below. RPMs are stored in the Red Hat CD-ROM in the `/RedHat/RPMS` directory. In this example, you will use the `wget` and `lynx` RPM files.

```
[root@redhat /root]# ftp ftp.redhat.com
Connected to ftp.redhat.com.
220 "Red Hat FTP server ready. All transfers are logged,
please have a nice day."
Name (ftp.redhat.com:root): anonymous
331 Guest login ok, send your complete e-mail address as
password.
```

```

Password: <Email Address>
230-Please read the file README
230- it was last modified on Tue Jan 25 08:51:37 2000 - 369
days ago
230-Please read the file README.roughcuts
230- it was last modified on Fri Jan 21 17:32:20 2000 - 373
days ago
230 Guest login ok, access restrictions apply.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd redhat/current/i386/en
250-Please read the file README
250- it was last modified on Fri Aug 25 12:31:19 2000 - 156
days ago
250 CWD command successful.
ftp> get RPM-GPG-KEY
local: RPM-GPG-KEY remote: RPM-GPG-KEY
200 PORT command successful.
150 Opening BINARY mode data connection for RPM-GPG-KEY (1908
bytes).
226 Transfer complete.
1908 bytes received in 0.0589 secs (32 Kbytes/sec)
ftp> cd RedHat/RPMS
250 CWD command successful.
ftp> mget wget* lynx*
mget wget-1.5.3-10.i386.rpm? y
200 PORT command successful.
150 Opening BINARY mode data connection for wget-
1.5.3-10.i386.rpm (156532 bytes).
226 Transfer complete.
156532 bytes received in 0.948 secs (1.6e+02 Kbytes/sec)
mget lynx-2.8.4-3.i386.rpm? y
200 PORT command successful.
150 Opening BINARY mode data connection for lynx-
2.8.4-3.i386.rpm (935337 bytes).
226 Transfer complete.
935337 bytes received in 5.29 secs (1.7e+02 Kbytes/sec)
ftp> quit
221-You have transferred 1250309 bytes in 4 files.
221-Total traffic for this session was 1253244 bytes in 7
transfers.
221-Thank you for using the FTP service on
pub.nyc.redhat.com.
221 Goodbye.

```

3. Import the Red Hat public key file into GnuPG by using the following command (If this is the first time running gpg, you will need to execute it twice):

```

[root@redhat /root]# gpg --import RPM-GPG-KEY
gpg: /root/.gnupg: directory created
gpg: /root/.gnupg/options: new options file created

```

```

gpg: you have to start GnuPG again, so it can read the new
options file
[root@redhat /root]# gpg --import RPM-GPG-KEY
gpg: /root/.gnupg/secring.gpg: keyring created
gpg: /root/.gnupg/pubring.gpg: keyring created
gpg: key DB42A60E: public key imported
gpg: /root/.gnupg/trustdb.gpg: trustdb created
gpg: Total number processed: 1
gpg:             imported: 1

```

4. Check the integrity of the newly downloaded RPM files by using the `-K` option. For example:

```

[root@redhat /root]# rpm -K *.rpm
lynx-2.8.4-3.i386.rpm: md5 gpg OK
wget-1.5.3-10.i386.rpm: md5 gpg OK

```

Debian labs

These labs should be done on a Debian-based system.

The following is an example of default `/etc/apt/sources.list` on a new Debian install. You will use it for the following labs.

```

# See sources.list(5) for more information, especially
# Remember that you can only use http, ftp or file URIs
# CDROMs are managed through the apt-cdrom tool.
#deb http://http.us.debian.org/debian stable main contrib non-free
#deb http://non-us.debian.org/debian-non-US stable/non-US main contrib non-free
#deb http://security.debian.org stable/updates main contrib non-free

# Uncomment if you want the apt-get source function to work
#deb-src http://http.us.debian.org/debian stable main contrib non-free
#deb-src http://non-us.debian.org/debian-non-US stable non-US

deb cdrom:[Debian GNU/Linux 2.2 r0 _Potato_ - Official i386 Binary-1 (20000814)]
 / unstable contrib main non-US/contrib non-US/main

```

Lab 3-4 Working with Debian packages

1. Remove the comment line from the first three source lines in the `sources.list` file. For example:

```

deb http://http.us.debian.org/debian stable main contrib non-free
deb http://non-us.debian.org/debian-non-US stable/non-US main contrib non-free
deb http://security.debian.org stable/updates main contrib non-free

```

2. Perform an `apt-get update` to refresh the available packages from the Debian package site.

```
debian:/etc/apt# apt-get update
Hit http://http.us.debian.org stable/main Packages
Hit http://http.us.debian.org stable/main Release
Hit http://http.us.debian.org stable/contrib Packages
Hit http://http.us.debian.org stable/contrib Release
Hit http://http.us.debian.org stable/non-free Packages
Hit http://http.us.debian.org stable/non-free Release
Hit http://security.debian.org stable/updates/main Packages
Hit http://security.debian.org stable/updates/main Release
Hit http://security.debian.org stable/updates/contrib Packages
Hit http://security.debian.org stable/updates/contrib Release
Hit http://security.debian.org stable/updates/non-free Packages
Hit http://security.debian.org stable/updates/non-free Release
Hit http://non-us.debian.org stable/non-US/main Packages
Hit http://non-us.debian.org stable/non-US/main Release
Hit http://non-us.debian.org stable/non-US/contrib Packages
Hit http://non-us.debian.org stable/non-US/contrib Release
Hit http://non-us.debian.org stable/non-US/non-free Packages
Hit http://non-us.debian.org stable/non-US/non-free Release
Reading Package Lists... Done
Building Dependency Tree... Done
```

3. Next, install a package from the remote site. In this example you will use our favorite `wget` application.

```
debian:/etc/apt# apt-get install wget
Reading Package Lists... Done
Building Dependency Tree... Done
The following NEW packages will be installed:
  wget
0 packages upgraded, 1 newly installed, 0 to remove and 45 not upgraded.
Need to get 228kB of archives. After unpacking 571kB will be used.
Get:1 http://http.us.debian.org stable/main wget 1.5.3-3 [228kB]
Fetched 228kB in 6s (35.2kB/s)
Selecting previously deselected package wget.
(Reading database ... 16992 files and directories currently installed.)
Unpacking wget (from ../archives/wget_1.5.3-3_i386.deb) ...
Setting up wget (1.5.3-3) ...
```

4. What files did you just install? Find out with the next command.

```
debian:/etc/apt# dpkg -L wget
/.
/usr
/usr/bin
/usr/bin/wget
/usr/share
```

```
/usr/share/info
/usr/share/info/wget.info-2.gz
/usr/share/info/wget.info-3.gz
/usr/share/info/wget.info.gz
/usr/share/info/wget.info-1.gz
/usr/share/locale
/usr/share/locale/cs
/usr/share/locale/cs/LC_MESSAGES
/usr/share/locale/cs/LC_MESSAGES/wget.mo
/usr/share/locale/de
/usr/share/locale/de/LC_MESSAGES
/usr/share/locale/de/LC_MESSAGES/wget.mo
/usr/share/locale/hr
/usr/share/locale/hr/LC_MESSAGES
/usr/share/locale/hr/LC_MESSAGES/wget.mo
/usr/share/locale/no
/usr/share/locale/no/LC_MESSAGES
/usr/share/locale/no/LC_MESSAGES/wget.mo
/usr/share/locale/it
/usr/share/locale/it/LC_MESSAGES
/usr/share/locale/it/LC_MESSAGES/wget.mo
/usr/share/locale/pt_BR
/usr/share/locale/pt_BR/LC_MESSAGES
/usr/share/locale/pt_BR/LC_MESSAGES/wget.mo
/usr/share/locale/ja
/usr/share/locale/ja/LC_MESSAGES
/usr/share/locale/ja/LC_MESSAGES/wget.mo
/usr/share/man
/usr/share/man/man1
/usr/share/man/man1/wget.1.gz
/usr/share/doc
/usr/share/doc/wget
/usr/share/doc/wget/html
/usr/share/doc/wget/html/wget_10.html
/usr/share/doc/wget/html/wget_11.html
/usr/share/doc/wget/html/wget_1.html
/usr/share/doc/wget/html/wget_2.html
/usr/share/doc/wget/html/wget_3.html
/usr/share/doc/wget/html/wget_4.html
/usr/share/doc/wget/html/wget_5.html
/usr/share/doc/wget/html/wget_6.html
/usr/share/doc/wget/html/wget_7.html
/usr/share/doc/wget/html/wget_8.html
/usr/share/doc/wget/html/wget_9.html
/usr/share/doc/wget/html/wget_foot.html
/usr/share/doc/wget/html/wget_toc.html
/usr/share/doc/wget/README
/usr/share/doc/wget/MAILING-LIST
/usr/share/doc/wget/TODO
/usr/share/doc/wget/AUTHORS
/usr/share/doc/wget/copyright
```

```

/usr/share/doc/wget/changelog.gz
/usr/share/doc/wget/NEWS.gz
/usr/share/doc/wget/changelog.Debian.gz
/usr/share/doc-base
/usr/share/doc-base/wget
/etc
/etc/wgetrc

```

5. What sort of information can you gather from this package? This is shown with the next command:

```

debian:/etc/apt# dpkg -s wget
Package: wget
Status: install ok installed
Priority: optional
Section: web
Installed-Size: 558
Maintainer: Nicol as Lichtmaier <nick@debian.org>
Version: 1.5.3-3
Depends: libc6 (>= 2.1)
Conffiles:
 /etc/wgetrc 9d49747a4cb2175768db37dd8deea36e
Description: utility to retrieve files from the WWW via HTTP and FTP
 Wget [formerly known as Geturl] is a freely available network utility
 to retrieve files from the World Wide Web using HTTP and FTP, the two
 most widely used Internet protocols. It works non-interactively, thus
 enabling work in the background, after having logged off.
.
The recursive retrieval of HTML pages, as well as FTP sites is
 supported -- you can use Wget to make mirrors of archives and home
 pages, or traverse the web like a WWW robot (Wget understands
 /robots.txt).

```

6. Finally, remove this package with the Purge option. For example:

```

debian:/etc# dpkg -P wget
(Reading database ... 17030 files and directories currently installed.)
Removing wget ...
Purging configuration files for wget ...

```

Lab 3-5 Using alien

1. Using FTP, download an RPM file from Red Hat. Below is an example FTP session that will get the `wget` RPM from Red Hat 6.2. Red Hat 7.0 uses a newer RPM format that the `alien` package installed by default with Debian v2.2 does not yet understand.

```

debian:~# ftp ftp.redhat.com
Connected to ftp.redhat.com.
220 "Red Hat FTP server ready. All transfers are logged, please have a nice
day."

```

```
Name (ftp.redhat.com:root): anonymous
331 Guest login ok, send your complete e-mail address as password.
Password: <Email Address>
230-Please read the file README
230- it was last modified on Tue Jan 25 08:51:37 2000 - 370 days ago
230-Please read the file README.roughcuts
230- it was last modified on Fri Jan 21 17:32:20 2000 - 374 days ago
230 Guest login ok, access restrictions apply.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd redhat/redhat-6.2/i386/RedHat/RPMS
250 CWD command successful.
ftp> mget wget*
mget wget-1.5.3-6.i386.rpm? y
200 PORT command successful.
150 Opening BINARY mode data connection for wget-1.5.3-6.i386.rpm (153711
bytes).
226 Transfer complete.
153711 bytes received in 4.04 secs (37.2 kB/s)
ftp> quit
221-You have transferred 153711 bytes in 1 files.
221-Total traffic for this session was 156424 bytes in 2 transfers.
221-Thank you for using the FTP service on ftp.redhat.com.
221 Goodbye.
```

2. The alien package needs to be installed, and you do that with the following apt-get command:

```
debian:~# apt-get install alien
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
 debhelper dpkg-dev file librpm1 patch rpm
The following NEW packages will be installed:
 alien debhelper dpkg-dev file librpm1 patch rpm
0 packages upgraded, 7 newly installed, 0 to remove and 45 not upgraded.
Need to get 853kB of archives. After unpacking 2126kB will be used.
Do you want to continue? [Y/n] y
Get:1 http://http.us.debian.org stable/main dpkg-dev 1.6.15 [73.7kB]
Get:2 http://http.us.debian.org stable/main file 3.28-1 [87.7kB]
Get:3 http://http.us.debian.org stable/main patch 2.5-2.2 [71.3kB]
Get:4 http://http.us.debian.org stable/main debhelper 2.0.86 [101kB]
Get:5 http://http.us.debian.org stable/main alien 6.54 [74.0kB]
Get:6 http://http.us.debian.org stable/main librpm1 3.0.3-1 [184kB]
Get:7 http://http.us.debian.org stable/main rpm 3.0.3-1 [261kB]
Fetched 853kB in 11s (72.8kB/s)
Selecting previously deselected package patch.
(Reading database ... 16992 files and directories currently installed.)
Unpacking patch (from ../patch_2.5-2.2_i386.deb) ...
Selecting previously deselected package dpkg-dev.
Unpacking dpkg-dev (from ../dpkg-dev_1.6.15_all.deb) ...
Selecting previously deselected package file.
```



```
Unpacking file (from ../archives/file_3.28-1_i386.deb) ...
Selecting previously deselected package debhelper.
Unpacking debhelper (from ../debhelper_2.0.86_all.deb) ...
Selecting previously deselected package librpm1.
Unpacking librpm1 (from ../librpm1_3.0.3-1_i386.deb) ...
Selecting previously deselected package rpm.
Unpacking rpm (from ../archives/rpm_3.0.3-1_i386.deb) ...
Selecting previously deselected package alien.
Unpacking alien (from ../archives/alien_6.54_all.deb) ...
Setting up patch (2.5-2.2) ...
Setting up dpkg-dev (1.6.15) ...

Setting up file (3.28-1) ...

Setting up debhelper (2.0.86) ...

Setting up librpm1 (3.0.3-1) ...

Setting up rpm (3.0.3-1) ...

Setting up alien (6.54) ...
```

3. The next step is to convert the RPM package to a .deb package with the following command:

```
debian:~# alien wget-1.5.3-6.i386.rpm
-- Examining wget-1.5.3-6.i386.rpm
-- Unpacking wget-1.5.3-6.i386.rpm
659 blocks
----
-- Automatic package debianization
-- Building the package wget_1.5.3-7_i386.deb
dh_testdir
# Nothing to do.
dh_testdir
dh_testroot
dh_clean -k
dh_installdirs
cp -a `ls |grep -v debian` debian/tmp
dh_installdocs
dh_installexamples
dh_installmenu
dh_installcron
dh_installchangelogs
dh_compress
dh_suidregister
dh_installdeb
dh_shlibdeps
dh_gencontrol
dh_makeshlibs
```

```
dh_md5sums
dh_builddeb
dpkg-deb: building package `wget' in `../wget_1.5.3-7_i386.deb'.

Generation of wget_1.5.3-7_i386.deb complete.
-- Successfully finished
```

4. The final step is to install the new package just like any other.

```
debian:~# dpkg -i wget_1.5.3-7_i386.deb
Selecting previously deselected package wget.
(Reading database ... 17317 files and directories currently installed.)
Unpacking wget (from wget_1.5.3-7_i386.deb) ...
Setting up wget (1.5.3-7) ...
```

Answers to Chapter Questions

Chapter Pre-Test

1. Use the `./configure` script to generate a Makefile.
2. Use the `ldconfig` command to update the `ld.so.cache` file.
3. The tool for converting package formats is called `alien`.
4. `apt-get` can access files from a CD-ROM, a local directory, or from the Internet using HTTP or FTP.
5. To upgrade a package with RPM, use `rpm -U` or `rpm --upgrade`.
6. The RPM package database is stored in `/var/lib/rpm`.
7. Use the `apt-get clean` command to clear the `apt-get` package cache.
8. To add sources to `apt-get`, edit the `/etc/apt/sources.list` file.
9. You can authenticate RPM packages using the GPG, PGP, or MD5 tool.
10. You can query a package's status with `dpkg`.

Assessment Questions

1. **C.** The `./configure` script runs through many tests and creates a Makefile for that specific system. The other options are invalid. See the “Running the configure script” section for more information.
2. **D.** The `make install` command installs software that has already been compiled. The `make` command is used to compile the software. See the “Installing the software” section for more information.

3. **A.** Red Hat created the RPM packaging system. Debian uses `.deb` packages. See the “Red Hat Package Manager” section for more information.
4. **B.** In some circumstances the RPM database can be corrupted, and the `rpm -rebuilddb` command will try to rebuild the database. The other options are invalid. See the “Red Hat Package Manager” section for more information.
5. **A, C, and D.** All three are supported by RPM. See the “Validating package integrity” section for more information.
6. **B.** `rpm -i processor-4.2-i386.rpm` or `rpm --install processor-4.2-i386.rpm`. See the “Installing packages” section for more information.
7. **D.** `ldconfig` creates the `ld.so.cache` file from `ld.so.conf`. See the “Managing Shared Libraries” section for more information.
8. **A.** Dynamic linking does not compile the libraries into the executable like static linking, therefore making the executable smaller. The other options are invalid. See the “Managing Shared Libraries” section for more information.
9. **A and C.** Both `rpm --uninstall` and `rpm -e` remove RPM packages. The other options are invalid. See the “Removing packages” section for more information.
10. **B.** The `spec` file has the compilation options. A `Makefile` is used to compile source code not in RPM format. See the “Creating binary packages from source packages” section for more information.
11. **A.** The `alien` tool converts package files. The `dpkg` tool is used to manipulate packages in Debian. See the “Using alien” section for more information.
12. **B.** The `dpkg --install` command installs Debian `.deb` packages. The `rpm` tool is used with RPM packages. There is no `-I` option for `apt-get`. See the “Installing packages” section for more information.
13. **D.** `dpkg -P` purges the package, which removes all files including the configuration files. See the “Removing packages” section for more information.
14. **A.** The `dselect` tool has all the functionality of `dpkg`, but uses a character-based graphic interface instead of a command-line interface. `apt-get` is used to retrieve and install packages. `gnorpm` is a graphical front end to the `rpm` tool. See the “Using dselect” section for more information.
15. **A.** The sources for `apt-get` are stored in the `sources.list` file. The other options are invalid. See the “Using apt-get” section for more information.
16. **D.** The `update` command checks all the sources in the `sources.list` file and updates the package database accordingly. The `upgrade` command tells `apt-get` to download and install all packages that are newer than those installed on the system. The other options are invalid. See the “Updating the available packages” section for more information.

17. **A, B, C, and D.** The `apt-get` tool can get packages from local NFS and CD-ROM drives and from the Internet via FTP and HTTP. See the “Using `apt-get`” section for more information.
18. **C.** The `autoclean` parameter removes only old packages that can no longer be retrieved. The other options are invalid. See the “Clearing the package archives” section for more information.
19. **A, B, and D.** `alien` supports Red Hat, Debian, and Slackware package formats, but not BSD. See the “Using `alien`” section for more information.
20. **C.** The `-d` option tells `alien` to create a Debian package. The `-r` option specifies RPM, and the `-t` option specifies `.tgz`. See the “Using `alien`” section for more information.

Scenario Answers

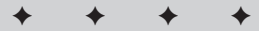
1. You could either compile the source code yourself or convert the RPM package to `.deb` format using `alien`. While either option should create a working application, only by converting the package can you manage the application with the package tools.
2. Since no packages are shown as having been installed, it would appear that the RPM database is incorrect or corrupt. The best option at this point is to run a rebuild with the `rpm -rebuilddb` command.

Getting Around in Linux

Now that you have a working Linux distribution installed, you can move into the use of the system. This includes how to move around efficiently using commands and utilities.

Chapter 4 introduces the tools and utilities used for working with text files. Also covered in this chapter are the tools used for data redirection. Chapter 5 goes in depth into tools and utilities used to create and manage file systems and partitions on Linux systems. Chapter 6 covers file management including the various commands to move, create, and remove files, quotas, and permissions. Chapter 7 covers the Linux documentation available to users on Linux systems. This includes man page documentation located on the Linux system as well as documentation available on the Internet. Chapter 8 provides detailed information on the Linux boot process including troubleshooting and recovery tips. Chapter 9 provides information on X, an application used on Linux systems to provide a graphical environment using window managers.

P A R T



In This Part

Chapter 4
Processing Text

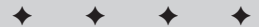
Chapter 5
Using Partitions and
File Systems

Chapter 6
Managing Files

Chapter 7
Using Documentation

Chapter 8
Understanding the
Boot Process

Chapter 9
Using X



Processing Text

4

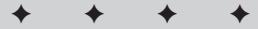
CHAPTER

EXAM OBJECTIVES

Exam 101 ♦ General Linux, Part 1

1.3 GNU and Linux Commands

- **Process Text Streams Using Text Processing Filters.** Send text files and output streams through text utility filters to modify the output in a useful way. Includes the use of standard unix commands found in the GNU textutils package such as sed, sort, cut, expand, fmt, head, join, nl, od, paste, pr, split, tac, tail, tr, and wc.
- **Use Unix Streams, Pipes, and Redirects.** Connect files to commands and commands to other commands to efficiently process textual data. Includes redirecting standard input, standard output, and standard error; and piping one command's output into another command as input or as arguments (using xargs); sending output to stdout and a file (using tee).
- **Perform Searches of Text Files Making Use of Regular Expressions.** Includes creating simple regular expressions and using related tools such as grep and sed to perform searches.



CHAPTER PRE-TEST

1. Which tool would be used to add line numbers to a file?
2. Which tool allows two files to be combined using fields?
3. Which utility is used to create lines of a specified length inside a file?
4. Which utility is used to view a file in reverse?
5. Which utility allows you to delete characters from a file?
6. Which utility allows files to be viewed in hexadecimal format?
7. Which character is used to redirect `stdout`?
8. Which characters are used to redirect `stdout` and `stderr`?
9. Which utility would be used to send a file both to `stdout` and to a file?
10. Which utility is used to alphabetize the contents of a file?

This chapter covers many of the text-processing tools that are available for use on Linux systems. These include the various *filtering utilities*, which are used to search and alter files, as well as the input and output tools. You need to understand the use of these tools, as some of them prove very useful in daily administrative tasks. For example, using the `sed` tool to find and replace text in scripts allows you to make changes much easier than manually performing the task might allow. Others will definitely make an appearance in exam questions, so be sure that you understand the differences and uses of each tool.

Working with Input and Output



1.3 GNU and Linux Commands

- **Use Unix Streams, Pipes, and Redirects.** Connect files to commands and commands to other commands to efficiently process textual data. Includes redirecting standard input, standard output, and standard error; and piping one command's output into another command as input or as arguments (using `xargs`); sending output to `stdout` and a file (using `tee`).

In Linux, you can read data from a file or the terminal or write data to a file or the terminal. Under normal circumstances, every Linux program has three *streams*, or data flow methods, opened for it when it starts up, one for input, one for output, and one for printing diagnostic or error messages. These streams are typically attached to the user's terminal but might instead refer to files or other devices, depending on what the parent process chose to set up.

The input stream, set by default as your keyboard, is referred to as *standard input*. The output stream, set by default to your screen, is referred to as *standard output*. The error stream is referred to as *standard error*. Standard error is usually set to standard output; however, if you would like errors to be sent to a file or other location, you can configure this. These terms are abbreviated to form the symbols used to refer to these files, namely `stdin`, `stdout`, and `stderr`.

Data sent to the utilities discussed in this chapter can often come from either `stdin` or from a file. Many utilities also write the output and errors to `stdout` and `stderr`, respectively. This works well for simply viewing the results; however, if you wish to save these results to a file where they can be accessed later, Linux has a solution to allow this: redirection.

Redirection

It is possible to redirect `stdin`, `stdout`, and `stderr` when working with a Linux shell. The `stdin` is redirected using the `<` symbol. Most utilities work using either information from the `stdin` or from a file, but you can also send data to these

utilities using redirection. An example of using `stdin` redirection is the following, which sends the contents of the file `nameslist` to the `sort` utility where it is sorted alphabetically according to the last name field.

```
# sort +1 < nameslist
Peter Anapol
Jeff Arellano
Scott Bessler
Alex Blauvelt
Michael Craig
Joakim Erdfelt
Johannes Erdfelt
Pete Gizzi
David Goolsby
Thomas McCanta
Angie Nash
Jason Nash
Neil Schroeder
Derek Stutsman
Paul Ward
```

Much more often you will want to redirect `stdout` instead. This is done using the `>` symbol. Simply viewing data in a file in alphabetical order probably isn't as valuable to you as saving that list for future reference. Using redirection, you can have the output saved to another file. The following example uses redirection to send the results of the `sort` command to the file `alphanames`. Then the `cat` command is used to output the contents of the file.

```
# sort +1 nameslist > alphanames
# cat alphanames
Peter Anapol
Jeff Arellano
Scott Bessler
Alex Blauvelt
Michael Craig
Joakim Erdfelt
Johannes Erdfelt
Pete Gizzi
David Goolsby
Thomas McCanta
Angie Nash
Jason Nash
Neil Schroeder
Derek Stutsman
Paul Ward
```

Along with redirecting `stdout` to create a new file you may also want to use redirection of `stdout` to append data to an existing file. This is done using the `>>` symbols. In the example that follows, the contents of the `nicks` file are sorted and then appended to the `alphanames` file. Finally, the `more` command is used to display the contents of the file.

```
# sort nicks >> alphanames
# more alphanames
Peter Anapol
Jeff Arellano
Scott Bessler
Alex Blauvelt
Michael Craig
Joakim Erdfelt
Johannes Erdfelt
Pete Gizzi
David Goolsby
Thomas McCanta
Angie Nash
Jason Nash
Neil Schroeder
Derek Stutsman
Paul Ward
Alchemist
Bammage
Dave
Dragonstr
Jackyl
Johan
Just_joe
Lordram
Netchick
Netjunkie
Pri
Rexmortis
Sting
Thomas
Zaphod
```

It is also possible to redirect `stderr` along with `stdout`. To redirect `stderr` and `stdout` to create a new file, you use the `>&` symbols. For example:

```
sort nameslist >& sortednameslist
```

If there are no errors, the contents of the files containing the redirection will appear the same as the contents of the files without the `stderr` redirection included when there are no errors. However, any errors that occur will be included in these files. The following example is for a file that was created with no errors during processing.

```
# cat sortednameslist
angie
denise
jill
judy
katrina
kim
lisa
monica
nikki
```

If you wish to redirect only `stderr` to a file, then you would use the `2>` symbols. For example, to redirect only the `stderr` for the `sort` command used previously, use the following:

```
sort nameslist 2> nameslisterrors
```

This is also useful if you wish to run a command while redirecting `stderr` and `stdout` to two separate files. The following example would send `stderr` to the `nameslisterrors` file while sending `stdout` to the `sortednameslist` file.

```
sort nameslist 2> nameslisterrors > sortednameslist
```

Pipes

Along with redirecting data to a file, you may want to use other commands at times. This procedure can be done using the pipes character (`|`). This allows you to use one command line to send output data from one command as input data to another. In the following example, the file `nameslist` is sorted alphabetically, and then line numbers are added to each name using the `nl` command.

```
# sort +1 nameslist | nl
 1 Peter Anapol
 2 Jeff Arellano
 3 Scott Bessler
 4 Alex Blauvelt
 5 Michael Craig
 6 Joakim Erdfelt
 7 Johannes Erdfelt
 8 Pete Gizzi
 9 David Goolsby
10 Thomas McCanta
11 Angie Nash
12 Jason Nash
13 Neil Schroeder
14 Derek Stutsman
15 Paul Ward
```

This data can also be piped into another utility or redirected to a file.

```
# sort +1 nameslist | nl > numberednames
# more numberednames
 1 Peter Anapol
 2 Jeff Arellano
 3 Scott Bessler
 4 Alex Blauvelt
 5 Michael Craig
 6 Joakim Erdfelt
 7 Johannes Erdfelt
```

```

8 Pete Gizzi
9 David Goolsby
10 Thomas McCanta
11 Angie Nash
12 Jason Nash
13 Neil Schroeder
14 Derek Stutsman
15 Paul Ward

```

This example shows how pipes and redirection can be used together to process data using multiple utilities and then to save the output to another file. Along with `stdout`, which is sent to another utility using the pipes symbol, `stderr` can also be included and sent to the utility using the `|&` symbols. For example, the following command sorts the contents of `nameslist`, adds line numbers, and appends any error information.

```
sort +1 nameslist |& nl
```

tee

The `tee` utility copies standard input to standard output and also to any files given as arguments, so that `stdin` appears in both `stdout` and the file. This is useful when you want not only to send some data down a pipe, but also to save a copy to a file. If a file you're writing to does not already exist, it is created. If the file you're writing to already exists, the data it previously contained is overwritten unless the append option is used. The options used with this utility are shown in Table 4-1.

Table 4-1
Options Used with tee

<i>Option</i>	<i>Use</i>
<code>-a</code>	Appends standard input to the specified file.
<code>-i</code>	Ignores interrupt signals that would be used to stop or restart processes.



The `tee` utility is useful for saving output along the way as utilities are processed. You can reference this output later, which may prove useful for troubleshooting purposes.

In the following example, the file `nameslist` is first sorted alphabetically and then piped to `tee` so that the output is also saved to the file `abcnames`. The data is then piped to `nl` so that each line is numbered, and the output is then redirected to a file named `abc123names`.

```
# sort +1 nameslist | tee abcnames | nl > abc123names
# more abcnames
Peter Anapol
Jeff Arellano
Scott Bessler
Alex Blauvelt
Michael Craig
Joakim Erdfelt
Johannes Erdfelt
Pete Gizzi
David Goolsby
Thomas McCanta
Angie Nash
Jason Nash
Neil Schroeder
Derek Stutsman
Paul Ward
# more abc123names
 1 Peter Anapol
 2 Jeff Arellano
 3 Scott Bessler
 4 Alex Blauvelt
 5 Michael Craig
 6 Joakim Erdfelt
 7 Johannes Erdfelt
 8 Pete Gizzi
 9 David Goolsby
10 Thomas McCanta
11 Angie Nash
12 Jason Nash
13 Neil Schroeder
14 Derek Stutsman
15 Paul Ward
```

xargs

The `xargs` utility is used to pass a large number of arguments to other commands. The `xargs` utility reads arguments from the standard input, delimited by blanks (the blanks function as normal characters when used with double or single quotes or a backslash) or line feed characters, and executes the command one or more times with any initial arguments followed by arguments read from standard input. Blank lines on the standard input are ignored. This enables a command to process more arguments than it could normally handle. In the following example the `locate` database is searched, using the `locate` link, for all README files. Using `xargs`, the filenames are then sent to the `cat` utility, which will then display the file contents. All of the text of these files is sent to the `fmt` utility, which formats the data for the line length of 60 characters. The output is then redirected to the file `/home/angie/readmes`.

```
locate README | xargs cat | fmt -60 > /home/angie/readmes
```



The `xargs` utility is used to feed arguments to another utility at a rate that it can handle. Using this command when it isn't needed won't produce any errors.

Modifying Text with Filters

Objective

1.3 GNU and Linux Commands

- **Process Text Streams Using Text Processing Filters.** Send text files and output streams through text utility filters to modify the output in a useful way. Includes the use of standard unix commands found in the GNU textutils package such as `sed`, `sort`, `cut`, `expand`, `fmt`, `head`, `join`, `nl`, `od`, `paste`, `pr`, `split`, `tac`, `tail`, `tr`, and `wc`.
- **Perform Searches of Text Files Making Use of Regular Expressions.** Includes creating simple regular expressions and using related tools such as `grep` and `sed` to perform searches.

Linux offers a variety of tools to use for processing and filtering text. These tools enable you to search for data and then manipulate it depending on the tool you use. The tools can be used from the command line or placed in scripts, which are executed to perform the needed tasks. The output is then either sent to standard output, where it can be redirected or piped into another utility, or sent to a file. The flexibility in the input and output of these tools enables you to customize the text for a variety of uses. Regardless of how many filters are run, the original text remains intact.

When working with text files you will likely use a *pager* such as `more` or `less`. These utilities are used to display text one page at a time. The `cat` utility can also be used to display the contents of a text file. These utilities aren't filters but are often helpful when working with them.

Sorting lines of a file

The `sort` utility sorts lines of text and displays them to standard output. It is used to sort, merge, and compare lines from files or standard input. This utility is used with the following syntax:

```
sort -option filename
```

Table 4-2 lists the options commonly used with the `sort` utility.

Table 4-2
Options Used with sort

<i>Option</i>	<i>Use</i>
-b	Ignores leading blanks.
-c	Checks to see whether the file is sorted.
-d	Considers only alphanumeric characters and sorts in phone directory order.
-f	Folds lowercase to uppercase characters.
-m	Merges already sorted files without resorting them.
-M	Compares sorted files.
-n	Sorts numerically.
-o FILE	Writes output to the specified file instead of standard output.
-r	Reverses the results.
--help	Displays help and exits.
--version	Displays version and exits.

An example of the use and output of the `sort` utility is as follows:

```
# sort list
3
32
41
5
90
Celestial Seasons
Centrum
Whiskas
bread
butter
eggs
flour
ice
milk
sugar
```

As you can see in this example, by default, lists are sorted first by numeric order followed by alphabetical order, with capital letters before lowercase. In the following example, the `-f` option tells `sort` to ignore case.

```
# sort -f list
3
32
```

```
41
5
90
bread
butter
Celestial Seasons
Centrum
eggs
flour
ice
milk
sugar
Whiskas
```

In the following example, two sorted files are merged. As you can see, the files are not resorted but are simply combined.

```
#sort -m alphanicks alphanames
angie
birdgrlmom
crystalmoon
denise
desteve
jcfraggle
jill
judy
katrina
kim
lisa
lma
loudhouse
monica
netchickie
nikki
nikks
trinityz
```

It is also possible to sort files based on fields. Fields can be separated by spaces or tabs and are numbered starting at zero. When sorting based on fields, the + symbol precedes the field number with each field being separated by spaces. In the following example, the `nameslist` file is sorted by the second field (+1) and then by the first field (+0):

```
#cat nameslist
Scott Bessler
Jason Nash
Angie Nash
Derek Stutsman
Jeff Arellano
Paul Ward
Alex Blauvelt
Peter Anapol
```

```
David Goolsby
Michael Craig
Johannes Erdfelt
Thomas McCanta
Joakim Erdfelt
Pete Gizzi
Neil Schroeder
# sort +1 +0 nameslist
Peter Anapol
Jeff Arellano
Scott Bessler
Alex Blauvelt
Michael Craig
Joakim Erdfelt
Johannes Erdfelt
Pete Gizzi
David Goolsby
Thomas McCanta
Angie Nash
Jason Nash
Neil Schroeder
Derek Stutsman
Paul Ward
```

The use of these fields allows a great deal of flexibility in sorting lists in files. It is important to remember that the `sort` utility does not change the original file. Output is sent to the standard output where it can be viewed or redirected to another command or file.

Cutting text

The `cut` utility is used to write selected parts of a file to standard output. The `cut` utility can be used to select columns or fields from specified files. It is possible to select a specific line address, several line addresses, or a range of line addresses. Table 4-3 covers the various options used with the `cut` utility.

Table 4-3
Options Used with `cut`

<i>Option</i>	<i>Use</i>
-b	Outputs only the specified bytes range.
-c	Outputs only the specified characters.
-f	Outputs only the specified fields, which are delimited by tabs.
-help	Displays help and then exits.
-version	Displays version information and then exits.

The following is an example of the use of these ranges to output only the first ten characters of each line:

```
# cut -c 1-10 nameslist
Scott Bess
Jason Nash
Angie Nash
Derek Stut
Jeff Arell
Paul Ward
Alex Blauv
Peter Anap
David Gool
Michael Cr
Johannes E
Thomas McC
Joakim Erd
Pete Gizzi
Neil Schro
```

The `cut` utility can specify bytes, characters, or fields in files to be displayed.

Pasting text

The `paste` utility enables you to join text from multiple files. Corresponding lines of the specified file are written to standard output with each line separated by a tab character. The options used with the `paste` utility are shown in Table 4-4.

Table 4-4
Options Used with `paste`

Option	Use
<code>-s</code>	Pastes lines from one file at a time.
<code>-d <i>delimiter-list</i></code>	Uses the characters specified in <i>delimiter-list</i> consecutively instead of the tab character when separating merged files.

The following is an example of using the `paste` utility:

```
# paste alphanames nicks
Peter Anapol      Bamage
Jeff Arellano    Rexmortis
Scott Bessler     Jackyl
Alex Blauvelt    Dragonstr
Michael Craig     Alchemist
Joakim Erdfelt   Just_joe
```

```

Johannes Erdfelt      Johan
Pete Gizzi           Sting
David Goolsby        Dave
Thomas McCanta       Thomas
Angie Nash           Netchick
Jason Nash            Netjunkie
Neil Schroeder       Pri
Derek Stutsman       Zaphod
Paul Ward             Lordram

```

Converting tabs to spaces

When creating a file on Linux the tab function is seen as one key, but the actual tab settings can vary from system to system. This can change the appearance of a file based on the system displaying the file. In order to create a uniform appearance in files, you may want to replace the tabs within a file with a specified number of spaces. This will cause the file to appear the same on all systems. The `expand` utility is used to expand tabs to spaces. This utility can work with text either in a file or on the standard input. The options for the `expand` utility are shown in Table 4-5.

Table 4-5
Options Used with `expand`

<i>Option</i>	<i>Use</i>
<code>-I</code>	Specifies that only tabs at the beginning of a line are converted.
<code>-t</code>	Uses a set of numbers to specify the location of tabs that are to be converted.
<code>--help</code>	Displays help information and then exits.
<code>--version</code>	Displays version information and then exits.

The default action of the `expand` utility is to convert all tabs within a file to eight spaces. Some examples of the `expand` utility are the following:

```

# expand marital
Scott Bessler
Jason Nash      M
Angie Nash     M
Derek Stutsman M
Jeff Arellano  S
Paul Ward      M
Alex Blauvelt  S
Peter Anapol   M
David Goolsby  S

```

```

Michael Craig   S
Johannes Erdfelt   S
Thomas McCanta  S
Joakim Erdfelt  S
Pete Gizzi      M
Neil Schroeder  S

```

In this example the default setting of eight spaces is used. Because the name Johannes Erdfelt has 16 characters, the tab is expanded to the 24th character field. In the following example, the setting of 9 spaces for the tab field allows the marital status fields to line up evenly.

```

# expand -t 9 marital
Scott Bessler   S
Jason Nash     M
Angie Nash     M
Derek Stutsman M
Jeff Arellano  S
Paul Ward      M
Alex Blauvelt  S
Peter Anapol   M
David Goolsby  S
Michael Craig  S
Johannes Erdfelt S
Thomas McCanta S
Joakim Erdfelt S
Pete Gizzi     M
Neil Schroeder S

```

Formatting paragraphs

The `fmt` utility formats each paragraph in a file and sends the output to the standard output. This utility is used to specify the width of lines and joins or separates lines in an effort to produce lines that are all the same length. The `fmt` utility attempts to separate lines at the end of a sentence. When this isn't possible, it avoids breaking lines after the first word or before the last word of the sentence. Sentences are determined to end with either a period, exclamation, or question mark (!?) followed by two spaces or a line feed character. The entire paragraph is read before any line breaks are introduced.

The default line width used with `fmt` is 75 characters. The default width can be overwritten using the appropriate option with the `fmt` utility. Table 4-6 covers some of the options used with this utility.

Table 4-6
Options Used with `fmt`

<i>Option</i>	<i>Use</i>
<code>-c</code>	Preserves the indentation at the beginning of a paragraph and aligns the paragraph with the left margin of the second line.
<code>-t</code>	Works like the <code>-c</code> option except if the indentation of the second line in a paragraph matches that of the first, the second line is considered to be a one-line paragraph.
<code>-s</code>	Specifies that lines are only to be split, not joined.
<code>-u</code>	Specifies that uniform spacing be used; this reduces spacing between all words to one space and spaces between sentences to two spaces.
<code>-NUMBER</code> or <code>-w NUMBER</code>	Sets the width of lines to the <i>NUMBER</i> specified.
<code>-p PREFIX</code>	Specifies that only lines beginning with <i>PREFIX</i> are to be formatted.

The `fmt` utility also preserves blank lines, indentations, and spacing so that any special formatting contained within the document is not changed. An example of the use of the `fmt` utility is shown below. As you can see in the example, all lines between paragraphs are maintained, and the utility has attempted to produce lines that are all 40 characters wide.

```
# fmt -40 mydoc
```

```
Linux offers a variety of tools to
use for processing and filtering text.
These tools enable you to search for
data and then manipulate it depending
on the tool you use. The tools can be
used from the command line or placed in
scripts, which are executed to perform
the needed tasks. The output is then
either sent to standard output, where
it can be redirected or piped into
another utility, or sent to a file.
The flexibility in the input and
output of these tools enables you to
customize the text for a variety of
uses. Regardless of how many filters
are run, the original text remains
intact. When working with text files
you will likely use a pager such as
more or less. These utilities are used
to display text one page at a time.
```

The `cat` utility can also be used to display the contents of a text file. These utilities aren't filters but are often helpful when working with them.

Deleting or substituting characters

There are times when you may want to search a document for specific characters and then either delete or replace them. An example of this use would be a document that contains both uppercase and lowercase characters, but you would prefer all characters to be lowercase. The `tr` utility provides this functionality using the following syntax:

```
tr option set1 set2
```

The options used with `tr` are shown in Table 4-7.

Table 4-7
Options Used with `tr`

Option	Use
<code>-d</code>	Deletes the characters specified.
<code>-s</code>	Replaces a sequence of characters with one character.
<code>--help</code>	Displays help information and then exits.
<code>--version</code>	Displays version information and then exits.

The following is an example of the use of the `tr` utility. First, the contents of the file are viewed using the `more` utility. The file contains a mixture of both uppercase and lowercase characters. In an effort to avoid confusion all uppercase characters are replaced with their lowercase equivalents using the `tr` utility.

```
# more
mkdir MyFiles
cd MyFiles
ls -al MyFiles > AllOfMyFiles

# tr 'A-Z' 'a-z' < mycommands
mkdir myfiles
cd myfiles
ls -al myfiles > allOfmyfiles
```


Viewing the beginning of a file

The `head` utility allows you to view the beginning of a file to aid with identification or other purposes. By default, the first ten lines of a file are displayed. It is important to remember that a line is everything before the line feed character, so the actual output could be much more than ten lines of display. Using options, it is possible to specify the number of lines displayed among other things. The options used with the `head` utility are covered in Table 4-8.

Table 4-8
Options Used with `head`

<i>Option</i>	<i>Use</i>
<code>-c NUMBER</code>	Displays the first <i>NUMBER</i> of bytes specified.
<code>-n NUMBER</code>	Displays the first <i>NUMBER</i> of lines specified.
<code>-q</code>	Specifies that the header is not to be displayed.
<code>-v</code>	Specifies that the header is to be displayed.
<code>--help</code>	Displays help information and exits.
<code>--version</code>	Displays version information and exits.

An example of the use of this file is shown below. In this example the first three lines of the `nameslist` file are displayed. The `-v` option is used to display the filename located in the header.

```
# head -n 3 -v nameslist
==> nameslist <==
Scott Bessler
Jason Nash
Angie Nash
```

Viewing the end of a file

There is also a utility that allows you to view the end of a file. Like `head`, the `tail` utility displays the last ten lines of a file by default. There are also several options used with the `tail` utility; these options, which are displayed in Table 4-9, allow more control of the data displayed.

Table 4-9
Options Used with tail

Option	Use
<code>-NUMBER</code>	Specifies the <i>NUMBER</i> of lines from the end of the file to begin with when printing.
<code>+NUMBER</code>	Specifies the <i>NUMBER</i> of lines from the beginning of the file to begin with when printing.
<code>--retry</code>	Instructs the utility to keep trying to open a file even if it is inaccessible. This option must be used with a name and will cause <code>tail</code> to retry even if the file doesn't exist.
<code>-c NUMBER</code>	Displays the last <i>NUMBER</i> of bytes specified.
<code>-f</code>	Specifies that <code>tail</code> should run in a loop so that appended data is displayed. This option is used with files that are growing in size because new data is being appended to the end.
<code>-n NUMBER</code>	Displays the last <i>NUMBER</i> of files specified.
<code>-q</code>	Specifies that headers are not to be displayed.
<code>-v</code>	Specifies that headers are to be displayed.
<code>--help</code>	Displays help information and then exits.
<code>--version</code>	Displays version information and then exits.

An example of the use of this utility is shown below. In this example the utility is used to display the last 50 bytes of the `nameslist` file.

```
# tail -c 50 nameslist
McCanta
Joakim Erdfelt
Pete Gizzi
Neil Schroeder
```

Joining multiple files

The `join` utility actually searches both files for common entries. The entries found in both files are then displayed to `stdout` where they can be redirected to a file. You can combine files using fields. The `join` utility uses *join fields* to combine lines from multiple files. Before using the `join` utility, the files must be sorted on the join fields. This is often done by using the `sort` utility based on the fields that are to be joined. So, if you are utilizing two files that contain both first and last names and wish to join the files using last names, then the two files should first be sorted alphabetically using the last name field.

The correct syntax of the `join` utility is shown below. The output produced consists of one line for each pair of input lines. The default join field used is the first line of each file.

```
join -options FILE1 FILE2
```

Some of the options used with the utility are shown in Table 4-10.

Table 4-10
Options Used with join

Option	Use
-I	Specifies that case is to be ignored when combining files.
-1 <i>FIELD</i>	Specifies the <i>FIELD</i> in <i>FILE1</i> to use when joining files.
-2 <i>FIELD</i>	Specifies the <i>FIELD</i> in <i>FILE2</i> to use when joining files.
-t <i>CHAR</i>	Specifies that the character <i>CHAR</i> is to be used as the separator of the input and output fields.
-v <i>FILE#</i>	Instructs that a line is to be printed for each unpairable line located in the specified <i>FILE#</i> .
--help	Displays help information and exits.
--version	Displays version information and exits.

Following is an example of the use of this utility.

```
# cat list_1
drew
jason
joe
john
mike
todd
# cat list_2
derek
jason
john
pete
scott
# join list_1 list_2
jason
john
```

Dividing files into multiple pieces

The `split` utility is used to divide one long file into many different files. This utility creates files all of a certain length, with a default length of 1,000 lines, and sequentially names the files. The filenames are made of a prefix, which is `x` by default, followed by a letter combination. The letter combination follows the pattern of `aa`, `ab`, `ac`, and so on. If there are more than 676 files being created, the syntax used is `zaa`, `zab`, and so on.

The `split` utility is capable of using either standard input or a file as input; when no file is specified, standard input is used. The correct syntax when using the `split` utility is as follows:

```
# split -options INPUT PREFIX
```

Several options, shown in Table 4-11, can be used to customize the output of the `split` utility.

Table 4-11
Options Used with `split`

Option	Use
<code>-l LINES</code>	Specifies the number of lines contained in each output file.
<code>-b BYTES</code>	Specifies the number of bytes to be placed in each file. If the number is followed by <code>b</code> , the amount is multiplied by 512; if a <code>k</code> is used, the number is multiplied by 1024; and if an <code>m</code> is used, the number is multiplied by 1,048,576.
<code>-C BYTES</code>	Works like the <code>-b</code> option except as many completed lines of data are placed in the file as possible without exceeding the number of <code>BYTES</code> specified.
<code>--verbose</code>	Writes diagnostics to standard error before each file is opened.
<code>--help</code>	Displays help information and exits.
<code>--version</code>	Displays version information and exits.

In the following example the `nameslist` file is split into several files each containing five lines.

```
# split -l 5 nameslist names
# ls names*
namesaa
namesab
```

```
namesac
nameslist
# cat namesaa
Scott Bessler
Jason Nash
Angie Nash
Derek Stutsman
Jeff Arellano
```

Displaying files in other formats

There are times that you might want to display files in nontext format. For displaying files in octal and other formats, the `od` utility is useful. Each line of output consists of the *offset*, which is the number of input bytes skipped before formatting and writing by default in the input. This is followed by groups of data from the file. By default, `od` prints the offset in octal, and each group of file data is two bytes of input printed as a single octal number. The options for this utility are shown in Table 4-12.

Table 4-12
Options Used with `od`

Option	Use
<code>-A RADIX</code>	Specifies the base to use for displaying the file. Any of the following can be used for <i>RADIX</i> : <ul style="list-style-type: none"> d is used for decimal o is used for octal x is used for hexadecimal n is used for none
<code>-j BYTES</code>	Specifies the input <i>BYTES</i> to skip before displaying the file.
<code>-N BYTES</code>	Specifies the maximum <i>BYTES</i> of input to display.
<code>-s [N]</code>	Instead of the normal output, the string constants are output. This is at least <i>N</i> (3 by default) consecutive ASCII graphic characters, followed by a null (zero) byte.
<code>-t TYPE</code>	Selects the format in which to output the file data. <i>TYPE</i> is a string of one or more of the following type indicator characters: <ul style="list-style-type: none"> a is for named character c is for ASCII character or backslash escape d is for signed decimal f is for floating point o is for octal u is for unsigned decimal x is for hexadecimal

Option	Use
<code>-w BYTES</code>	Outputs the specified number of <i>BYTES</i> per line. The default width is 32 bytes.
<code>--help</code>	Displays help information and exits.
<code>--version</code>	Displays version information and exits.

The following is an example of the use of the `od` utility. In this example, the `nicks` file is displayed in hexadecimal format with a width of eight bytes.

```
# od -A x -w8 nicks
000000 060502 066555 063541 005145
000008 062522 066570 071157 064564
000010 005163 060512 065543 066171
000018 042012 060562 067547 071556
000020 071164 040412 061554 062550
000028 064555 072163 045012 071565
000030 057564 067552 005145 067512
000038 060550 005156 072123 067151
000040 005147 060504 062566 052012
b000048 067550 060555 005163 062516
000050 061564 064550 065543 047012
000058 072145 072552 065556 062551
000060 050012 064562 055012 070141
000068 067550 020144 046012 071157
000070 071144 066541 000012
000075
```



You should pay close attention to these rarely used utilities such as `od` and `join`, as they make excellent exam material.

Converting files for printing

The `pr` utility formats and prepares files for printing. The `pr` utility writes each file to standard output, paginating and optionally outputting the file in multicolumn format. It also can merge all files, printing all in parallel, one per column. An example of the correct syntax for this utility is as follows:

```
pr -options FILE
```

By default, a five-line header is printed at each page: two blank lines, a line with the date, the filename, and the page count, and two more blank lines. A footer of five blank lines is also printed. Several options can be used to specify the formatting produced with the `pr` utility. Several of these options are shown in Table 4-13.

Table 4-13
Options Used with pr

Option	Use
- COLUMN	Produces <i>COLUMN</i> -sized columns and balances the number of lines in the columns on each page.
-a	Prints columns across rather than down.
-d	Double-spaces the output.
-f	Uses form feeds instead of newline characters to separate pages.
-h HEADER	Uses a centered <i>HEADER</i> instead of the filename as the header.
-l PAGELength	Sets the number of lines per page to <i>PAGELength</i> .
-m	Prints all files in parallel, one per column.
-N NUMBER	Starts counting with <i>NUMBER</i> at first line of first page printed.
-w WIDTH	Sets page width to <i>WIDTH</i> (default value is 72) characters for multiple text-column output only.
-W WIDTH	Sets page width to <i>WIDTH</i> (default value is 72) characters always.

Displaying files backwards

The `tac` utility is used to display lines of a file in reverse where, by default, a new line begins after the line feed character. This is the opposite of the `cat` command in spelling and function.

The syntax for the `tac` utility is as follows:

```
# tac -options file
```

The options used with this utility are shown in Table 4-14.

Table 4-14
Options Used with tac

Option	Use
-b	Attaches the separator to the beginning of the line that precedes it in the file.
-r	Treats the separator string as a regular expression.
-s SEPARATOR	Uses the <i>SEPARATOR</i> character instead of the line feed character as the record separator.

Displaying numeric details of a file

The `wc` utility counts the number of bytes, white space-separated words, and line feed characters in each given file. It prints one line of counts for each file, and if the file was given as an argument, it prints the filename following the counts. If more than one file is given, the utility prints a final line containing the cumulative counts, with the filename total. The counts are printed in this order: line feed characters, words, and bytes. By default, each count is output right-justified in a 7-byte field with one space between fields so that the numbers and filenames line up nicely in columns. Several of the options available with the `wc` utility are shown below in Table 4-15.

Table 4-15
Options Used with `wc`

<i>Option</i>	<i>Use</i>
<code>-c</code>	Displays only the byte count.
<code>-w</code>	Displays only the word count.
<code>-l</code>	Displays only the line count.
<code>-L</code>	Displays the length of the longest line.
<code>--help</code>	Displays help information and then exits.
<code>--version</code>	Displays version information and then exits.

The following are examples of the use of the `wc` utility.

```
# wc nameslist
15      30      203 nameslist

#wc -c nameslist
203 nameslist

#wc -L nameslist
16 nameslist
```

Adding line numbers to a file

The `nl` utility is useful for displaying line numbers in a file. This utility displays a file with line numbers added to each line. The utility breaks its input into *logical pages*. By default, the line number is reset to 1 at the top of each logical page. It treats all of the input files as a single document and does not reset line numbers or logical pages between files.

A *logical page* consists of three sections: header, body, and footer. Any of the sections can be empty. Each can be numbered in a different style from the others. An empty line of output replaces a section separator. Any text that comes before the first section separator string in the input file is considered to be part of a body section, so the `nl` utility treats a file that contains no section delimiters as a single body section.

Several options can be used to customize the output gained using this utility, and several of those options are covered in Table 4-16.

Table 4-16
Options Used with `nl`

Option	Use
<code>-a</code>	Numbers all lines.
<code>-t</code>	Numbers only nonempty lines.
<code>-n</code>	Does not number lines (the default for headers and footers).
<code>-i NUMBER</code>	Increments line numbers by <i>NUMBER</i> with the default value of one.
<code>-p</code>	Does not reset line numbers at the start of each logical page.
<code>-s STRING</code>	Adds the <i>STRING</i> after the added line number.
<code>-v NUMBER</code>	Sets the initial line number on each logical page to <i>NUMBER</i> .
<code>-w NUMBER</code>	Specifies the <i>NUMBER</i> of character spaces to reserve for line numbers. The default value is six.

Below is an example of the use the `nl` utility. In this example the `nicks` file is printed with a number corresponding to each line.

```
# nl nicks
 1 Bamage
 2 Rexamortis
 3 Jackyl
 4 Dragonstr
 5 Alchemist
 6 Just_joe
 7 Johan
 8 Sting
 9 Dave
10 Thomas
11 Netchick
12 Netjunkie
13 Pri
14 Zaphod
15 Lordram
```

Using the stream editor

The `sed` utility is a *stream editor* that takes input either from a file or from data that is piped into the utility. The `sed` utility works globally within a file unless addressing symbols are used to limit the scope of the command. That the utility works globally means that every instance matching the specified pattern is replaced. Addressing can be used to specify the location that is to be searched for a matching pattern. This addressing can specify either a line or a range of lines in a file. You can also exclude line addresses from the search using the exclamation symbol (!). Regular expressions can also be used to specify locations within a file.

The `sed` command can be used to make simple substitutions and more powerful changes to a file. Simple substitutions throughout a file are made using the following syntax:

```
sed -option s/REGEXP/replacement/flag filename
```

The `sed` command will work using text from standard input as well as text from specified files. The original file is left intact, and the changes are written to a new file. *REGEXP* stands for *regular expression*, which is a way of searching for particular characters. Regular expressions are discussed in the following section. The `s` command instructs the `sed` utility to locate the *REGEXP* and remove that while adding the replacement in its place. You can perform multiple substitutions using the `-e` option. Two examples of how to use this are:

```
sed -e 's/lisa/Lisa/' -e 's/nikki/Nikki/' myfriends
sed -e 's/lisa/Lisa/'; 's/nikki/Nikki/' myfriends
```

These commands search the `my friends` file and replace the characters *lisa* and *nikki* with *Lisa* and *Nikki*, respectively.

You can also use the `sed` utility to run a script on a file by using the `-f` option. This enables you to store frequently used options and simplifies larger commands. The `-f` option is used as shown below:

```
sed -f scriptname filename
```

When used with a script, the script must contain the `sed` command(s) in the following format:

```
s/REGEXP/replacement/flags
```

The various options for the `sed` utility are covered in Table 4-17.

Table 4-17
Options Used with sed

Option	Use
-V	Displays version information and then exits.
-h	Displays help information and then exits.
-n	Prevents the file from being displayed after it has been processed.
-e <i>command</i>	Appends the commands to those being processed.
-f <i>file</i>	Appends the commands in the specified script file to those being processed.

In addition to using these options, flags can be used with the `s` command. These flags allow for further configuration of the command. The flags available for use with the `s` command are shown in Table 4-18.

Table 4-18
Flags Used with s

Flag	Use
g	Applies the changes globally.
p	Prints all lines that contain a substitution. (Normally used with the <code>-n</code> option.)
NUMBER	Replaces only the NUMBER match.
w <i>filename</i>	Writes all lines with substitutions to the file specified.
I	Ignores case when matching <i>REGEXP</i> .

An example of the use of a flag with the `sed` utility is the following command that will only replace the fifth occurrence of the word `stop` with the word `STOP` in `myfile`.

```
sed s/stop/STOP/5 myfile
```

The `sed` utility also uses addresses to specify the location of a file change. The addresses can be line numbers, line addressing symbols, or regular expressions. The various types of addressing that can be used with the `sed` utility are shown in Table 4-19. These addresses precede the command in the `sed` utility. An example of the correct use for an address is the following:

```
sed '3,10s/REGEXP/replacement/' filename
```

In this example the third through tenth lines of the filename are searched for *REGEXP*, and if found, *REGEXP* is replaced with *replacement*.

Addressing can also be used to exclude *REGEXP* matches from being replaced. This is done with the ! (exclamation) character. The following example shows how to ignore the third occurrence of the *REGEXP*.

```
# sed `!3s/REGEXP/replacement/' filename
```



Pay careful attention to the correct use of regular expressions when using *sed*. You should understand the various ways a string can be located.

Table 4-19
Addressing Used with *sed*

Address	Use
<i>number</i>	Specifies the line number to match.
<i>number</i> , <i>number</i>	Specifies the line numbers to match and includes all lines located between these numbers.
\$	Specifies to match the last line.
!	Matches all lines except for the lines specified.

Using *grep*

The *grep* utility is used to search files for the pattern specified. The default action of the utility is to print the matches to the search. The *grep* utility can accept filenames to search or it can search data from standard input. This utility is used with the following syntax:

```
grep -options [-e searchpattern] [-f filename]
```

This utility is used with three variants controlled by the options shown in Table 4-20.

Table 4-20
Options Used with *grep*

Option	Use
-G	The default behavior which interprets the pattern as a basic regular expression.
-E	Interprets the pattern as an extended regular expression. This option functions the same as the -G option with GNU <i>grep</i> .
-F	Interprets the pattern as a list of fixed strings.

This utility can be extremely helpful for locating text within a file. Combined with the functionality provided with regular expressions, `grep` helps make it possible to locate a wide range of patterns easily.

Enhancing Searches with Regular Expressions

Objective

1.3 GNU and Linux Commands

- **Perform Searches of Text Files Making Use of Regular Expressions.** Includes creating simple regular expressions and using related tools such as `grep` and `sed` to perform searches.

Regular expressions are patterns of characters, some which have special meaning, that are useful when using text filters. They allow for increased flexibility in searches with the use of special characters. Most characters used with regular expressions represent themselves; for example, the character *r* is used to represent the letter *r*. Some characters, however, are special and can be used to represent other characters or groups of characters. These special characters are known as *metacharacters*. These special characters look like the wildcard characters used by the shell; however, the characters have different functions when used in regular expressions. When using regular expressions at a shell prompt, you surround them with single quotation marks to differentiate them. Several of the metacharacters used in regular expressions are shown in Table 4-21.

**Table 4-21
Metacharacters**

Metacharacter	Use
<code>\</code>	The slash is used to locate any of the characters following the slash.
<code>*</code>	The asterisk matches zero or more occurrences of the preceding regular expression. The zero or more occurrence matching is useful when using this character along with others. For example, when searching for <code>*are</code> , matches for <i>are</i> and <i>stare</i> are returned.
<code>.</code>	A dot matches any single character; this character is used as a wildcard.
<code>^</code>	The caret is used to locate the start line; it is often followed by another character to locate a line starting with that character. Using this metacharacter to search for <code>^A</code> would locate all lines beginning with A.
<code>\$</code>	The dollar sign locates the end of the line, and when preceded by another character, it will locate lines ending with that character. So, using <code>a\$</code> will locate all lines ending with a.

Metacharacter	Use
[]	Brackets are used to locate specific characters; a range of characters can also be specified within brackets. When a range is specified such as 1-5, the numbers 1, 2, 3, 4, and 5 are specified.
[^]	Brackets with a caret as the first character contained between them search for all characters except those that are also contained within the brackets. So, for [^1-9] all characters are found except for the numbers ranging from one to nine.
\{ \}	These symbols are used to locate a range or specific number of instances. The expression a\{3\} will search for <i>aaa</i> while the expression a\{1,3\} will locate <i>a</i> , <i>aa</i> , and <i>aaa</i> .
\< \>	The slash and less than symbols are used with a set of characters followed by the slash and greater than symbols. The characters that are located within the symbols are searched for at the word boundary. This allows you to locate complete words, regardless of where they are located within a sentence.

**Caution**

It is important to understand that although these characters may appear to be the same as wildcard characters used at a shell prompt, they function differently.

The following are some examples of how to use these metacharacters. The following example will search for the string *an*. The search is case-sensitive, so *An* would not be located. The words *an*, *angie*, *and*, and *man* would all be located.

```
\an
```

If you wished to locate the strings *an* and *An* you could use the following regular expression. This expression would locate *an*, *An*, *Angie*, *man*, and so on.

```
\[Aa]n
```

It is also possible to search for just the word *an* using the following example. This would locate the words *An* and *an*.

```
\<[Aa]n\>
```

If you want to locate all words ending with the letters *nation* you could use the following:

```
\*nation
```

This would locate the words *nation*, *imagination*, and so on.

Using the following example you could search for all words beginning with *a* and ending with the letter *l*. This search would include the words *all* and *accidental*.

```
\a*.l
```

These examples are some ways in which regular expressions can be used. You can combine metacharacters in regular expressions to create more powerful searches, such as shown by the example that combined a word search with two possible beginning characters. Some of these metacharacters are relatively new and will not work with all utilities, such as older versions of `vi`, so it is important to verify that they work for the utility you are using.

Key Point Summary

This chapter covers many important utilities for processing text on a Linux system, which enables you to manipulate almost all files. Some of the most important items covered in this chapter include the following:

- ♦ By default, data is sent to utilities using standard input, `stdin`, output is sent to standard output, `stdout`, and error messages are sent to standard error, `stderr`.
- ♦ It is possible to use redirection on `stdin`, `stdout`, and `stderr`. `stdin` is redirected using `<`, `stdout` is redirected with `>`, `stderr` is redirected along with `stdout` using `>&`, and `stderr` alone can be redirected using `2>`.
- ♦ Pipes are used to send data output from one utility as input to another utility.
- ♦ The `tee` utility is used to send output from a utility both to `stdout` and to a file.
- ♦ The `xargs` utility allows a utility to process more arguments than that utility would normally be able to handle.
- ♦ Linux includes a number of utilities for editing and processing data files. These include the following: `sed`, `sort`, `cut`, `paste`, `expand`, `fmt`, `tr`, `head`, `tail`, `join`, `split`, `od`, `pr`, `tac`, `wc`, and `nl`.
- ♦ The `sed` utility is a stream editor that allows data files to be edited from a shell prompt or script file.
- ♦ The `sort` utility allows files to be sorted numerically or alphabetically.
- ♦ The `cut` utility sends lines from a data file to `stdout`.
- ♦ The `paste` utility is used to combine lines of data files.
- ♦ The `expand` utility is used to change tabs to spaces so that formatting will be uniform regardless of the system displaying the file.
- ♦ The `fmt` utility is used to create lines in a file of the same length.

- ♦ The `tr` utility allows files to be searched for specific characters and then either deletes or replaces the characters.
- ♦ The `head` utility is used to view the beginning of a file.
- ♦ The `tail` utility is used to view the end of a file.
- ♦ The `join` utility is used to combine fields from multiple files.
- ♦ The `split` utility is used to divide files into multiple parts of equal length.
- ♦ The `od` utility allows files to be viewed in octal and other formats.
- ♦ The `pr` utility formats files to prepare them for printing.
- ♦ The `tac` utility displays files line by line from last to first.
- ♦ The `wc` utility is used to count the words, lines, and bytes of a file.
- ♦ The `nl` utility adds line numbers to a file.
- ♦ Regular expressions create powerful tools for specifying and searching for text.



STUDY GUIDE

The following questions and exercises will allow you to review the information covered in this chapter. Take your time when completing this section of the chapter, carefully reviewing any questions that you may have. Simply answering the question correctly is not as important as understanding the answer, so review any material that you might still be unsure of. Being comfortable with the questions and answers presented here will help you be more prepared for the certification exam questions.

Assessment Questions

1. Which of the following would be used to locate the following words: *and*, *made*, *named*, *standard*?
 - A. `a*d`
 - B. `a.*d`
 - C. `[a..d]`
 - D. `a.d`
2. Which command would search lines 2 through 20 of the file records for the characters *1st* and replace them with the characters *first*?
 - A. `sed 's2-20/1st/first/' records`
 - B. `sed '2-20s/1st/first/' records`
 - C. `sed 's2,20/1st/first/' records`
 - D. `sed '2,20s/1st/first/' records`
3. Which command would separate the file `researchpaper` into multiple files, each containing 60 lines? (Select all that apply.)
 - A. `split -60 researchpaper`
 - B. `split -C 60b researchpaper`
 - C. `split -C 60 researchpaper`
 - D. `split -l 60 researchpaper`

4. Which utility is used to combines the lines from two files? (Select all that apply.)
- A. `split`
 - B. `join`
 - C. `paste`
 - D. `cut`
5. Which of the following would send the data from the `ls` command to the file `myfiles`?
- A. `ls | myfiles`
 - B. `ls > myfiles`
 - C. `ls < myfiles`
 - D. `ls | xargs myfiles`
6. Which of the following would be used to view the last five lines of the file `myfiles`?
- A. `tac myfiles`
 - B. `tail myfiles`
 - C. `tac -5 myfiles`
 - D. `tail -5 myfiles`
7. Which of the following allows you to view the file `myfiles` in octal format? (Select all that apply.)
- A. `od myfiles`
 - B. `od -t o myfiles`
 - C. `od -t x myfiles`
 - D. `od -o myfiles`
8. The _____ utility is used to allow a utility to handle more arguments than it normally could.
9. Which of the following are used to redirect `stdout` and `stderr` to a file?
- A. `<&`
 - B. `>&`
 - C. `|\&`
 - D. `&&`

10. Which of the following would alphabetize the file `mylist`, number the list, and then separate it into files each with 60 lines?
- A. `sort mylist | nl > -60 lists`
 - B. `sort mylist > nl > split -60 > lists`
 - C. `sort mylist | nl | split -60 lists`
 - D. `sort mylist | nl | tee lists | split -60 lists`
11. The _____ utility is used to view a file in reverse.
12. Which of the following is a metacharacter used with regular expressions? (Select all that apply.)
- A. -
 - B. .
 - C. *
 - D. _
13. Which utility is used to provide a total count of all lines in a file?
- A. `nl`
 - B. `ln`
 - C. `wc`
 - D. `tr`
14. The _____ utility is used to ensure that files appear the same, regardless of the system used to view them, by changing tabs to spaces.
15. Which utility attempts to create lines of equal length throughout a file?
- A. `nl`
 - B. `ln`
 - C. `fmt`
 - D. `expand`
16. Which utility would be easily used to replace all lowercase letters in a file with uppercase letters?
- A. `cut`
 - B. `sed`
 - C. `tac`
 - D. `tr`

17. Which of the following is used to verify that a file is alphabetized?
 - A. `sort -c`
 - B. `sort -d`
 - C. `sort -v`
 - D. `sort -m`
18. The _____ utility can provide information used for troubleshooting by saving output piped into another command.
19. The `*` symbol used in regular expressions works the same as the `*` wildcard used at the shell prompt.
 - A. True
 - B. False
20. Which of the following should surround regular expressions to ensure that they are processed properly?
 - A. parentheses
 - B. backslashes
 - C. double quotation marks
 - D. single quotation marks

Scenarios

1. Your company stores a large number of small files in a directory on a Linux system. You need to combine the contents of these files in one large file. You will then need to split this large file into multiple files of equal length. How would you do this?
2. You need to print the large file created in Scenario 1 so that each page is numbered and has a header containing your company name and the current date. How would you complete this task?

Lab Exercises

Lab 4-1 Using filtering tools

In this lab you will work with the `/etc/services` file. You will change this file by using several of the text filtering tools covered in this chapter.

1. Examine the contents of the `/etc/services` file using the `cat` command.

2. Using the `sed` utility, locate and replace all of the `#` characters with a blank character.
3. Then send this data to the `tee` command using pipes to create a file named `alteredservices`.
4. Pipe the results of that utility to the `sort` utility so that all services are arranged in alphabetical order.
5. Then save this data to a file named `alphaservices`. All these tasks can be accomplished with the following command:

```
cat /etc/services | sed 's/#//'| tee alteredservices | sort >
alphaservices
```

This command causes the contents of `/etc/services` to be displayed on the terminal. The file is then processed with `sed` so that all `#` symbols are replaced with a blank space. The output is then sent to a file named `altered-services` containing the symbol replacements, and it is also sent to standard output. This file is then sorted and sent to a file named `alphaservices`. This file contains the services file in alphabetized form. A portion of this file is shown below:

```

                100 - reserved
24 - private
26 - unassigned
3com-tsmux      106/tcp          poppassd
3com-tsmux      106/udp          poppassd
afbackup        2988/tcp          Afbbackup system
afbackup        2988/udp          Afbbackup system
afpovertcp     548/tcp          AFP over TCP
afpovertcp     548/udp          AFP over TCP
alias here. This should work for programs asking for this service.
amanda         10080/udp        amanda backup services
amandaidx      10082/tcp        amanda backup services
amidxtape      10083/tcp        amanda backup services
are included, only the more common ones.
asp            27374/tcp        Address Search Protocol
asp            27374/udp        Address Search Protocol
at-echo       204/tcp          AppleTalk echo
at-echo       204/udp
at-nbp        202/tcp          AppleTalk name binding
at-nbp        202/udp
at-rtmp       201/tcp          AppleTalk routing
at-rtmp       201/udp
```

Answers to Chapter Questions

Chapter Pre-Test

1. The `nl` utility is used to number the lines in a file.
2. The `join` utility is used to join fields from multiple files.
3. The `fmt` command is used to create lines of equal length in a file.
4. The `tac` command displays a file backwards, the opposite of `cat`.
5. The `tr` and `sed` utilities are used to delete and substitute characters in a file.
6. The `od` utility is used to view files in octal, hexadecimal, and other formats.
7. The `>` character is used to redirect `stdout` to a file.
8. The `>&` characters are used to redirect `stdout` and `stderr` to a file.
9. The `tee` utility is used to send the results of a command both to `stdout` and to a file.
10. The `sort` utility is used to arrange the contents of a file in alphabetical and numerical order.

Assessment Questions

1. **B.** In regular expressions, the `.` is used to specify any character and the `*` is used to specify any occurrence of the previous character including no occurrences. See the “Enhancing Searches with Regular Expressions” section for more information.
2. **D.** When using addressing with the `sed` utility the line numbers and ranges are specified before the `s` command. The range is separated by a comma. See the “Sorting lines of a file” section for more information.
3. **A and D.** The `-l` option is used to specify the number of lines contained in each file using `split`. However, when no option is given the number specified is assumed to be the number of lines. See the “Dividing files into multiple pieces” section for more information.
4. **B and C.** The `paste` and `join` utilities are used to combine lines from a file. The `split` command is used to divide a file into multiple pieces. `cut` removes text from a file. See the “Pasting text” and “Joining files” sections for more information.
5. **A.** Pipes are used to send data from one command to another. See the “Pipes” section for more information.

6. **D.** The `tail` command is used to view the end of a file and the `-5` option specifies the number of lines to view. See the “Viewing the end of a file” section for more information.
7. **A and B.** The `od` utility is used to view files in octal format by default. The `-t` option, followed by `o`, is used to specify octal format. See the “Displaying files in other formats” section.
8. **xargs.** The `xargs` utility allows a command to handle more arguments than it would normally be able to process. See the “xargs” section for more information.
9. **B.** The `>&` characters are used to send both `stdout` and `stderr` to a file. See the “Redirection” section for more information.
10. **D.** The pipes are used to send data from one command to another, so answers A and B are incorrect. Answer C does not create two files, so only D is correct. See the “Pipes” section for more information.
11. **tac.** The `tac` utility is used to view a file starting at the last line and ending with the first. See the “Displaying files backwards” section for more information.
12. **B and C.** The `.` and `*` are both metacharacters used with regular expressions. See the “Enhancing Searches with Regular Expressions” section for more information.
13. **C.** The `wc` utility is used to provide totals of a file including word count, line count, and byte count. See the “Displaying the numeric details of a file” section for more information.
14. **expand.** The `expand` utility is used to convert tab characters to spaces. See the “Converting tabs to spaces” section for more information.
15. **C.** The `fmt` utility attempts to create lines of equal length throughout a file. See the “Formatting paragraphs” section for more information.
16. **D.** The `tr` utility is used to delete and replace characters in a file. See the “Deleting or substituting characters” section for more information.
17. **A.** Use the `-c` option with the `sort` utility to verify that a file has been sorted. See the “Sorting lines of a file” section for more information.
18. **tee.** The `tee` utility is used to send output from a command to both a file and `stdout`. See the “tee” section for more information.
19. **B.** The metacharacters used in regular expressions have different meanings and uses than wildcards. See the “Enhancing Searches with Regular Expressions” section for more information.
20. **D.** Regular expressions should be surrounded by single quotation marks. See the “Enhancing Searches with Regular Expressions” section for more information.

Scenarios

1. `ls | xargs cat | tee largefile | split -60`

The `xargs` utility is used for working with large amounts of data. The `tee` command sends the data both to a file and to `stdout`. Finally the data is split into pages each containing 60 lines.

2. `pr -h "Companyname" largefile`

The `pr` command automatically prints the date and page number as part of the header information. To add the company name to the header the `-h` option is used with the company name surrounded by quotation marks. The argument, `largefile`, is the name of the file created in the previous scenario that is now being displayed using the `pr` command.

Using Partitions and File Systems

EXAM OBJECTIVES

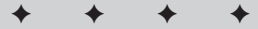
Exam 101 ♦ General Linux, Part 1

2.4 Devices, Linux File Systems, Filesystem Hierarchy Standard

- **Create partitions and filesystems.** Create disk partitions using `fdisk`, create hard drive and other media filesystems using `mkfs`.
- **Maintain the integrity of filesystems.** Verify the integrity of filesystems, monitor free space and inodes, fix simple filesystem problems. Includes commands `fsck`, `du`, `df`.
- **Control filesystem mounting and unmounting.** Mount and unmount filesystems manually, configure filesystem mounting on bootup, configure user-mountable removable filesystems. Includes managing file `/etc/fstab`.

5

C H A P T E R



CHAPTER PRE-TEST

1. Which file contains information on currently mounted file systems?
2. Which command is used to display the number of inodes on a file system?
3. Which command is used to mount all file systems listed in the `/etc/fstab` file?
4. What is the default file system type for Linux partitions?
5. Which utility is used to create a partition on a Linux system?
6. Which utility is used to format a partition using the Linux swap file system?
7. Which file contains the file systems mounted automatically by the operating system?
8. Which type of partition is bootable?
9. What is the device name used by the second partition on the second IDE drive?
10. Drive information such as labels, blocks, and inode tables is stored where?

This chapter covers the tools and practices associated with file systems and their management. Simply put, a file system is a way for the operating system to arrange files on the storage media so it can find them when it needs them. You use these tools to create, maintain, and control the file systems. These tools and concepts are used throughout the book and are important when working with Linux systems. Several exam objectives are covered in this chapter so you need to have a thorough understanding of all material presented here.

Linux File Systems Overview

As you have already discovered, Linux file systems are discussed throughout the book. File systems are a primary component of all operating systems, and understanding the function and use of these systems is important. You will be required to know the differences in the various file systems as well as the proper use of each.

A variety of media can contain files for a Linux system. Hard disks, CD-ROMs, floppies, network drives, and other removable media can all be used to store files. Each of these media uses file systems for organization. These file systems arrange files in a tree-like directory structure with subdirectories branching from the root directory. The media and operating system used to store files dictate the file system used. Linux can support a wide variety of media and file systems depending on the configuration of the Linux kernel. With this support, you can create, access, and modify these file systems.



This chapter covers the usage of file system tools while Chapter 13 details how to install the kernel options.

The `mount` command is used to connect various other file systems to the primary Linux file system, which currently defaults to the `ext2` file system. The root user has control over the location of these additional file systems. The root user can grant standard users the proper access to mount specific file systems such as CD-ROMs and floppy disks for use on the system. When working with removable devices, remember that each disk must be mounted to be used. If you wish to access another disk, you must first unmount the current disk, swap the disks in the drive, and then mount the new disk for use. You can also configure file systems to be mounted automatically when the system is started. This ability is useful when working with file systems stored on the network or on local hard disks in the system. The tools and files used to allow these functions are covered later in this chapter.

File system types

Many different file systems can be accessed using Linux systems. Table 5-1 shows many of these file systems and their use.

Table 5-1
Linux File Systems

<i>File System</i>	<i>Use</i>
ext2	Linux file system.
iso9660	CD-ROM file system.
minux	Minix file system.
msdos	16-bit MS-DOS FAT file system.
vfat	32-bit Windows FAT file system utilizing long filenames.
hpfs	OS/2 file system.
proc	Linux processes file system.
nfs	Network file system used for accessing remote systems.
swap	Linux swap file system.
sysv	UNIX System V file system.

These file systems can be thought of as languages. Linux is multilingual but must know the correct language to speak when communicating with each file system. As you can see, Linux supports file systems used by various other operating systems. This is useful for a system that dual boots with these other operating systems. Using the support for these file systems you can access non-Linux partitions for reading and writing files. The NTFS file system used by Windows NT and Windows 2000 is one file system not listed in Table 5-1; however, support for this file system is currently being developed for Linux systems to allow for read and write access to these partitions as well. The Reiser file system, reiserfs, is another not listed in Table 5-1. This is a journaling file system used by some Linux systems to allow for greater recovery in case of a system failure. This file system will be included with later versions of the Linux kernel. Later in this chapter you will learn how to specify the file system using files and commands.



For more information on reiserfs go to <http://www.reiserfs.org>

Along with the option of the file system type, understanding the device names used when working with storage media is important. Table 5-2 shows the various device prefixes used with Linux devices.

Table 5-2
Device Names

<i>Device Name</i>	<i>Use</i>
hd	IDE hard drive partitions
sd	SCSI hard drive partitions
sr	SCSI CD-ROM drives
fd	Floppy drives
st	SCSI tape drives
ht	IDE tape drives
tty	Terminals
lp	Printers
pty	Remote terminals
js	Joystick ports
midi	MIDI ports
ttyS	Serial ports
cua	COM ports
cdrom	CD-ROM drives. This is often just a link to the actual IDE or SCSI device.
modem	Modem devices

The prefixes are combined with a device number. With hard disks partitions the drive is specified with a letter such as *a* for the first drive, *b* for the second drive, and so on. The partition is specified with a number such as *1* for the first partition, *2* for the second partition, and so on. Examples of this naming:

```
hda1    The first partition on the first IDE hard drive.
hdb2    The second partition on the second IDE hard drive.
cdrom   The first CD-ROM drive.
cdrom1  The second CD-ROM drive.
sda1    The first partition on the first SCSI hard drive.
fd0     The first floppy disk drive.
```

These device names are used to address devices on a Linux system. All devices are stored in the `/dev` directory. The device names can be linked to other devices; for example, `cdrom` may be linked to `/dev/sr0` if there is a SCSI CD-ROM installed on the system. These links allow for standard addressing of devices on systems. By examining the contents of the `/dev` directory you can view the links and exact locations of the devices on a system.



Exam questions will reference devices, so understanding this naming scheme is important.

Considerations when making a file system

Remember several considerations when creating a new file system. The file system does not just contain the data of the files stored on the disk. Some of the disk is used for overhead associated with file systems. This includes space for pointers that store the location of data that comprises files, as well as the file system size and label. All of this information uses space on the disk drive. The default settings for these components can affect what is stored on the partition, so you need to understand these components before creating a new partition. It is more difficult to correct problems once the partition has been created and data is stored in the file system.

Inodes

The pointers used to identify the location of data stored in files are known as *inodes*. These are used for UNIX-based file systems and are not used with FAT file systems. When you create a file system, the inodes to be used are also created. This sets the number of files that can be stored on the file system. Unless you specify the number of inodes to be created, Linux will try to determine the number of inodes needed based on the partition size. This can cause wasted space if the file system will contain a small number of large files. Disk space can also be lost if the file system will store a large number of small files. Once the inodes have been filled, no new files can be created. The default inode configuration allows for the partition to be filled with files that are 2K in size.



Be sure that you understand the importance of inodes. Once you have run out of inodes on a file system, no new files can be created and the rest of the space on the file system will be unusable.

Superblocks

The inodes for a file system are stored within the superblock. The superblock is a record that also contains information about file system size and location on the disk. Other important file system configuration information such as cylinders and disk block usage is also stored here. The information stored within the superblock is crucial for accessing the file system. Because of this, several copies of the superblock are stored throughout the disk. This provides fault tolerance so that if one superblock is damaged another can be used and the file system can be restored. A backup of the superblock is always stored every 8K blocks of the file system.

Creating Partitions and File Systems

Objective**2.4 Devices, Linux File Systems, Filesystem Hierarchy Standard**

- **Create partitions and filesystems.** Create disk partitions using `fdisk`, create hard drive and other media filesystems using `mkfs`.

When working with disk drives you need to perform several steps before the disk is usable by the system. First, you must partition the disk; this allows the drive to be structured for data storage. Once the disk has been segmented into partitions, you can create the file system. Linux provides the tools necessary to create the partitions and the file systems on a hard disk. This section covers these tools and how to use them.

Partition types

Hard disk drives used by Linux and other systems follow standard partitioning strategies. The partition information is stored on the physical disk and allows several different operating systems to coexist on a single system. Disk partitioning is useful for a variety of reasons. You can store system data on separate partitions to ensure that all of the various parts of the system have the disk space necessary. Keeping user and system data on separate partitions also allows for a degree of safety by providing a physical separation between space accessed by normal users and that accessed by the system.

The reasons for partitioning data are too numerous to list. They can vary from issues related to security, politics, and physical disk issues. Regardless of the reasons for creating partitions, the types of partitions remain the same. Disks can contain primary, extended, and swap partitions.

Primary

All hard disks containing file systems use a *primary partition*. This is the first partition created on a disk. If all disk space is used by the primary partition, it will be the only partition located on the disk. It is possible to have multiple primary partitions on one physical disk. These partitions are used for booting and are limited to four on a physical disk.

Extended

If more than four partitions are needed, you need to create an *extended partition*. When an extended partition exists on a disk, you can have only three primary partitions. An extended partition by itself is not useful. It acts as a container for logical partitions and can hold many of these logical drives. These partitions are not bootable but enable you to have a large number of partitions on the system. The logical partitions can exist only inside of the extended partition.

Swap

Linux systems also use up to eight swap partitions. These partitions are used to store temporary data and increase system performance. A swap partition is used as virtual memory and required for a system with less than 16MB of RAM. In the past, the recommended size of the swap partition was equal to the amount of RAM in the system or 16MB, whichever were larger. It is now often recommended that the swap partition be equal to twice the size of the RAM, so a system with 128MB of RAM would need a swap partition of at least 256MB. Kernels earlier than 2.2 were limited to 128MB swap partitions; however, beginning with version 2.2, the swap partition on i386-based systems is 2GB. Linux systems combine the amount of RAM and the swap partition to determine the total amount of virtual memory available to the system. The optimal amount of virtual memory needed by a system varies depending on the applications you're using.



Remember that RAM is faster than the swap partition contained on the hard disk. If you are working with applications that consume large amounts of memory, you will probably want to invest in adding more RAM to the system.

File system tools

Several tools are used for creating partitions and file systems on Linux systems. The `fdisk` utility is used for dealing with partitions on hard disks. The `fips` utility is used for resizing partitions, and file systems are created using the `mkfs` utility. You can use these tools together to prepare hard disks for use on a Linux system.



Many Linux distributions include tools that automatically create the Linux partitions and file systems during installation. Some people never find it necessary to utilize the `fdisk` or `mkfs` utilities. However, the advanced user and systems administrator need to be familiar with the tools.

fdisk

The `fdisk` utility is used to manage partitions on Linux systems. This is a command-line tool with the function of the MS-DOS `fdisk`; however, they are two distinct tools with different options and usage. Table 5-3 covers some of the commands used with the `fdisk` utility.

Table 5-3
Commands Used with `fdisk`

<i>Option</i>	<i>Use</i>
p	Displays partition information.
d	Deletes a partition.
n	Creates a new partition.

Option	Use
q	Exits without saving.
w	Saves changes and exits.
m	Displays commands.
v	Verifies the partition table.
a	Toggles the bootable flag.

The proper use of the `fdisk` utility is shown below. In this example the first IDE hard drive is examined, the second partition is deleted and then recreated as an extended partition. Then the utility is exited with the changes saved.

```
# fdisk /dev/hda
The number of cylinders for this disk is set to 2495.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help):p
Disk /dev/hda: 255 heads, 63 sectors, 2495 cylinders
Units = cylinders of 16065 * 512 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/hda1  *           1           578     4642753+  b   Win95 FAT32
/dev/hda2                579        2495    15398302+  f   Win95 Ext'd (LBA)
/dev/hda5                579           580       16033+   83   Linux
/dev/hda6                581        1101     4184901    7   HPFS/NTFS
/dev/hda7                1102        1957     6875788+  b   Win95 FAT32
/dev/hda8                1958        2467     4096543+  83   Linux
/dev/hda9                2468        2495     224878+   82   Linux swap

Command (m for help): d
Partition number (1-10): 2
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
e
Partition number (1-4): 3
First cylinder (1-8190, default 1): 579
Last cylinder or +size or +sizeM or +sizeK (4000-8190, default 8190):
2495Command (m for help): w
```

To run the `fdisk` utility, you must be logged in as root. When displaying partition information, the physical disk information and file system information are also displayed. The commands are issued within the `fdisk` utility that is run with the device as the argument. None of the changes are saved until the `w` command is issued. In the example the partitions are displayed and the third partition is deleted.

fips

The `fips` utility is not a Linux utility. This utility is not covered on the exam but is useful to know when adding Linux to an existing system. It is a MS-DOS utility provided with Red Hat to resize partitions on hard disks. This utility produces two partitions, the resized original partition and a new partition created from the free space. You then need to delete the newly created partition so the space can be used for Linux. Before using the `fips` utility it is important to back up data stored on the current partition. The utility is located on the Red Hat CD-ROM in the `dosutils` directory. The utility is run from the MS-DOS command line, and the arrow keys are used to resize the current partitions. Care should be taken when using this utility.



The `fips` utility is similar to PartitionMagic and other commercial software that is used to resize and create new partitions.

mkfs

Once you have created a partition, the file system must be added so that Linux can use the space. The `mkfs` utility is used for creating file systems on bare partitions. The `mkfs` utility is used with several different options, which are shown in Table 5-4. This is actually just a front end for the various utilities covered in Table 5-5.

Table 5-4
Options Used with `mkfs`

<i>Option</i>	<i>Use</i>
<code>-V</code>	Produces verbose output, including all file system specific commands that are executed.
<code>-t fstype</code>	Specifies the type of file system to be built. If not specified, the default file system type (currently ext2) is used.
<code>fs-options</code>	File system-specific options to be passed to the real file system builder. Although not guaranteed, most file system builders support the following options. <ul style="list-style-type: none"> <code>-c</code> – Check the device for bad blocks before building the file system. <code>-l filename</code> – Read the bad blocks list from filename. <code>-v</code> – Produce verbose output.

The options used by `mkfs` are followed by an argument specifying the partition that is to be formatted. After the command has been run an exit code of 0 (zero) will indicate success while the exit code of 1 (one) indicates failure. An example of the syntax used is as follows:

```
mkfs -option argument
```

When creating a file system using `mkfs`, several commands can be used to specify the file system type. Table 5-5 shows the various commands to use when creating file systems. As you can see, there are specific commands to run for each file system type you want to create. These commands are accessible only by the root user.



The utilities to understand for the test, and for real world usage, are covered here. Each file system type is created using a specific command. Be sure to know which commands are used for creating ext2, swap, and MS-DOS file systems.

Table 5-5
Utilities for File System Creation

<i>Command</i>	<i>Use</i>
<code>mkfs.ext2</code> or <code>mke2fs</code>	Creates an ext2 file system.
<code>mkfs.msdos</code> or <code>mkdosfs</code>	Creates an MS-DOS file system.
<code>mkswap</code>	Creates a Linux swap partition file system.
<code>mkraid</code>	Initializes and upgrades RAID device arrays.
<code>mkfs.minix</code>	Creates a Minix file system.
<code>mkfs.bfs</code>	Creates a SCO BFS file system.



More details on the options available when creating each file system can be located in the Linux man pages.

To create a file system the correct tool specified in Table 5-5 must be used. An example of the proper use of these utilities is creating an ext2 partition using `mkfs.ext2` as shown below:

```
# mke2fs /dev/hda3
```

Checking the File System

Objective

2.4 Devices, Linux File Systems, Filesystem Hierarchy Standard

- **Maintain the integrity of filesystems.** Verify the integrity of filesystems, monitor free space and inodes, fix simple filesystem problems. Includes commands `fsck`, `du`, `df`.

You only need to create a file system occasionally; most people configure their partitions and create file systems only once. The ongoing process is maintaining the file system. Linux provides several useful tools for verifying, monitoring, and fixing file systems. In this section we examine the tools used for these purposes: `fsck`, `du`, and `df`.

fsck

The `fsck` utility is a file system checker utility that is useful for examining file systems to locate and possibly repair problems that are found. When specifying multiple file systems, the checks are run in parallel, unless you use the `-s` option to specify that they should be processed serialized. The `fsck` utility is similar to `mkfs` in that a different utility is used for each file system. Table 5-6 covers the `fsck` utilities and their use.

Table 5-6
fsck Utilities

<i>Utility</i>	<i>Use</i>
<code>e2fsck</code> or <code>fsck.ext2</code>	Linux ext2 file system checker
<code>dosfsck</code> or <code>fsck.msdos</code>	MS-DOS file system checker
<code>fsck.minix</code>	Minix file system checker

When running these utilities you can use several options to control the actions taken. Table 5-7 shows the options used with the `fsck` utilities.

Table 5-7
Options Used with fsck

<i>Option</i>	<i>Use</i>
<code>-p</code>	Automatically repairs without prompting.
<code>-n</code>	Makes no changes to the file system.

Option	Use
-y	Assumes yes to all questions.
-c	Checks for bad blocks.
-f	Forces a check even if the file system is marked clean.
-r	Interactively prompts for changes.
-v	Be verbose.
-b SUPERBLOCK	Uses an alternative superblock.
-l BADBLOCKS-FILE	Adds to bad blocks list.
-L BADBLOCKS-FILE	Sets bad blocks list.
-A	Checks all file systems listed in <code>/etc/fstab</code> .
-C	Displays a completion/progress bar. Currently only works for ext2 file systems.
-N	Doesn't execute, simply displays what would be done.
-P	Used with <code>-A</code> ; processes the root file system in parallel with others.
-R	Used with <code>-A</code> ; causes the root file system to be skipped.
-V	Displays version information.

These options are especially useful for specifying bad blocks and the superblock. The ability to specify an alternative superblock allows for the file system to be recovered in the case of a corrupt or damaged superblock. Copies of the superblock are stored throughout the disk, and an alternative superblock can be specified. As mentioned earlier in this chapter, a backup superblock is stored at throughout the file system.

During the `fsck` check of a system the following is done:

- ♦ Pass 1: Checking inodes, blocks, and sizes
- ♦ Pass 2: Checking directory structure
- ♦ Pass 3: Checking directory connectivity
- ♦ Pass 4: Checking reference counts
- ♦ Pass 5: Checking group summary information

Normally during system boot, `fsck -p` is run to check all file systems listed in the `/etc/fstab` file. This will automatically detect and fix problems with inodes, link counts, data blocks, and the superblock. If a more serious error is located, `fsck -p` will ask for help and then exit. Some of the errors that will cause this are as follows:

- ♦ Blocks claimed by multiple files
- ♦ Blocks claimed outside of the file system
- ♦ Too few link counts
- ♦ Unaccounted blocks
- ♦ Directories that correspond to unallocated inodes
- ♦ Format errors

In cases where the parent directory of a file cannot be determined, the file will be placed in `/lost+found`. Files here are renamed with their inode numbers. Examining the contents of this directory can be useful when missing files after a system error.

The exit information presented by the `fsck` utility provides useful information about the results of the operation. Each code represents a specific exit condition for the utility. The actual code returned is the sum of the exit conditions. The exit codes are displayed on the command line when the command has finished its operation. The exit codes are shown in Table 5-8.

Table 5-8
fsck Exit Codes

Code	Meaning
0	No errors.
1	File system errors corrected.
2	System should be rebooted.
4	File system error left uncorrected.
8	Operational error
16	Usage or syntax errors.
128	Shared library error.

The `fsck` utility should not be run on currently mounted devices. When checking a device it is recommended that you first unmount the device and then run `fsck`. The following examples show the correct use of the `fsck` utility. First, the drive is unmounted so that `fsck` can access the partition. The first example performs a

verbose check of the first partition on the second IDE drive. The drive is marked clean, and the number of used files versus possible files and used blocks versus total blocks is shown.

```
# umount /fun
# fsck.ext2 -v /dev/hdb1
e2fsck 1.18, 11-Nov-1999 for EXT2 FS 0.5b, 95/08/09
/dev/hdb1: clean, 9692/1237888 files, 1754206/2474002 blocks
```

In the second example a check is verbosely forced on the same partition. The drive is carefully examined and more detailed information regarding files, links, and blocks is displayed.

```
# fsck.ext2 -v -f /dev/hdb1
e2fsck 1.18, 11-Nov-1999 for EXT2 FS 0.5b, 95/08/09
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information

    9692 inodes used (0%)
    157 non-contiguous inodes (1.6%)
        # of inodes with ind/dind/tind blocks: 1194/205/0
1754206 blocks used (70%)
    0 bad blocks

    8884 regular files
    776 directories
    0 character device files
    0 block device files
    1 fifo
    0 links
    22 symbolic links (22 fast symbolic links)
    0 sockets
-----
    9683 files
# mount /fun
```

When you're finished checking the drive, you must mount it for it to be available for use by the system.

du

The `du` utility is used to report the amount of disk space used by the files and sub-directory for the specified directory. If no directory is specified for the utility, `du` works with the `pwd`, or present working directory. The user must have read access to the files and directories in order for this utility to run. The function of the `du` utility can be controlled using a variety of options, which are covered in Table 5-9.

Table 5-9
Options Used with du

<i>Option</i>	<i>Use</i>
-a	Shows counts for all files and directories.
-b	Displays size in bytes.
-c	Prints a total for all arguments after they are processed.
-h	Creates human readable output, appending letters such as M for megabytes.
-k	Displays size in kilobytes.
-m	Displays size in megabytes.
-s	Displays a summary total for each argument.
-x	Skips directories containing file systems other than the one contained in the argument.

The following example shows the correct usage of the `du` utility to report the space used by the `/usr` directory.

```
# du -sh /usr
1007M /usr
```

df

The `df` utility is used to display disk space used and available on mounted file systems. When no options are used, the `df` utility reports the space used and available on all currently mounted file systems. A variety of options can be used to control the output produced by the `df` utility; these options are shown in Table 5-10.

Table 5-10
Options Used with df

<i>Option</i>	<i>Use</i>
-a	Shows counts for all file systems.
-t <i>FILESYSTEM</i>	Limits the listing to those of the specified file system type.
-h	Creates human readable output, appending letters such as M for megabytes.
-k	Displays size in kilobytes.

Option	Use
-m	Displays size in megabytes.
-i	Displays inode usage information.
-l	Limits the listing to those in the local file system.
-x <i>FILESYSTEM</i>	Excludes the specified file system from the listing.

Following are some examples of output produced using the `df` utility. The first example is using the `df` utility with no options. The second example produces human readable sizes for the file systems listed. The third example displays inode information for all mounted file systems.

```
# df
Filesystem            1k-blocks      Used Available Use% Mounted
on
/dev/hda8              4096380      1469176   2627204   36% /
/dev/hda5              15522        3710     11011    25% /boot
/dev/hdb1             9740592      6861408   2384384   74% /fun

# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/hda8             3.9G  1.4G  2.5G   36% /
/dev/hda5             15M   3.6M   11M   25% /boot
/dev/hdb1             9.3G  6.5G  2.3G   74% /fun

# df -i
Filesystem            Inodes   IUsed   IFree IUse% Mounted on
/dev/hda8            4294967295      0 4294967295    0% /
/dev/hda5              4016         27   3989    1% /boot
/dev/hdb1            1237888     9692 1228196    1% /fun
```



Be sure to know the option used to display inode information for the file systems.

Mounting and Unmounting File Systems

Objective

2.4 Devices, Linux File Systems, Filesystem Hierarchy Standard

- **Control filesystem mounting and unmounting.** Mount and unmount filesystems manually, configure filesystem mounting on bootup, configure user-mountable removable filesystems. Includes managing file `/etc/`.

The last step required for using a file system is that it must be *mounted*. Mounting a file system makes it available to the system and the user. This section covers the utilities used for mounting and unmounting file systems. Also covered here are the files that contain information for file systems on a Linux system. These tools are used on a daily basis when working with Linux systems and are of particular interest when preparing to use a system and when preparing for the exam.

Mounting file systems

The `mount` command is used for mounting file systems on Linux. The standard form of the `mount` command is as follows:

```
mount -t type device mountpoint
```

This command specifies the file system type, the device containing the file system, and the directory where the file system is to be located. A common location for mounting file systems on many distributions is the `/mnt` directory. This directory can be configured with subdirectories such as `/mnt/floppy` and `/mnt/cdrom`. An example of the use of this command is shown in the following with the `/dev/hda2` device using the `ext2` file system to the mount point of `/fun`.

```
# mount -t ext2 /dev/hda2 /fun
```

Several options can be used with the `mount` command; these are covered in Table 5-11.

Table 5-11
Options Used with mount

Option	Use
-V	Displays version information.
-h	Displays help information.
-v	Mounts the file system verbosely.
-a	Mounts all file systems listed in <code>/etc/fstab</code> .
-f	Fakes the mounting of file systems.
-l	Adds the <code>ext2</code> labels to the output displayed.
-n	Mounts a file system without adding it to the <code>/etc/mtab</code> file.
-r	Mounts the file system read-only.
-w	Mounts the file system in write mode.
-L	Mounts the file system with the specified label.
-t FILESYSTEM_TYPE	Mounts the device as the specified file system type.

Unmounting file systems

The `umount` command is used to unmount file systems on Linux systems. This is often used to unmount a removable media device before it is ejected. No open files can exist on the file system when unmounting. File systems can be unmounted using either the device name or the mount directory. Several options can be used with the `umount` command; these are covered in Table 5-12.

Table 5-12
Options Used with umount

Option	Use
-V	Displays version information.
-h	Displays help information.
-v	Unmounts the file system verbosely.
-n	Unmounts without writing to the <code>/etc/mtab</code> file.
-r	If unmounting fails, attempts to remount in read-only mode.
-a	Unmounts all file systems listed in the <code>/etc/mtab</code> file.
-f	Forces unmounting of the file system.
-V	Displays version information.

Following is an example of the correct use of the `umount` command.

```
# umount /mnt/cdrom
```

This command will unmount the device currently mounted to `/mnt/cdrom` so that it can be ejected.

There are two files you need to know about when mounting and unmounting files: `/etc/fstab` and `/etc/mtab`.

Checking available file systems with `/etc/fstab`

The `/etc/fstab` file contains the file systems mounted when a Linux system starts. This file also contains file systems that are mounted manually. If a file system is listed in this file and the proper rights are set, it can be mounted simply by issuing the `mount` command and the directory, as in the following:

```
# mount /mnt/floppy
```

An example of the `/etc/fstab` file is shown below. As you can see from this file, device names are listed followed by the mount point. Next is the file system type and whether the file system is mounted automatically by the operating system or manually by the user. Umask information is also stored here. The umask information is used to set the default permissions on newly created files and directories. This example shows devices mounted using the `supermount` option available as an option in some Linux kernels. `Supermount` is designed to prevent the manual mounting and unmounting currently required when changing removable media such as floppy and CD-ROM disks. When working with the `mount` commands and

storage media, remember that the system treats fixed and removable devices equally. When a device has been mounted it is available for use until it has been unmounted or removed.

```
/dev/hda8 / ext2 defaults 1 1
/dev/hda5 /boot ext2 defaults 1 2
none /dev/pts devpts mode=0620 0 0
/dev/hdb1 /fun ext2 defaults 1 2
/mnt/cdrom /mnt/cdrom supermount fs=iso9660,dev=/dev/cdrom 0 0
/mnt/cdrom2 /mnt/cdrom2 supermount fs=iso9660,dev=/dev/cdrom2 0 0
/mnt/floppy /mnt/floppy supermount fs=vfat,dev=/dev/fd0 0 0
/dev/hda1 /mnt/win_c vfat user,exec,umask=0 0 0
/dev/hda7 /mnt/win_d vfat user,exec,umask=0 0 0
none /proc proc defaults 0 0
/dev/hda9 swap_upgrade swap defaults 0 0
```

For example, the line `/dev/hda8 / ext2 defaults 1 1` indicates that partition 8 of hard drive A is mounted in the root (`/`) directory. The file system is type `ext2`, and the default options are used for this device. Some of the options available for use are `noauto`, which specifies that the device should not be automatically mounted, and `user`, which specifies that the devices are user mountable. The file `0 0` field is used to determine which devices need to be dumped. A value of `0` here specifies that no dump should be performed.

The `user` option here is very important and careful attention should be given to this option. It specifies that a file system is user mountable. This privilege is useful for devices such as removable media devices, allowing users to mount and unmount devices so that the media can be changed. Because this privilege allows users to mount and unmount devices it should not be granted to key file systems.

Checking mounted file systems with `/etc/mtab`

The `/etc/mtab` file contains a listing of the file systems currently mounted by the system. Below is an example of the `/etc/mtab` file. As you can see the device name is listed along with the mount point. The fields in this file correspond with the fields used in the `/etc/fstab` file. The file system type and access is also displayed here.

```
/dev/hda8 / reiserfs rw 0 0
proc /proc proc rw 0 0
/dev/hda5 /boot ext2 rw 0 0
devpts /dev/pts devpts rw 0 0
/dev/hdb1 /fun ext2 rw,noexec,nosuid,nodev 0 0
0
/mnt/cdrom2 /mnt/cdrom2 supermount fs=iso9660,dev=/dev/cdrom2 0
0
/mnt/floppy /mnt/floppy supermount fs=vfat,dev=/dev/fd0 0 0
/dev/hda1 /mnt/win_c vfat user,exec,umask=0 0 0
/dev/hda7 /mnt/win_d vfat user,exec,umask=0 0 0
none /proc proc defaults 0 0
/dev/hda9 swap_upgrade swap defaults 0 0
```

Key Point Summary

Managing the file system is an essential component to administering and using a Linux system. The topics covered in this chapter range from those used daily to those used only when adding a new file system. It is important to know all of the utilities, files, and concepts covered here.

- ♦ There are a number of file systems used by Linux systems; these include MS-DOS, ext2, iso9660, Minix, and NFS.
- ♦ Inodes contain the physical location of data contained in files. These determine the number of files that can be stored on a file system.
- ♦ Superblocks contain the inode table as well as information regarding the blocks and clusters used by the file system.
- ♦ Primary partitions are bootable by the system. A drive cannot contain more than four primary partitions.
- ♦ Each hard disk can contain one extended partition that acts as a container for an unlimited number of logical partitions. These logical and extended partitions are not bootable.
- ♦ Swap partitions are used by Linux systems to provide virtual memory for the system.
- ♦ The `fdisk` utility is used to view and modify partitions on a hard drive.
- ♦ The `fips` utility is a MS-DOS utility used to resize partitions.
- ♦ The `mkfs` utility is used to format partitions in Linux with the specified file system.
- ♦ The `fsck` utility is a file system consistency checker that can verify and correct some problems on Linux file systems.
- ♦ The `du` utility is used to view disk space information for directories including information on file and subdirectory space usage.
- ♦ The `df` utility is used to view disk space information for partitions. This utility also provides information on inode usage.
- ♦ The `mount` command is used to mount file systems.
- ♦ The `umount` command is used to unmount file systems.
- ♦ The `/etc/fstab` file contains information on file systems that are mounted automatically and manually.
- ♦ The `/etc/mtab` file contains information on file systems that are currently mounted.



STUDY GUIDE

The following questions and exercises will allow you to review the information covered in this chapter. Take your time when completing this section of the chapter, carefully reviewing any questions that you may have. Simply answering the question correctly is not as important as understanding the answer, so review any material that you might still be unsure of. Being comfortable with the questions and answers presented here will help you be more prepared for the certification exam questions.

Assessment Questions

1. Which type of file system is used for accessing remote systems?
 - A. ext2
 - B. hpfs
 - C. swap
 - D. nfs
2. Which of the following would represent the third partition on the second IDE drive?
 - A. /dev/hdb3
 - B. /dev/sdc2
 - C. /dev/hdc2
 - D. /dev/hda5
3. Which of the following is the pointer used to identify the location of data in files?
 - A. Superblock
 - B. Inode
 - C. Partition
 - D. File system

4. Logical drives are contained within what type of partition?
 - A. Primary
 - B. Extended
 - C. Swap
 - D. Root

5. Which command within `fdisk` is used to create a partition?
 - A. `c`
 - B. `d`
 - C. `n`
 - D. `p`

6. The _____ command is used to format a swap partition.

7. Which option, used with `e2fsck`, specifies an alternate superblock to use when checking the file system?
 - A. `-A`
 - B. `-b`
 - C. `-C`
 - D. `-l`

8. The _____ command displays the disk space utilized on mounted file systems.

9. Which of the following files contains information on currently mounted file systems?
 - A. `/etc/mtab`
 - B. `/etc/fstab`
 - C. `/proc`
 - D. `/dev`

10. The `fsck` utility checks a file system using how many passes?
 - A. Three
 - B. Four
 - C. Five
 - D. Six

11. Which command is used to view the disk usage within a directory?
 - A. df
 - B. du
 - C. mkfs
 - D. fsck

12. The _____ contains the inode table along with block and cluster information for the file system.

13. Which command is used to make a file system available to the system?
 - A. fsck
 - B. mount
 - C. fdisk
 - D. mkfs

14. User mountable file systems are specified in the _____ file.

15. Which type of file system is used for Linux systems?
 - A. ext2
 - B. hpfs
 - C. swap
 - D. nfs

16. Which type of file system is used to provide virtual memory for Linux systems?
 - A. ext2
 - B. hpfs
 - C. swap
 - D. nfs

17. Which type of file system is used for OS/2 systems?
 - A. ext2
 - B. hpfs
 - C. swap
 - D. nfs

18. How many primary and extended partitions are allowed on a hard disk?
- A. 1
 - B. 2
 - C. 3
 - D. 4
19. How many primary partitions are allowed on a hard disk?
- A. 1
 - B. 2
 - C. 3
 - D. 4
20. Before a file system can be examined using fsck it must first be _____.

Scenarios

1. There was recently a power loss that caused your computer to be shut down improperly. When you rebooted, the `fsck` utility was run automatically. Now the system works, but some of the files that were stored in your home directory are now missing. How would you locate these files?
2. There was recently a power loss that caused your computer to be shut down improperly. Now you get errors about a corrupt superblock on the drive. How would you recover from this problem?

Lab Exercises

Lab 5-1 Creating and using a file system

In this lab you will create a new partition and file system and configure it to be mounted when the system boots.

1. Use `fdisk` to create a new partition.

```
# fdisk /dev/hda
The number of cylinders for this disk is set to 2495.
There is nothing wrong with that, but this is larger than
1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of
LILO)
```

2) booting and partitioning software from other OSs
(e.g., DOS FDISK, OS/2 FDISK)

```
Command (m for help): n
Command action
  e   extended
  p   primary partition (1-4)
p
Partition number (1-4): 2
First cylinder (1-2495, default 1): 1500
Last cylinder or +size or +sizeM or +sizeK (1501-2495,
default 2495): 2495
Command (m for help): w
```

Here the `fdisk` utility is used with the device name of the first IDE hard disk, `/dev/hda`. The `n` command is used to create a new partition. The `p` command selects a primary partition, with the `2` used to specify the partition number of `2`. The beginning and ending cylinders are specified; then the changes are written to the disk as the utility is exited with the `w` command.

2. Create the ext2 file system on the partition created. To create an ext2 file system, you use the `mke2fs` command along with the device name of the partition, in this case `/dev/hda2`.

```
# mke2fs /dev/hda2
```

3. Check the file system using `fsck`. The ext2 file system is checked using the `fsck.ext2` command. The `-v` option makes the check verbose so that information is displayed as the command is executed. The device name, in this case `/dev/hda2`, is required for this command.

```
# fsck.ext2 -v /dev/hda2
e2fsck 1.18, 11-Nov-1999 for EXT2 FS 0.5b, 95/08/09
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information

    9692 inodes used (0%)
    157 non-contiguous inodes (1.6%)
        # of inodes with ind/dind/tind blocks: 1194/205/0
1754206 blocks used (70%)
    0 bad blocks

8884 regular files
 776 directories
   0 character device files
   0 block device files
   1 fifo
   0 links
  22 symbolic links (22 fast symbolic links)
   0 sockets
-----
9683 files
```

4. Add an entry for the file system to the `/etc/fstab` file. The `vi` editor is used to edit the `/etc/fstab` file. The `i` command is used to insert the line for the file system. The `ESC :wq` command is used to save and exit the editor.

```
# vi /etc/fstab
i
/dev/hda2 /fun ext2 defaults 1 2
<ESC>
:wq
```

5. Mount the file system manually. Once the data exists in the `/etc/fstab` file, the file system can be mounted using the `mount` command and the target location.

```
# mount /fun
```

6. View space information and inode information using the `df` utility.

```
# df
Filesystem            1k-blocks      Used Available Use%
Mounted on
/dev/hda8              4096380       1469176   2627204   36% /
/dev/hda5              15522         3710      11011    25% /boot
/dev/hda2              9740592        0    2384384   0% /fun
# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/hda8             3.9G  1.4G  2.5G  36% /
/dev/hda5             15M   3.6M   11M  25% /boot
/dev/hda2             9.3G   0G   9.3G   0% /fun
# df -i
Filesystem            Inodes   IUsed   IFree IUse% Mounted on
/dev/hda8            4294967295      0 4294967295   0% /
/dev/hda5             4016         27   3989    1% /boot
/dev/hda2            1237888     9692 1228196    1% /fun
```

The `df` utility is first shown here without options, displaying the disk usage information in blocks. The second use of this command uses the `-h` option to provide data in human-readable form. The last use of the command uses the `-i` option to provide inode information.

7. View the `/etc/mstab` file. Here the `cat` command is used to display the `/etc/mstab` file, which contains the currently mounted file system. In the `/etc/mstab` file you can view the entry for the newly created file system, in this case the `/dev/hda2` device.

```
# cat /etc/mstab
proc /proc proc rw 0 0
/dev/hda1 /boot ext2 rw 0 0
devpts /dev/pts devpts rw 0 0
/dev/hda2 /fun ext2 rw,noexec,nosuid,nodev 0 0
```

Answers to Chapter Questions

Chapter Pre-Test

1. The `/etc/mtab` file contains information on currently mounted file systems.
2. The `df -i` command displays inode information.
3. The `mount -a` command mounts all file systems contained within the `/etc/fstab` file.
4. The `ext2` file system is the default used for Linux partitions.
5. The `fdisk` utility is used to create partitions on Linux systems.
6. The `mkswap` utility is used to format a Linux swap partition.
7. The `/etc/fstab` file contains the file systems mounted automatically by the system.
8. Primary partitions are bootable.
9. The `/dev/hdb2` device name is used for the second partition of the second IDE drive.
10. Drive information is stored within the superblock of the system.

Assessment Questions

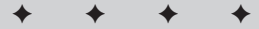
1. **D.** The `nfs` file system is used for access remote systems. See the “File system types” section for more information.
2. **A.** The third partition on the second drive is `/dev/hdb3`. The `b` specifies the second drive and the `3` is used for the third partition. See the “File system types” section for more information.
3. **B.** The inode is a pointer which identifies the location of data on the file system. See the “Considerations when making a file system” section for more information.
4. **B.** Logical drives exist within extended partitions. See the “Partition types” section for more information.
5. **C.** The `n` command is used to create a new partition in `fdisk`. See the “fdisk” section for more information.
6. **mkswap.** The `mkswap` utility is used to format swap partitions. See the “mkfs” section for more information.
7. **B.** The `-b` option is used to specify an alternate superblock with `fsck`. See the “fsck” section for more information.
8. **df.** The `df` command is used to display disk usage of file systems. See the “df” section for more information.

9. **A.** The `/etc/mtab` file contains the currently mounted file systems. See the “Checking mounted file systems with `/etc/mtab`” section for more information.
10. **C.** The `fsck` utility utilizes five passes to check the file system. See the “`fsck`” section for more information.
11. **A.** The `du` command is used to display disk usage of directories. See the “`du`” section for more information.
12. **superblock.** The superblock contains inodes and information on the file system including clusters. See the “Superblocks” section for more information.
13. **B.** The `mount` utility allows a file system to be accessed by the system. See the “Mounting file systems” section for more information.
14. **/etc/fstab.** The `/etc/fstab` file specifies which file systems are mountable by users. See the “Checking available file systems with `/etc/fstab`” section for more information.
15. **A.** The `ext2` file system is used by Linux systems. See the “File system types” section for more information.
16. **C.** Swap file systems are used to provide virtual memory in Linux systems. See the “File system types” section for more information.
17. **B.** The `hpfs` file system is used by OS/2 systems. See the “File system types” section for more information.
18. **D.** Only four total partitions of primary and extended type can exist on a drive. See the “Partition Types” section for more information.
19. **D.** Four primary partitions can exist on a single drive. See the “Partition Types” section for more information.
20. **unmounted.** A file system must be unmounted before it can be inspected with `fsck`. See the “`fsck`” section for more information.

Scenarios

1. The `/lost+found` directory should be checked for any files that are lost after a disk problem. This is where files that have been orphaned are placed during the `fsck` process.
2. As shown in the following example, the `-b` option is used with `e2fsck` to specify an alternate superblock to use when examining the file system. The superblock contains information such as inodes and disk clusters. An alternate superblock is located at every 8K blocks on the drive.

```
# e2fsck -b 8193
```

Managing Files

EXAM OBJECTIVES

Exam 101 ♦ General Linux, Part 1

1.3 GNU and Unix Commands

- **Perform Basic File Management.** Use the basic unix commands to copy and move files and directories. Perform advanced file management operations such as copying multiple files recursively and moving files that meet a wildcard pattern. Use simple and advanced wildcard specifications to refer to files.

2.4 Devices, Linux File Systems, Filesystem Hierarchy Standard

- **Set and view disk quota.** Setup disk quota for a filesystem, edit user quota, check user quota, generate reports of user quota. Includes quota, edquota, repquota, quotaon commands.
- **Use file permissions to control access to files.** Set permissions on files, directories, and special files, use special permission modes such as suid and sticky bit, use the group field to grant file access to workgroups, change default file creation mode. Includes chmod and umask commands. Requires understanding symbolic and numeric permissions.
- **Manage file ownership.** Change the owner or group for a file, control what group is assigned to new files created in a directory. Includes chown and chgrp commands.
- **Create and change hard and symbolic links.** Create hard and symbolic links, identify the hard links to a file, copy files by following or not following symbolic links, use hard and symbolic links for efficient system administration.

EXAM OBJECTIVES (CONTINUED)

- **Find system files and place files in the correct location.** Understand the filesystem hierarchy standard, know standard file locations, know the purpose of various system directories, find commands and files. Involves using the commands: find, locate, which, updatedb. Involves editing the file: /etc/updatedb.conf

CHAPTER PRE-TEST

1. Which command is used to search for files using a database?
2. Where is the system kernel located?
3. What tool is used to set default permissions?
4. Which type of file link can span file systems?
5. Quotas are enabled using which command?
6. Which command is used to list files in a directory?
7. Which command is used to create a directory?
8. What command can be used to create an empty file?
9. What command is used to search the `PATH` statement for a command?
10. Which command recursively searches the directory for a specified filename?

This chapter covers many of the various tools used for managing files. Basic commands used for creating, moving, and deleting files are covered. Also covered in this chapter are commands used for dealing with directories that contain files. A variety of tools used for locating and compressing files are explained in this chapter. You will also learn about the tools used for managing quotas and permissions, two areas where users and files are closely connected. Finally, you will learn how file links are used and where the standard location of many of the files on a Linux system is. File management is a major part of working with a Linux system, so thoroughly understanding the material covered in this chapter is important.

Managing Files

Objective

1.3 GNU and Unix Commands

- **Perform Basic File Management.** Use the basic unix commands to copy and move files and directories. Perform advanced file management operations such as copying multiple files recursively and moving files that meet a wildcard pattern. Use simple and advanced wildcard specifications to refer to files.

When working with a Linux system, you need to be familiar with basic commands used for file management. In this section we cover the most commonly used tasks, including moving around the file system and working with directories and files.



You will need to know the purpose of each command along with the commonly used options and functionality.

When working with files on a Linux system, you should keep in mind a few practices and restrictions. Hidden files begin with a period. Filenames can begin with a number, but they can't contain slashes, question marks, asterisks, or other reserved characters. File extensions aren't always required but can be useful for keeping track of file types.

The following sections examine some of the basic commands used when dealing with files and directories.

Changing directories

Directories on a Linux system are arranged in a tree structure. The / directory, known as the *root* directory, contains a number of system directories. Each of these directories can contain further directories and so on as the tree branches outward. Figure 6-1 shows this directory structure. The system directories and their locations are covered later in this chapter.

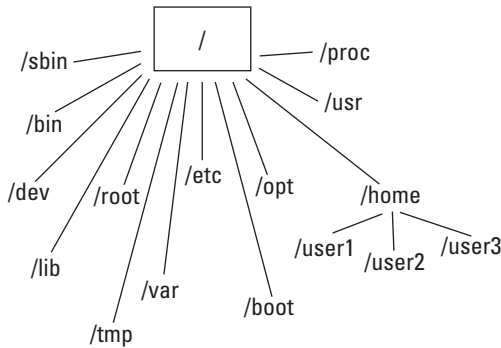


Figure 6-1: Basic Linux directory structure

The `cd` command is used to move through the directories at a Linux shell prompt; `cd` stands for *change directory*. This is one of the simplest commands used on a Linux system. The present working directory, or `pwd`, is the directory that you are in at any moment. When used alone, the `cd` command will change the user's working directory to their home directory. It is also possible to specify the directory you wish to make your working directory. The following is the syntax for the `cd` command.

```
cd /directory
```

The `cd` command can be used with a leading slash when specifying the *absolute directory path* starting at the root directory. This allows you to move to any location within the file system by specifying its absolute path. Without the leading slash, the system searches for the directory you specified within the `pwd`. Finding a directory within the `pwd` is known as a *relative path*. An example of using relative paths is shown in the following example. In this example the `pwd` command is used to view the present working directory.

```
# pwd
/home
# cd angie
# pwd
/home/angie
```

The `cd` command can also be used with shell symbols and environment variables. The `.` and `..` directories are hidden files that exist in every directory. The `.` represents the current directory while the `..` represents the parent directory. For the `/` directory the `..` file points to the `/` directory. Another shell symbol commonly used is the `~` character. This character is used to represent the user's home directory. Finally, the `-` character can be used with the `cd` command to return to the previous directory. These are all special characters used as shell symbols. Another special

character used at the shell prompt is the wildcard character (*). This character is used to present either no or multiple characters. The following command will display all files and directories beginning with the letter a, including any that are simply named a.

```
# ls a*
```

The following example uses shell symbols to move up one directory within the directory structure.

```
# cd ..
```



Notice the space between the `cd` command and the periods. This space is required on Linux systems unless an alias has been created so that `cd..` is translated as `cd ..`.

Following is an example of using an environment variable with the `cd` command. This will change the working directory to the directory specified in the `HOME` environment variable. The `$` symbol is used to specify an environment variable at the command line.

```
cd $HOME
```

No options are used with the `cd` command. It simply uses the argument given as the path that is to become the `pwd`. This simple command may well be the most frequently used of all commands on Linux systems.

Listing directory contents

Another commonly used command is the `ls` command, which is used to list the contents of a directory; `ls` is short for *list*. When the command is used without arguments or options, it simply lists all of the files and directories located within the `pwd` that aren't marked as hidden files. By default, the output is displayed in alphabetical order. For example:

```
# ls
abcnames          list1             mycommands
numberednames
alphanames        list2             mydoc             readmes
alphanicks        list_1            myfile.12.11.00
sortednameslist
alteredservices  list_2            myfiles           status
auto              longerfile.gz    mynewcommands    stuff
doomnicks         longerfile2.gz   names             test
fivenamesaa       longerfile3.gz   nameslist         test2
fivenamesab       longerfile~.gz   nameslisterrors  xaa
fivenamesac       marital          new               xab
infodoc           mergedalpha      newtest           xac
```

The `ls` command can accept paths as arguments and will provide a listing of the files and directories located within the path. You can use absolute paths, like this:

```
# ls /home/angie/stuff
alphanames list nameslist nicks readmes
```

Alternatively, you can use a relative path, like this:

```
# ls stuff
alphanames list nameslist nicks readmes
```

The argument can also contain strings and wildcards, in which case a listing of all files and directories matching the string are displayed, like this:

```
# ls a*
abc123names abcnames alphanames alteredservices
```

When paths are used in conjunction with wildcards, the path is shown along with the results:

```
# ls /home/angie/stuff/n*
/home/angie/stuff/nameslist /home/angie/stuff/nicks
# ls stuff/n*
stuff/nameslist stuff/nicks
```

The `ls` command can accept options along with arguments. A large number of options are available for this command to allow for a large degree of control on the output produced. In the tables that follow we have arranged the options according to use for ease of reference. Table 6-1 contains the options used to configure the file listings displayed with the `ls` command.

Table 6-1
File Listing Options Used with `ls`

Option	Use
-a	Lists all contents of the directory, including those beginning with a period.
-A	Works like -a except . and .. are not listed.
-B	Files ending with ~ are not displayed.
-d	Displays the directory name instead of the contents.
-L	Displays file information for the file referenced in a link instead of information for the link.
-R	Displays contents of the directories recursively so that the contents of child directories are also displayed.

A variety of options can be used to specify the information displayed with the file listings. Table 6-2 covers the information listing options used with the `ls` command.

Table 6-2
Information Listing Options Used with `ls`

<i>Option</i>	<i>Use</i>
<code>-G</code>	Specifies that group information is not to be displayed.
<code>-i</code>	Displays the inode number.
<code>-l</code>	Displays file type, permissions, hard link count, file owner, group owner, and modification time.
<code>-o</code>	Displays the same information as the <code>-l</code> option except group information is excluded.
<code>-s</code>	Displays the file size in 1024-byte blocks.

The `ls` command also uses options to order the output that is displayed. These options are shown in Table 6-3.

Table 6-3
Output Ordering Options Used with `ls`

<i>Option</i>	<i>Use</i>
<code>-c</code>	Displays output according to the status change time or <code>ctime</code> of the inode.
<code>-f</code>	Displays output in the order which they were saved in the directory.
<code>-r</code>	Displays output in reverse order.
<code>-S</code>	Displays files according to size from largest to smallest.
<code>-t</code>	Displays files by modification time beginning with most recent.
<code>-u</code>	Displays files by access time beginning with most recent.

The output produced with the `ls` command can also be controlled with another set of options. The options shown in Table 6-4 control the appearance of the output produced.

Table 6-4
Output Appearance Options Used with ls

<i>Option</i>	<i>Use</i>
-l	Output is displayed one file per line.
-C	Output is displayed in columns.
-F	Displays filenames with an appended character to specify file type.
-k	Displays file size in kilobytes.
-m	Displays filenames horizontally separated by commas.
-n	Displays the user name and group id numbers.
-p	Displays filenames with an appended character to indicate type.
-x	Displays filenames in horizontally ordered columns.
-T COLS	Displays filenames with tab stops set COLS columns wide.
-w COLS	Displays filenames with screen width set COLS columns wide.

These options can be combined to create very specific output using the `ls` command. The following are some examples of how these options can be used to control what data is displayed along with how it is displayed. Refer to the preceding tables to ensure that you understand the reasons each of these commands produces the resulting output.



You can be sure that the `ls` command used with options and arguments will appear on the exam. Be sure you understand how to use these elements together to produce a variety of output.

The following example lists all contents of the directory with details, one file per line:

```
# ls -al
drwx----- 3 angie  angie      4096 Dec 10 23:11 ..
-rw----- 1 angie  angie     11087 Dec 10 23:16 .bash_history
-rw-r--r-- 1 angie  angie      24 Dec 10 23:16 .bash_logout
-rw-r--r-- 1 angie  angie     230 Dec 10 23:16 .bash_profile
-rw-r--r-- 1 angie  angie     124 Dec 10 23:16 .bashrc
-rwxr-xr-x 1 angie  angie     333 Dec 10 23:16 .emacs
-rw----- 1 angie  angie    12288 Dec 10 23:16 .namelist.swp
-rw-rw-r-- 1 angie  angie      62 Dec 10 23:16 .plan
-rw-r--r-- 1 angie  angie    3394 Dec 10 23:16 .screenrc
-rw-rw-r-- 1 angie  angie     320 Dec 10 23:17 alphanames
-rw-rw-r-- 1 angie  angie      87 Dec 10 23:18 list
-rw-rw-r-- 1 angie  angie     203 Dec 10 23:17 nameslist
-rw-rw-r-- 1 angie  angie     117 Dec 10 23:18 nicks
-rw-rw-r-- 1 angie  angie   2288432 Dec 10 23:18 readmes
```

The following example lists filenames according to access time, with an appended character / showing the directories.

```
# ls -pu
morestuff/ nicks list readmes alphanames nameslist
```

The following example lists the contents of the /home/angie/stuff directory and all its subdirectories.

```
# ls -R /home/angie/stuff
alphanames list morestuff nameslist newstuff nicks oldstuff readmes

/home/angie/stuff/morestuff:

/home/angie/stuff/newstuff:
newnameslist newnicks

/home/angie/stuff/oldstuff:
oldnameslist oldnicks
```

As you can see in these examples, the `ls` command can be a powerful tool for gathering information about files and directories. This command is often used in conjunction with other commands and files using pipes and redirection. This allows output to be saved to a file or processed by other utilities. Understanding the use of the `ls` command alone and with other utilities will prepare you for the exam and for the job of working with Linux systems.

Determining a file type

The `ls` command provides a high degree of functionality when examining files. However, it provides limited information about the contents of the file. The `file` command can be used to learn more about the contents of files on a Linux system. The output of the `file` command includes one of the following words: text, executable, data, or directory. This command accepts arguments to specify which files to examine. A variety of options can be used with this command; these are shown in Table 6-5.

Table 6-5
Options Used with file

<i>Option</i>	<i>Use</i>
-b	Specifies that filenames are not to be included with the output.
-f <FILENAME>	Specifies that the <FILENAME> file contains the names of files that are to be examined by the <code>file</code> command.

Continued

Table 6-5 (continued)

Option	Use
-n	Flushes <code>stdout</code> after checking a file. This can be useful when working with a list of files that are being sent to another command.
-v	Displays version information and exits.
-z	Attempts to examine the contents of compressed files.

Some example uses of the `file` command and its options follow. In the first example, the `*` wildcard is used, so the command examines all the files in the `pwd`.

```
# file *
alphanames: ASCII text
list:       ASCII text
morestuff:  directory
nameslist:  ASCII text
newstuff:   directory
nicks:      ASCII text
oldstuff:   directory
readmes:    English text
```

In the following example, the filename isn't included with the output:

```
# file -b /etc/lilo.conf
ASCII text
```

In the following example, the `file` command is used to examine the contents of a compressed file:

```
# file -z /usr/info/tar.info.gz
/usr/info/tar.info.gz: English text (gzip compressed data,
deflated, last modified: Wed Dec 31 19:00:00 1969, max
compression, os: Unix)
```

Changing file time stamp

The `touch` command allows you to change the time stamp on a file. If a filename is specified, but that file doesn't exist, an empty file is created with that name. The options available for use with the `touch` command are covered in Table 6-6.

Table 6-6
Options Used with touch

Option	Use
-a	Only changes the access time on the file.
-c	Do not create a file if none exists.
-d <STRING>	Used to specify the <STRING> stamp on the file instead of using the current time.
r <file>	Uses the time of the <file> instead of the current time.
-t <TIME>	Uses an argument to specify the time instead of using the current time.

As you can see, the `touch` command allows you to use the current time or to specify a time for the time stamp. The command is often used to create an empty file with the specified name. This can be useful when running other scripts that expect a file to exist. Log files are one example of a type of file that is automatically written to and sometimes must exist to avoid errors in logging. If the file is archived elsewhere you might want to delete the original file. You can then use `touch` to create an empty file used to store the log information. This command is often run in scripts as `cron` jobs at specified times. The `touch` command allows this process to run automatically with no user intervention and no error messages generated due to a missing file.

The following example shows a script in which the `touch` command is used to create an empty file.

```
cp mylogs stuff/mylogs.`date +%m.%d.%y`  
rm -f mylogs  
touch mylogs
```

This will copy the contents of the `mylogs` file to a file located in the `stuff` directory with a filename of `mylogs.month.day.year`, (for example, `mylogs.12.11.00`). The original `mylogs` file is deleted, and then the `touch` command is used to create an empty file named `mylogs`. This is one common use of the `touch` command.

Copying files

When working with files on any system, you often need to copy files. Linux includes two commands for file copying. The `cp` command is used to copy files and directories. This is the standard command used when copying files from one location on the system to another. When copying files from one file format to another the `dd` command is used.

cp

The `cp` command (short for *copy*) is used for standard file copies on Linux systems. This command is used to create a new, independent copy of the original file or directory. Several options are used with the `cp` command to customize the copies created. These options are covered below in Table 6-7.

Table 6-7
Options Used with `cp`

Option	Use
-a	Specifies that links and attributes of the original files are to be transferred to the new copy.
-d	Specifies that links are to be preserved when copied.
-f	Overwrites any existing destination files.
-i	Prompts before overwriting any existing destination files.
-l	Specifies that hard links, which are discussed later in the chapter, are to be created instead of copies of files.
-p	Preserves the original file's owner, group, permissions, and time stamps.
-r	Copies directories and contents recursively while copying all files as standard files. This option shouldn't be used with special files.
-R	Copies directories and contents recursively, preserving nondirectories.
-s	Creates symbolic links, covered later in the chapter, of nondirectory files.
-v	Displays the names of all files as they are being copied.

These options can be used in combination when copying files. Along with these options, arguments are used with the `cp` command. The correct syntax for the `cp` command is as follows:

```
cp -option source target
```

When the target specified is a directory, the source file is copied to that directory with the same name as the original file. When the target specified is not a directory, the original file is copied to the specified location with the target name. The following are some examples of the use of the `cp` command. In the first example the file `marital` is copied to the directory `stuff`.

```
# cp marital stuff
```

In the second example the directory `/home/angie/stuff` and its contents are copied to the directory `/home/angie/otherstuff`. This is particularly useful as it allows the entire directory and its contents to be copied using one command.

```
# cp -r /home/angie/stuff /home/angie/otherstuff
```



This is an important option that you are likely to see as a test question.

In the third example the files and directories located in the `/home/angie/stuff` directory are copied verbatim to the `/home/angie/otherstuff` directory.

```
# cp -rv /home/angie/stuff/* /home/angie/morestuff
/home/angie/stuff/abcnames -> /home/angie/morestuff/abcnames
/home/angie/stuff/alphanames ->
/home/angie/morestuff/alphanames
/home/angie/stuff/list -> /home/angie/morestuff/list
/home/angie/stuff/marital -> /home/angie/morestuff/marital
/home/angie/stuff/morestuff -> /home/angie/morestuff/morestuff
/home/angie/stuff/mycommands.12.11.00 ->
/home/angie/morestuff/mycommands.12.11.00
/home/angie/stuff/nameslist -> /home/angie/morestuff/nameslist
/home/angie/stuff/newstuff -> /home/angie/morestuff/newstuff
/home/angie/stuff/newstuff/newnameslist ->
/home/angie/morestuff/newstuff/newnameslist
/home/angie/stuff/newstuff/newnicks ->
/home/angie/morestuff/newstuff/newnicks
/home/angie/stuff/nicks -> /home/angie/morestuff/nicks
/home/angie/stuff/oldstuff -> /home/angie/morestuff/oldstuff
/home/angie/stuff/oldstuff/oldnameslist ->
/home/angie/morestuff/oldstuff/oldnameslist
/home/angie/stuff/oldstuff/oldnicks ->
/home/angie/morestuff/oldstuff/oldnicks
/home/angie/stuff/readmes -> /home/angie/morestuff/readmes
```

dd

The `dd` command (short for direct dump) is used to copy and convert files to a different file type simultaneously. This command has different options and a different syntax than the `cp` command. The syntax used by the `dd` command is as follows:

```
dd [options]
```

The `dd` command, by default, writes data from standard input to standard output. Options can be used to overwrite these defaults. The options used with the `dd` command are shown in Table 6-8.

Table 6-8
Options Used with dd

<i>Option</i>	<i>Use</i>
<code>if=FILE</code>	Specifies the location of a source to use instead of standard input. This is the input file.
<code>of=FILE</code>	Specifies the location of the destination to use instead of standard output. This is the output file.
<code>ibs=BYTES</code>	Specifies the number of <i>BYTES</i> read at a time.
<code>obs=BYTES</code>	Specifies the number of <i>BYTES</i> written at a time.
<code>bs=BYTES</code>	Specifies the number of <i>BYTES</i> to read and write at a time.
<code>cbs=BYTES</code>	Specifies the number of <i>BYTES</i> to convert at a time.
<code>skip=BLOCKS</code>	Specifies the <i>BLOCKS</i> to skip in the input file before copying.
<code>seek=BLOCKS</code>	Specifies the <i>BLOCKS</i> to skip in the output file before writing.
<code>count=BLOCKS</code>	Specifies the <i>BLOCKS</i> of the input file to copy instead of copying the entire file.

The `dd` command can be used for a variety of special tasks. For example, suppose the system you currently use is running both Windows NT and Windows 98. The first partition contains Windows 98 system files and is formatted with FAT. The second partition contains Windows NT system files and is formatted with NTFS. A third partition, formatted as FAT, contains data that is shared between the two operating systems. This drive also contains 3GB of space that you wish to use for Linux. Because this computer is used by your young children, you prefer to use the NT boot loader that they are familiar with using. This can be done by installing the Linux boot sector along with LILO to a location that doesn't overwrite the master boot record; in this case you install the Linux boot partition to `/dev/hda5`. While in Linux, mount the Windows 98 system partition so that you can write to it. Then run the following command:

```
# /bin/dd if=/dev/hda5 bs=512 count=1 of=/mnt/win_c/bootsek.lin
```

This will create a file named `bootsek.lin` on the Windows 98 system partition. The file will be written as one block with a size of 512 bytes. You will then need to boot to either NT or Windows 98 and edit the `boot.ini` file to include the following line:

```
c:\bootsek.lin="Linux"
```

This will add a line to the NT boot loader for Linux. When this line is selected during boot, the system will then boot to the LILO boot loader. Any time the `lilo` command is run on the Linux system the `bootsek.lin` file will need to be rewritten.

In this example, the `dd` command is used to copy a file from the Linux boot partition to another partition formatted using the FAT file system. This command is also useful when converting files from ASCII to EBCDIC, the Extended Binary Coded Decimal Interchange Code used by IBM, such as when working with tape backup systems.

Moving files

It is possible to move a file manually by copying it to the new location and erasing the original file. However, Linux includes a command for moving files that automates these tasks. The `mv` command (short for *move*) allows you to move and rename files on Linux systems. This command works much like the `cp` command covered earlier in the chapter, using the same command syntax. The options used with the `mv` command are a bit different and are covered in Table 6-9.

Table 6-9
Options Used with `mv`

Option	Use
<code>-f</code>	Removes existing files without prompting.
<code>-i</code>	Prompts the user before overwriting files.
<code>-u</code>	Specifies that files are not moved to a destination with the same or newer modification times.
<code>-v</code>	Verbosely prints the names of files as they are moved.

In the following example, all filenames matching the pattern beginning with `file` are moved verbosely to the `myfiles` directory.

```
# mv -v file* myfiles
fileaa -> myfiles/fileaa
fileab -> myfiles/fileab
fileac -> myfiles/fileac
filead -> myfiles/filead
fileae -> myfiles/fileae
filespace -> myfiles/filespace
```

In the second example, the directory `myfiles` is renamed to `files`.

```
# mv myfiles files
```

Deleting files

Another task often required when working with files and directories is removing them. The `rm` command (short for *remove*) is used to delete files and directories on Linux systems. The `rm` command uses the following syntax:

```
rm -options FILE
```

Several options can be used with the `rm` command. The most frequently used options are shown in Table 6-10.

Table 6-10
Options Used with `rm`

Option	Use
-d	Used by superuser. Removes directories regardless of whether they are empty.
-f	Processes filenames without prompting, even if they don't exist.
-i	Prompts the user when removing files.
-r	Removes directory contents recursively.
-v	Verbosely removes files.

In the following example all files in the `pwd` beginning with `nn` are deleted.

```
# rm nn*
```

The second command removes all files located in the `files` directory verbosely, along with the directory itself.

```
# rm -frv files
removing files/fileaa
removing files/fileab
removing files/fileac
removing files/filead
removing files/fileae
removing files/filespace
removing the directory itself: files
```

Creating directories

At this point we have covered the creation and removal of files as well as the removal of directories. Now we will examine the `mkdir` command (short for *make directory*) that is used to create directories. This is a very basic command that can create one directory layer at a time. When used with the `-p` option parent

directories can also be created as needed. In order to create the directories `/home/angie/ourfiles`, `/home/angie/ourfiles/myfiles`, `/home/angie/ourfiles/yourfiles` the following command can be used.

```
# mkdir /home/angie/ourfiles /home/angie/ourfiles/myfiles
/home/angie/ourfiles/yourfiles
```

In the above command the directory `/home/angie/ourfiles` is created first. The command then creates the directories `myfiles` and `yourfiles` within the `ourfiles` directory.

Understanding File System Hierarchy

Objective

2.4 Devices, Linux File Systems, Filesystem Hierarchy Standard

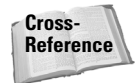
- **Find system files and place files in the correct location.** Understand the filesystem hierarchy standard, know standard file locations, know the purpose of various system directories, find commands and files. Involves using the commands: `find`, `locate`, `which`, `updatedb`. Involves editing the file: `/etc/updatedb.conf`

The Linux file system follows a tree-like structure. The root (`/`) contains primary directories. Each directory located off the root can contain other directories as it branches off. Being familiar with several key directories when working with Linux is important. In this section we examine many of those directories.

Standard file locations

The following are some standard file locations on Linux systems.

- ♦ `/etc` — Contains many of the script and configuration files used on the system.
- ♦ `/etc/skel` — Contains the files that are to be copied to each user's home directory.



The `/etc/skel` file is covered in greater detail in Chapter 10.

- ♦ `/usr` — Contains many subdirectories that store applications and source files used by the user and superuser accounts. Some of these include the following:
 - `/usr/bin` — Contains executables used by users. This should be included in the user's path statement.
 - `/usr/sbin` — Contains executables used by superuser.

- `/usr/local`—Contains applications that aren't part of the Linux operating system.
- `/usr/local/bin`—Contains software installed after the initial operating system installation. This should also be in the user's path.
- `/usr/local/sbin`—Contains administrative software installed after the initial operating system installation.
- ♦ `/var/log`—Contains log files.
- ♦ `/var/spool`—Contains mail and printing files.
- ♦ `/bin`—Contains binaries used during system startup.
- ♦ `/sbin`—Contains administrative binaries used by the superuser.

System directories

The following are a number of system directories with specific purposes on Linux systems:

- ♦ `/`—System root directory.
- ♦ `/root`—Home directory for superuser.
- ♦ `/home`—Contains user's home directories.
- ♦ `/boot`—Contains files used by the boot loader including the kernel.
- ♦ `/dev`—Contains peripheral access files.
- ♦ `/proc`—A virtual directory containing system information.

Locating Files

Objective

2.4 Devices, Linux File Systems, Filesystem Hierarchy Standard

- **Find system files and place files in the correct location.** Understand the filesystem hierarchy standard, know standard file locations, know the purpose of various system directories, find commands and files. Involves using the commands: `find`, `locate`, `which`, `updatedb`. Involves editing the file: `/etc/updatedb.conf`

Another capability that you will require on a Linux system is locating files. Luckily, Linux provides several methods for searching for files. The `find`, `locate`, `which`, and `whereis` commands all prove useful for this task.

find

The `find` utility is used to search for files. This utility begins searching at a specified directory and will then search all subdirectories contained within that directory for filenames matching the specified pattern. When no directory is specified, the `find` utility begins by searching the `pwd` and then searches all subdirectories located within that directory. The correct syntax for this utility is as follows:

```
find /path expression
```

Searches using the `find` utility can take a while and use a lot of system resources depending on the search being done. Table 6-11 shows some of the various conditions that can be searched for using `find`. Following is an example of the use of the `find` utility.

```
# find /home/angie -name stuff
/home/angie/stuff
# find /home/angie -user root
/home/angie/morestuff
```

Table 6-11
Conditions Used with `find`

<i>Option</i>	<i>Use</i>
<code>-atime</code>	Searches based on the number of days since last accessed.
<code>-ctime</code>	Searches based on the number of days since directory entry last changed.
<code>-group</code>	Searches for files belonging to the specified group.
<code>-newer</code>	Searches for files more recent than the specified file.
<code>-name</code>	Searches for files with names matching the specified string.
<code>-user</code>	Searches for files belonging to the specified user.

locate

The secure `locate` command provides a secure way to index and quickly search for files on your system. It uses incremental encoding to compress its database to make searching faster, but it will also store file permissions and ownership so that users will not see files they do not have access to. The `locate` command uses the `slocate` database, by default, to find file locations. It is possible to specify other database files to use. This command uses the following syntax:

```
locate -options arguments
```

The database is updated using the `updatedb` command. An example of the use of the `updatedb` command is shown below. Only the root user is authorized to use this command on the system.

```
# updatedb
```

The `locate` utility uses the options shown below in Table 6-12.

Table 6-12
Options Used with locate

<i>Option</i>	<i>Use</i>
-u	Start at / when creating the <code>slocate</code> database.
-U /PATH	Start at the specified path when creating the <code>slocate</code> database.
-e DIR	Exclude the specified directories when creating the <code>slocate</code> database.
-c	Parses the <code>/etc/updatedb.conf</code> file when creating the database.
-i	Searches without regard to case.
-o FILE	Specifies the output file to create.
-d PATH	Specifies the path of databases to search.

which

The `which` command takes one or more arguments. For each of its arguments it prints to `stdout` the full path of the executables that would have been executed when this argument is entered at the shell prompt. It does this by searching for an executable or script in the directories listed in the environment variable `PATH` using the same algorithm as `bash`.

This allows you to see the full path to a command before it is run. This can be useful for verifying that you are running the command that you meant to run. The utility uses the following syntax:

```
which -options programname
```

An example of the proper use of this command is shown in the example below to discover the location of the `locate` command.

```
# which locate  
/usr/bin/locate
```

whereis

The `whereis` command locates source/binary and manuals sections for specified files. The supplied names are first stripped of leading path name components and any (single) trailing extension of the form `.ext`, for example, `.c`. Prefixes of `s.` resulting from use of source code control are also dealt with. The `whereis` utility then attempts to locate the desired program in a list of hard coded locations. Table 6-13 covers the options used with the `whereis` utility.

Table 6-13
Options Used with `whereis`

<i>Option</i>	<i>Use</i>
<code>-b</code>	Searches for binaries.
<code>-m</code>	Searches for manual entries.
<code>-s</code>	Searches for sources.
<code>-u</code>	Searches for unusual entries that don't have one entry for each type.
<code>-B</code>	Changes where <code>whereis</code> searches for binaries.
<code>-M</code>	Changes where <code>whereis</code> searches for manual sections.
<code>-S</code>	Changes where <code>whereis</code> searches for source code.

Following is an example of the use of the `whereis` utility to locate information for the `ls` command:

```
# whereis ls
ls: /bin/ls /usr/man/man1/ls.1.gz
```

Creating File Links

Objective

2.4 Devices, Linux File Systems, Filesystem Hierarchy Standard

- **Create and change hard and symbolic links.** Create hard and symbolic links, identify the hard links to a file, copy files by following or not following symbolic links, use hard and symbolic links for efficient system administration.

When working with files sometimes you may wish to have one file in two locations. This can be useful for many reasons. You may wish to have a variety of permissions assigned to the file based on the location, or you may want to move files across the system without interrupting users. A program may expect to find a file in a specific location that isn't convenient. Links allow a single file to appear to exist in multiple locations. Links come in two varieties: hard links and soft links.

Hard links

A *hard link* maintains the same permissions and access times of the original file; in fact, both have the same inode number. The hard link simply appears to have a different filename and/or location. Changes to either the original file or the link affect both. The two files have equal importance. However, deleting either of the files won't affect the other. The hard link files appear to be regular files when examined and they function this way as well.

Hard links do have certain limitations, however. Normal users can't create hard links to directories. Also, hard links aren't allowed to span file systems.

To create a hard link, you use the `ln` command. An example of creating a hard link of the `stuff` file to `/home/angie/stuff` is as follows:

```
# ln stuff2 /home/angie/stuff
#ls -al
-rw-r--r--  2 root    root      89704 Aug 27  2000 stuff
-rw-r--r--  2 root    root      89704 Aug 27  2000 stuff2
```

As you can see from the above example these files are the same size. The number two that appears after the permissions indicates that two hard links of the file exist.

Symbolic links

The other type of link is a *symbolic* or *soft link*. These links allow a greater degree of flexibility. Users can create symbolic links to directories, and symbolic links can span file systems. Links can even be created to nonexistent files. The symbolic link maintains permissions separate from those of the original file. Deleting the original file won't remove the symbolic link, but the file itself will be removed. This can cause the symbolic link to no longer function.

To create a soft link, you use the `ln` command as displayed in the following example. Here a soft link of the `stuff` file is created at `/home/angie/stuff`.

```
# ln -s stuff /home/angie/stuff
```

Soft links can be identified by listing the details of a directory's contents. This is shown in the following example.

```
# ls -al
lrwxrwxrwx  1 angie   angie      7 Mar 10 17:06 stuff -
> /fun/stuff
```

It is important to understand that although opening, reading, and writing to a link affects the target file, removing the symbolic link removes only the link, not the target. When the symbolic link file is copied, the original file is placed in the target location. This is an independent copy of the original file and changes to this copy do not affect the original.

Working with Permissions

Objective

2.4 Devices, Linux File Systems, Filesystem Hierarchy Standard

- **Use file permissions to control access to files.** Set permissions on files, directories, and special files, use special permission modes such as `suid` and `sticky` bit, use the `group` field to grant file access to workgroups, change default file creation mode. Includes `chmod` and `umask` commands. Requires understanding symbolic and numeric permissions.
- **Manage file ownership.** Change the owner or group for a file, control what group is assigned to new files created in a directory. Includes `chown` and `chgrp` commands.

When you're working with files and directories, permissions are a very important detail to consider. Each user account is associated with a *userid* and a *groupid*. Users can also change to other groups using the `newgrp` command. Permissions are assigned to files and directories according to these accounts. In this section we discuss the basics of permissions on Linux systems.

Symbolic and numeric permissions

Permissions can be expressed symbolically or numerically. The most common permissions used with files and directories are *read*, *write*, and *execute*. These can be expressed using the letters *r*, *w*, and *x*, or in binary form 4, 2, and 1. Binary digits double in value when moving from right to left; therefore, read has a value of 4, write has a value of 2, and execute has a value of 1. It is very important to know the symbolic and numeric representation of each of these permissions.

These permissions can be combined. The permissions of read and write would appear as *rw*- in symbolic format, which adds up to 6 in numeric format. The dash in the third space indicates that the execute permission is not present. The permissions of read, write, and execute appear as *rwx*, 7 in numeric format. Be sure you understand how to combine these permissions using both formats.

Files, directories, and special files

These permissions are applied to files, directories, and special files. The left-most bit in a permissions listing is used to specify the file type. Standard files do not have a value on this bit, and that place is represented with a dash. Directories are represented by a *d*, links are represented with an *l*, and character files are presented with a *c*. The following is an example of permissions shown on a directory listing.

```
# ls -l
-rwxrwxr-- 1 angie  angie          82 Dec 11 21:48 auto
-rw-rw-r-- 1 angie  angie        1030 Dec  9 16:14 infodoc
-rw-rw-r-- 1 angie  angie          103 Dec  8 23:54 list.gz
```

```

-rw-rw-r-- 1 angie  angie  478 Dec  7 16:02 longerfile.gz
-rw-rw-r-- 1 angie  angie  491 Dec  7 15:58 longerfile2.gz
-rw-rw-r-- 1 angie  angie  489 Dec  7 15:59 longerfile3.gz
-rw-rw-r-- 1 angie  angie  480 Dec  7 15:49 longerfile~.gz
-rw-rw-r-- 1 angie  angie  234 Dec  9 15:26 marital
-rw-rw-r-- 1 angie  angie    0 Dec 11 21:49 mycommands
-rw-rw-r-- 1 angie  angie    0 Dec 11 21:45 myfile.12.11.00
drwxrwxr-x 4 angie  angie 4096 Dec 13 00:20 myfiles
-rw-rw-r-- 1 angie  angie   55 Dec  9 16:44 mynewcommands
-rwxrwxr-x 1 angie  angie    0 Dec 13 00:40 new
-rw-rw-r-- 1 angie  angie  117 Dec  9 01:39 nicks
-rw-rw-r-- 1 angie  angie   308 Dec 10 00:36 numberednames
-rw-rw-r-- 1 angie  angie 2288432 Dec 10 01:18 readmes
lrwxrwxrwx 1 angie  angie    7 Dec 13 05:43 status -> marital
drwxrwxr-x 5 angie  angie 4096 Dec 12 17:53 stuff
-rw-rw-r-- 1 angie  angie  244 Dec  8 23:30 test
-rw-rw-r-- 1 angie  angie   72 Dec  7 15:35 test2

```

As you look through this example, it is easy to spot the directories and link files using this character. This is important because permissions have different meanings based on the file type. The read permission on directories allows the contents to be listed, whereas the read permission on a standard file allows the contents of the file to be viewed. The write permission on a directory allows you to create and delete files within the directory. This permission on a standard file allows you to alter the file. The execute right on a directory allows you to make that directory your `pwd`. On standard files, this right allows you to run the command. Clearly, knowing what type of file you are examining is important in order to determine what access is being granted to that file.

User and group permissions

Files and directories utilize three sets of permissions, all of which are shown on the left side of the file listing in the previous section. The left-most permissions are granted to the owner, the middle set of permissions is given to the group, and the right-most permissions are assigned to all other users. This allows file access to be restricted so that the access varies according to the user. In the example in the previous section, the file `infodoc` has read and write permissions for the owner and the group, but only read permission for other users.

You can use several tools to reassign permissions and ownership of files, giving you the capabilities to change the owner and the group and to edit the permissions directly.

chown

The `chown` command is used to change ownership of files and directories. This command is used with the following syntax:

```
chown -options owner.group file
```

The *owner* is left unchanged if the parameter isn't provided. The *group* is left unchanged if the parameter isn't provided, but changed to the login group, which is the default group membership assigned at login, if the parameter is represented by a period or a colon. You can use a variety of options with this command, several of which are explained in Table 6-14.

Table 6-14
Options Used with `chown`

<i>Option</i>	<i>Use</i>
-c	Displays when a change has been made.
-f	Silently makes the changes without displaying messages.
-R	Changes ownership on files and directories recursively.
-v	Verbosely displays information for all files processed.

The following are some examples of the proper use of the `chown` command. The first example recursively assigns ownership of the `stuff` directory to `angie`.

```
# chown -R angie stuff
```

In the second example the ownership of all files beginning with `longer` within the `pwd` is assigned to `angie`. This action is done verbosely, and `angie` retains ownership on all of the files.

```
# chown -v angie longer*
owner of longerfile.gz retained as angie
owner of longerfile2.gz retained as angie
owner of longerfile3.gz retained as angie
owner of longerfile~.gz retained as angie
```

chgrp

The `chgrp` command can also be used to change the group assigned to a file. This command searches the `/etc/group` file to verify that existence of the group specified before changing the permissions. Options used with this command are shown in Table 6-15. This command is used with the following syntax:

```
chgrp newgroup file
```


Table 6-15
Options Used with chgrp

<i>Option</i>	<i>Use</i>
-c	Verbosely reports when a change is made.
-h	Changes symbolic links without changing the referenced file.
-f	Suppresses error messages.
-R	Operates recursively throughout the directory.
-v	Verbosely displays information for all files processed.

chmod

The `chmod` command allows the highest degree of control over file permissions. This command can be used to change permissions assigned to owner, group, and other. Only the file owner and superuser are allowed to change the permissions for a file. This utility follows a similar syntax to the previous utilities.

```
chmod -options mode file
```

The permissions specified as the *mode* can use either symbolic or numeric values. The following example uses numeric values. As we discussed previously, permissions are assigned to user, group, and other using the binary numbers of 4 for read, 2 for execute, and 1 for write. So, to change the permissions on the file `mystuff` to allow the owner to read, write, and execute (7); the group to read and execute (6); and others to have only read access to a file (4), you would use the following command.

```
# chmod 764 mystuff
```



For the exam, be sure you are familiar with setting permissions using the numeric values; you are likely to see multiple questions requiring this skill.

You can also accomplish this task using symbolic values for permissions. However, several commands may be required to perform the same task. The following example shows one way to accomplish this task using the symbolic values.

```
# chmod u=rx mystuff
# chmod g=rx mystuff
# chmod o=r mystuff
```

When using symbolic values, as in the preceding example, *a* can be used to represent all users, *u* represents the owner of the file, *g* is for the group, and *o* is for other users. Permissions can be explicitly assigned as they are in the preceding example,

they can be added and subtracted, and multiple assignments can be made at once when they are the same for all users. In the following example assignments are made for both the owner and group allowing read, write, and execute:

```
# chmod ug=wrx mystuff
```

In the second example, the execute permission has been taken from other users on the `mystuff` file.

```
# chmod o-x mystuff
```

In the third example, the read permission is given to other users on the `mystuff` file.

```
# chmod o+r mystuff
```

You can use a variety of options with the `chmod` command to control the output produced. These options are shown in Table 6-16.

Table 6-16
Options Used with `chmod`

Option	Use
-c	Verbosely reports when a change is made.
-h	Changes symbolic links without changing the referenced file.
-f	Suppresses error messages.
-R	Operates recursively throughout the directory.
-v	Verbosely displays information for all files processed.

umask

The `umask` command can be used to set default permissions that are assigned to newly created files. The `umask` command works using the numeric values for file permissions. This value is often assigned through a profile script and can be viewed by typing the `umask` command. The `/etc/profile` file contains the `umask` setting, and this is the file used to change the system-wide `umask` setting.

```
# umask
002
```

The `umask` value works to filter permissions on newly created files. The value shown here would block the write permission to other users. It is possible to change this value using the `umask` command; however, the next time you log onto the system the old value will be restored. Adding these values to your profile file

will allow these values to be assigned at each login. These values are actually the inverse of permissions. Rights shown in the `umask` are the rights filtered in the permissions assigned to files.



Be sure to understand what effect the `umask` has on file permissions. There will be questions concerning this material on the exam.

SUID and SGID

When using permissions, you can configure a file so that any user executing the file has permissions of either the owner or the group. This can be risky because it means that a user is allowed to run a command as another user. However, there may be times when you need this ability. The SUID, set user id, and SGID, set group id, bits can be set using the `chmod` command. In the following examples, these bits are set using the symbolic values. The SUID bit is applied to the owner permission in the first example, while in the second example the SGID bit is applied to the group permission.

```
# chmod o+s mystuff
# chmod g+s mystuff
```

These values can also be set using numeric values with the `chmod` command. These special values have the following binary values: SUID 4, SGID 2, and sticky bit 1. The following examples illustrate how to set SUID and SGID with numeric values. The `mystuff` file has the owner permission of 7 (rwx) and the user and other permissions of 5 (rw). In the first example the SUID value is assigned while in the second example the SGID value is assigned.

```
# chmod 4755 mystuff
# chmod 2755 mystuff
```



Be careful when using these permissions. Be sure you know what the files do and what the permissions of the owner are.

Sticky bit

Another special bit that can be set is the *sticky bit*. The sticky bit is often used with shared directories to allow for greater ease of management. This bit can allow all users to write to the directory but allow only a file owner to delete a file within the directory. This allows many users to share files within the directory and prevents someone from overwriting or deleting files that don't belong to that user. The sticky bit uses the numeric value of 1 and the alphabetic value of *t*. This value can also be set using the `chmod` command as shown in the following two examples. In each of these examples the sticky bit is being set on permissions with the first example using alphabetic value *t* and the second example using numeric value 1.

```
# chmod u+t mystuffdir
# chmod 1777 mystuffdir
```

Using Compression Tools

At time, when working with files, you may need to combine or compress files, or to work with those that have been. The compression tools are also discussed in Chapter 11. Combining multiple files into one allows for easier management and downloading of files while compressed allows for the files to use less disk space.

The tools covered in this section include the `tar` command, which is used to group files together, along with the `gzip`, `gunzip`, `compress`, and `bzip2` commands, which are used to compress and uncompress files.

tar

The `tar` utility includes a lot of flexibility when dealing with combined and compressed files. This utility archives files by combining several files into one, called a *tarball*, and can also compress the files simultaneously. The `tar` utility uses the following standard command syntax:

```
tar -options FILENAME DIRECTORY
```

A number of options allow you control over the `tar` utility's functions. Table 6-17 covers the most frequently used options.

Table 6-17
Options Used with tar

Option	Use
-c	Creates a new archive file.
-d	Displays difference, if any, between the archived and unarchived files.
--delete	Removes a file from the archive.
-r	Appends files to the end of an archive.
-t	Displays the contents of a archive.
-u	Specifies that only files newer than those in the archive are appended.
-x	Extracts files from an archive.
-C	Changes to the specified directory.
-f	Specifies the file or device used for output.
-L	Specifies the tape length; the user is prompted to change tape after the specified number of kilobytes.

Continued

Table 6-17 (continued)

Option	Use
-0	Extracts and sends data to standard output.
-p	Extracts and retains permissions on files.
-P	Utilizes absolute paths for filenames.
-v	Verbosely lists filenames.
-W	Specifies that the files are verified after archival.
--exclude	Specifies a file to exclude from archival.
-X <FILE>	Specifies the <FILE>, which contains the files to exclude from archival.
-Z	Specifies that the <code>compress</code> utility be used to either compress or uncompress the archive.
-z	Specifies that the <code>gzip</code> utility be used to either compress or uncompress the archive.

The `tar` utility is widely used by all Linux distributions. Along with being a useful tool for backups, tarball files are also a convenient way to distribute files. They allow a single file download and because compression can be handled with `tar` they also create smaller files for download. Tarball files do not have to end with the `.tar` extension, but it is a good practice to include that extension when creating tarballs. Files that are joined and compressed often use the `.tar.gz` or `.tgz` extensions. Such naming conventions allow for quick identification of archive files.



The `tar` utility is particularly useful because of its ability to combine and compress files in one action. Knowing the options used to complete both tasks will prove especially useful as you work with Linux files.

gzip and gunzip

The `gzip` utility provides file compression and decompression on Linux systems. This utility preserves the permissions and modification times for files. When creating a compressed file the `gzip` utility replaces the original file with a compressed file of the same name and attaches the `.gz` extension. The `gunzip` utility is used to uncompress these files. Several different extensions can be used to denote `gzip` compressed files, the `.tgz` and `tar.gz` extensions can be used for tarball files that are compressed with `gzip`. Several options can be used to control the `gzip` and `gunzip` utilities; these options are explained in Table 6-18.

Table 6-18
Options Used with gzip and gunzip

<i>Option</i>	<i>Use</i>
-c	Sends output to <code>stdout</code> and doesn't overwrite the original file.
-d	Decompresses files.
-f	Forces the specified actions to occur without prompting.
-h	Displays help information and then exits.
-l	Displays compression information for files.
-q	Compresses files without displaying warning messages.
-r	Recursively compresses directories.
-v	Verbosely displays compression information.

The following is an example of how the `gzip` utility can be used. In this example all of the matching files are compressed and renamed to include the `.gz` extension.

```
# gzip -v longer*
longerfile:          51.1% -- replaced with longerfile.gz
longerfile2:        49.5% -- replaced with longerfile2.gz
longerfile3:        49.7% -- replaced with longerfile3.gz
longerfile~:        51.0% -- replaced with longerfile~.gz
```

compress

The `compress` utility is also used to create compressed files. The files compressed with `compress` retain their permissions, ownership, and modification times. As with the `gzip` utility, the original file is replaced with its compressed counterpart. The `.Z` extension is added to files created with the `compress` utility. Files created with the `compress` utility can be uncompressed using the `uncompress`, `compress`, or `gunzip` utilities. The `compress` utility is used with many of the same options used with the `gzip` utility. These options are shown in Table 6-19.

Table 6-19
Options Used with compress

<i>Option</i>	<i>Use</i>
-c	Results are sent to standard output while the file is left intact. This ensures that the original file is preserved.
-r	Recursively compresses directories.

Continued

Table 6-19 (continued)

Option	Use
-f	Forces compression.
-d	Uncompresses the specified file.
-V	Displays version information.
-v	Verbosely displays information.

The `compress` and `uncompress` utilities also use the standard syntax used by the `gzip` utility. The correct syntax is as follows:

```
compress -options FILENAME
```

bzip2

Another commonly used compression utility is `bzip2`. As with the `gzip` and `compress` utilities, the permissions, ownership, and file modification times remain intact. The original file is replaced with a compressed file of the same name ending in the `.bz2` extension. These files can be uncompressed using the `bunzip2` utility. The options that can be used with the `bzip2` utility are shown in Table 6-20.

Table 6-20
Options Used with `bzip2`

Option	Use
-c	Results are sent to standard output while the file is left intact. This ensures that the original file is preserved.
-d	Decompresses the specified file.
-z	Compresses the specified file.
-t	Perform a trial compression of the specified file.
-f	Forces files to be overwritten.
-k	Keeps the original file when compressing or decompressing.
-q	Suppresses warning messages.
-V	Displays version information.
-v	Verbosely displays information.

The `bzip2` utility uses the same syntax as the other compression utilities. The correct syntax is as follows:

```
bzip2 -options FILENAME
```

Managing Quotas

Objective

2.4 Devices, Linux File Systems, Filesystem Hierarchy Standard

- **Set and view disk quota.** Setup disk quota for a filesystem, edit user quota, check user quota, generate reports of user quota. Includes `quota`, `edquota`, `repquota`, `quotaon` commands.

Another part of managing files and directories on a Linux system involves working with *quotas*. Users can quickly fill system drives, and it seems that regardless of the amount of disk space installed on a system, that space is never enough. To prevent the users on your system from using all of the available space to store personal files, including large image and music files, you can use disk quotas. Quotas place a limit on the amount of disk space any single user can use. If the user reaches the quota, he or she is unable to create new files.

Linux includes several tools for managing quotas. The `quota`, `edquota`, `repquota`, and `quotaon` utilities work together to provide a total quota solution. Each of these utilities is covered in the sections below.

quota

The `quota` utility is used to view information about disk usage and user quotas. Users are able to view information only for their own user account and groups, but the superuser can view this information for all users and groups. Quota information can be displayed for all file systems listed in `/etc/fstab`: this file contains information regarding whether quotas can be used on the various file systems. The files used to manage quotas are located in file system root and are `quota.user` for user accounts and `quota.group` for group quotas. These files must be created by the superuser and must be created with read and write permissions only for the superuser account. These files are not manually edited when working with quotas. Instead, the tools covered in the following sections are utilized when managing quotas. Quotas are specified at the partition level, which enables you to control which partitions use quotas and to set these quotas independently.

Understanding the relationship between user and group quotas is important. User and group quotas are set individually. When a user creates a new file, that user's account and group both have ownership of the file. If a user's group has exceeded

its quota limit, the user will be unable to create new files, regardless of the user's account quota. This is important to remember when defining quotas. An example of the correct use of the quota utility is the following:

```
# quota angie
Disk quotas for user angie (uid 501): none
```

The quota utility has several options to specify which information is displayed. These options are covered in Table 6-21.

Table 6-21
Options Used with quota

<i>Option</i>	<i>Use</i>
-g	Displays group quota information.
-u	Displays user quota information. This is the default action.
-v	Displays quotas on file systems with no storage allocated.
-q	Displays information for file systems with usage exceeding the quota.

edquota

The edquota utility is used by the superuser to edit user and group quotas. The superuser uses this utility to set three different quota limits. The *soft limit* is a limit that can be exceeded for a period of time. The *grace period* is the period of time in which a user can exceed their soft limit. The *hard limit* is a limit at which a user can no longer create new files. The correct use of the edquota utility is shown below. In this example angie is given the same quota information as the jason account.

```
# edquota -p jason -u angie
```

The vi editor is the default editor used with the edquota utility. This utility writes the changes to the quota.user and quota.group files. You can use several options for editing quotas. These options are explained in Table 6-22.

Table 6-22
Options Used with edquota

<i>Option</i>	<i>Use</i>
-u	Specifies to edit the user quota. This is the default action.
-g	Specifies to edit the group quota.

<i>Option</i>	<i>Use</i>
-p user	Duplicates the quotas of the user specified for each user specified.
-t	Used to edit the grace period used for soft limits for each file system.

repquota

The `repquota` utility displays a summary of the disk usage and quotas for the specified file systems. The current number of files and amount of space (in kilobytes) for each user is displayed, along with any quotas created using `edquota`. Users on the system can use this utility for viewing quota information for their account; the superuser can view quota information for all users. Table 6-23 covers the options used with `repquota`.

Table 6-23
Options Used with `repquota`

<i>Option</i>	<i>Use</i>
-a	Displays information on all file systems indicated in <code>/etc/fstab</code> with quotas.
-v	Displays information for all quotas, even if there is no usage.
-g	Displays quota information for groups.
-u	Displays quota information for users. This is the default.

Following is an example of output produced by the `repquota` command:

```
# repquota -v /
*** Report for user quotas on / (/)
          Block limits
User      used  soft  hard  grace  used  soft  hard  grace
root     --    8     0     0      1     0     0
```

quotaon and quotaoff

The `quotaon` and `quotaoff` utilities are used to enable and disable file system quotas. The `quota.user` and `quota.group` files must exist before you can run the `quotaon` utility. Several options are available for use with both commands. The use of the option varies somewhat depending on whether the option is used with `quotaon` or `quotaoff`. The options and uses for both commands are shown in Table 6-24.

Table 6-24
Options Used with `quotaon` and `quotaoff`

<i>Option</i>	<i>Use with <code>quotaon</code></i>	<i>Use with <code>quotaoff</code></i>
-a	Enables quotas for all file systems in <code>/etc/fstab</code> that are configured with read/write quotas.	Disables quotas for all file systems listed in <code>/etc/fstab</code> .
-v	Displays a message for each file system with quotas enabled.	Displays a message for each file system affected.
-u	Enables user quotas; this is the default action.	Disables user quotas; this is the default action.
-g	Enables group quotas.	Disables group quotas.

Key Point Summary

File management is an important part of working with any system. Linux provides a variety of powerful tools that make this task easier and allow greater control over files. These tools are extremely important to understand for daily work with Linux systems and essential for the exams. Be sure that you are familiar with all points in the summary presented here before scheduling your exams.

- ♦ Directory changing is done with the `cd` command.
- ♦ Directory contents are displayed using `ls`.
- ♦ The `file` command is useful for examining files to determine file type.
- ♦ The `touch` command is used to create files and change time stamps on existing files.
- ♦ Files are copied with `cp` and moved or renamed with `mv`.
- ♦ The `dd` command is useful for copying and converting files simultaneously.
- ♦ The `rm` command is used to delete files.
- ♦ The `mkdir` command creates new directories.
- ♦ The `/etc` directory contains configuration files; the `/etc/skel` directory contains files copied to each user's home directory.
- ♦ The `/usr` directory contains `/usr/bin` with binaries executed by users, `/usr/sbin` with binaries executed by root user, `/usr/local` with applications that aren't part of the Linux operating system, `/usr/local/bin` with third-party software used by users, and `/usr/local/sbin` with software used by the root user that is installed after the initial operating system installation.

- ♦ The `/var/log` directory contains log files while `/var/spool` contains mail and printing files.
- ♦ The `/bin` directory holds binaries used during setup, and `/sbin` has binaries used by the root user.
- ♦ The `/` directory is the root directory, and it contains all other directories. The `/root` directory is the home directory for the root user. The `/home` directory contains the users' home directories. `/boot` stores the kernel and other files used by the boot loader. The `/dev` directory stores the devices used on the system, and the `/proc` directory is a virtual directory containing system information.
- ♦ The `find` utility is used to locate files matching the specified criteria.
- ♦ The `locate` utility uses the `slocate` database to index and search for files on the system. This database is updated using the `updatedb` command.
- ♦ The `which` utility is used to display the path of a specified command.
- ♦ The `whereis` command searches specific locations for the specified file and manual page entries.
- ♦ Links function as pointers to files. Hard links cannot span file systems, and changes to a soft link occur in all links to that file.
- ♦ Permissions are granted to users and group accounts using numeric and letter values.
- ♦ The `chown` command is used to change ownership of files and directories while the `chgrp` command is used to change the group assignment of files and directories.
- ♦ The `chmod` command is used to change permissions on files and directories.
- ♦ The `umask` is used to set the default permissions for newly created files.
- ♦ The SUID and SGID permissions allow files to be executed using a different user or group account.
- ♦ The sticky bit is used to allow only an owner to delete files and directories while allowing other users to write to them.
- ♦ The `tar` utility allows files to be archived as a single file and for this file to be compressed or uncompressed in one step.
- ♦ The `gzip`, `gunzip`, `compress`, and `bzip2` utilities are used to compress and uncompress files.
- ♦ Quotas are viewed with `quota` and `repquota`. They are enabled with `quotaon`, disabled with `quotaoff`, and edited using `edquota`.



STUDY GUIDE

The following questions and exercises will allow you to review the information covered in this chapter. Take your time when completing this section of the chapter, carefully reviewing any questions that you may have. Simply answering the question correctly is not as important as understanding the answer, so review any material that you might still be unsure of. Being comfortable with the questions and answers presented here will help you be more prepared for the certification exam questions.

Assessment Questions

1. Which of the following shows the use of an absolute path?
 - A. `ls`
 - B. `ls -al`
 - C. `ls /home/angie`
 - D. `/bin/ls`
2. The _____ command will display the contents of the current directory beginning with the letter a.
3. Which of the following commands will create a new file with the name `ourgroup`?
 - A. `file ourgroup`
 - B. `touch ourgroup`
 - C. `ls ourgroup`
 - D. `mkfile ourgroup`
4. Which command is used to rename the `ourgroup` file to `mygroup`?
 - A. `rn ourgroup mygroup`
 - B. `rn mygroup ourgroup`
 - C. `mv ourgroup mygroup`
 - D. `mv mygroup ourgroup`

5. The _____ command is used to copy files while converting them.
6. Which directory contains the system kernel?
 - A. /etc
 - B. /
 - C. /boot
 - D. /proc
7. Which directory contains the system configuration files?
 - A. /etc
 - B. /
 - C. /boot
 - D. /proc
8. Which directory contains the root user's home directory?
 - A. /home
 - B. /root
 - C. /sbin
 - D. /usr
9. Which directory contains the mail files?
 - A. /proc/mail
 - B. /var/spool
 - C. /var/mail
 - D. /usr/mail
10. Which utility is used to update the `slocate` database?
 - A. `locate`
 - B. `find`
 - C. `whereis`
 - D. `updatedb`
11. The _____ command is used to display the path to a specified command.
12. The default action of the `ln` command is to create a _____.

13. Which of the following results when a soft link is copied?
- A. A new copy of the soft link is created.
 - B. A hard link to the original file is created.
 - C. A hard link to the soft link is created.
 - D. A new copy of the original file is created.
14. Which of the following represents read, write, and execute permissions for owner and read and execute for all others?
- A. 755
 - B. 022
 - C. 733
 - D. 557
15. Which `umask` would create the default file permissions of read, write, and execute permissions for owner and read and execute for all others?
- A. 755
 - B. 022
 - C. 220
 - D. 002
16. Which command is used to change permissions of a file?
- A. `chown`
 - B. `chperm`
 - C. `chgrp`
 - D. `chmod`
17. Which value is used to represent the sticky bit?
- A. 1
 - B. 2
 - C. 3
 - D. 4
18. Which command (with options) is used to create a new archive file named `ang_home.tar.gz` that is compressed containing the `/home/angie` directory contents?

19. Which command is used to view quota information for a specific user?
- A. `edquota`
 - B. `quota`
 - C. `repquota`
 - D. `quotaon`
20. File systems utilizing quotas are configured in which file?
- A. `/etc/services`
 - B. `/etc/hosts`
 - C. `/etc/fstab`
 - D. `/etc/mtab`

Scenarios

1. When running the `ls` command you begin having trouble because the wrong command is being executed. What can you do to verify the path to the command is what you expected?
2. In the above example, what command can you type to ensure that the proper `ls` command is being run?

Answers to Chapter Questions

Chapter Pre-Test

1. The `locate` command utilizes the `slocate` database to find files.
2. The system kernel is located in the `/boot` directory.
3. The `umask` sets the default permissions for newly created files.
4. Soft links are allowed to span across file systems.
5. Quotas are enabled using the `quotaon` command.
6. The `ls` command is used to list files in a directory.
7. The `mkdir` command is used to create a directory.
8. The `touch` command can be used to create an empty file.
9. The `whereis` command is used to search the `PATH` statement for a command.
10. The `find` command recursively searches the directory for a specified filename.

Assessment Questions

1. **D.** The absolute path to the `ls` command is `/bin/ls`. See section “Managing Files” for more information.
2. `ls a*`. The wildcard is utilized to specify any character or set of characters, including none. This command would list files and directories with the name of simply `a` as well as all files and directories that begin with the letter `a`. See the section “Listing directory contents” for more information.
3. **B.** Files can be created using the `touch` command if the filename specified doesn’t already exist. If the file exists then the access time is simply changed. See the section “Changing file time stamp” for more information.
4. **C.** Files are renamed and moved using the `mv` command. See the section “Moving files” for more information.
5. `dd`. The `dd`, or direct dump, command is used to convert and copy files. See the “`dd`” section for more information.
6. **C.** The system kernel is stored in the `/boot` directory. See the “System directories” section for more information.
7. **A.** The system configuration files are located in the `/etc` directory. See the “Standard file locations” section for more information.
8. **B.** The root user’s home directory is the `/root` directory. See the “System directories” section for more information.
9. **B.** Mail files are found in the `/var/spool` directory. See the “Standard file locations” section for more information.
10. **D.** The `sllocate` database is updated using the `updatedb` command. See the “`locate`” section for more information.
11. `which`. The `which` command displays the path to the specified command. See the “`which`” section for more information.
12. **hard link**. By default, the `ln` command creates a hard link to the specified file. See the “Hard links” section for more information.
13. **D.** When a soft link is copied, a new copy of the original file is stored at the target location. See the “Symbolic links” section for more information.
14. **A.** The permission `7` presents the owner of the file with read (4), write (2), and execute (1) permissions. The group and other permissions here are read (4) and execute (1) for a total of 5. See the “Symbolic and numeric permissions” section for more information.
15. **B.** The `umask` used to create default permissions of `755` is `022`. The `umask` is used to filter permissions from those that are assigned to new files by default. This causes the `umask` to be the inverse of the permissions assigned to the files. In this case, the `umask` filters the write permissions from the group and other permissions. See the “`umask`” section for more information.

16. **D.** Permissions on files and directories is changed using the `chmod` command. The `chown` command is used to change ownership while the `chgrp` command changes the group assigned to the files or directories. The `chperm` command is invalid. See the “User and group permissions” section for more information.
17. **A.** The sticky bit value is (1) while the SUID value is 4 and SGID value is 2. See the “SUID and SGID” and “Sticky bit” sections for more information.
18. **tar -czf ang_home.tar.gz /home/angie.** The `tar` command is used with the `c` option to create a new archive and the `z` option specifies that the archive is compressed using the `gzip` utility. The `f` option specifies that the archive should be created as the specified file. Utilizing the absolute path ensures that all data within that directory is included in the archive. See the “tar” section for more information.
19. **B.** Quotas for specific users are viewed with the `quota` command. The `repquota` command will provide a report of quota usage for all users. The `edquota` command is used to edit quotas, and `quotaon` is used to enable quotas on the system. See the “Managing quotas” section for more information.
20. **C.** File systems utilizing quotas are configured in the `/etc/fstab` file. See the “quotaon and quotaoff” section for more information.

Scenarios

1. `which ls`

The `which` command uses the `ls` command as an argument. It locates the `ls` command that would run at the command line and displays the full path to that command. This allows you to verify that the command being executed is in the correct location.

2. `/bin/ls`

If the results from the first scenario aren't what you expect, then you can specify the absolute path to the `ls` command. This ensures that only the `ls` command at the specified location, in this case the `/bin` directory, is executed.

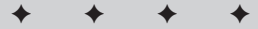
Using Documentation

EXAM OBJECTIVES

Exam 101 ♦ General Linux, Part 1

1.8 Documentation

- **Use and Manage Local System Documentation.** Use and administer the man facility and the material in /usr/doc/. Includes finding relevant man pages, searching man page sections, finding commands and man pages related to one, configuring access to man sources and the man system, using system documentation stored in /usr/doc/ and related places, determining what documentation to keep in /usr/doc/.
- **Find Linux documentation on the Internet.** Find and use Linux documentation at sources such as the Linux Documentation Project, vendor and third-party websites, newsgroups, newsgroup archives, mailing lists.
- **Write System Documentation.** Write documentation and maintain logs for local conventions, procedures, configuration and configuration changes, file locations, applications, and shell scripts.
- **Provide User Support.** Provide technical assistance to users via telephone, email, and personal contact.



CHAPTER PRE-TEST

1. What is the default location for Linux man page source documents?
2. Which section of the Linux man pages contains information on system administration commands?
3. The `apropos` and `whatis` commands search what database?
4. What Web site would you use to search Usenet postings for specific keywords?
5. Which documents contain task-specific information?
6. What utility will search command descriptions for specific keywords?
7. Where on the local system would you look to find documentation for the word processor that you have installed?
8. Which file contains configuration information used by the `man` command?
9. Which environment variable is used when displaying man pages?
10. Which option displays only the sections that contain matching man pages for the specified command?

When working with Linux, at times you will require further information on the use of commands, utilities, and system configuration. Although this book and other books can be very helpful, no book can contain up-to-date information on every subject. Luckily many terrific resources are available when more information is needed. Some reference sources are found on the local system, while others are available on the Internet. This chapter will inform you of some of the most useful places to search for more information. Knowing this information can prove invaluable for saving time when working with Linux systems and is essential when preparing for the exam.

Getting Help with Man Pages

Objective

1.8 Documentation

- **Use and Manage Local System Documentation.** Use and administer the man facility and the material in `/usr/doc/`. Includes finding relevant man pages, searching man page sections, finding commands and man pages related to one, configuring access to man sources and the man system, using system documentation stored in `/usr/doc/` and related places, determining what documentation to keep in `/usr/doc/`.

The Linux *manual pages* or *man pages* are the best place to check with questions concerning syntax and options for commands and utilities on the system. The man page documents are stored in a compressed format. The `man` command uncompresses and formats the pages for viewing. The pages are accessed using the `man` command followed by the command being researched. An example of the correct syntax is as follows:

```
# man ls
```

This command would search the manual pages for the `ls` command. When you open the man pages, the first thing displayed is a banner with the command and the man page being accessed. Also shown here is the FSF for Free Software Foundation. This appears as follows:

```
LS(1)                                FSF                                LS(1)
```

This is then followed by the command name and its function.

```
NAME
  ls - list directory contents
```

The command syntax is shown next.

SYNOPSIS

```
ls [OPTION]... [FILE]...
```

This is followed by a description of the command. After the description the command options are displayed and explained.

DESCRIPTION

List information about the FILES (the current directory by default). Sort entries alphabetically if none of `-cftuSUX` nor `--sort`.

```
-a, --all
    do not hide entries starting with .

-A, --almost-all
    do not list implied . and ..

-b, --escape
    print octal escapes for nongraphic characters

--block-size=SIZE
    use SIZE-byte blocks

-B, --ignore-backups
    do not list implied entries ending with ~
```

The man page ends with information concerning the author of the page, known bugs and bug reporting information, copyright, and directions to more information on the command.

AUTHOR

Written by Richard Stallman and David MacKenzie.

REPORTING BUGS

Report bugs to <bug-fileutils@gnu.org>.

COPYRIGHT

Copyright (c) 1999 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.
There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR
A PARTICULAR PURPOSE.

SEE ALSO

The full documentation for `ls` is maintained as a Texinfo manual. If the `info` and `ls` programs are properly installed at your site, the command

```
info ls
should give you access to the complete manual.

GNU fileutils 4.0p      March 2000      1
```

The spacebar is used to scroll down through the man page display one page at a time. The `Q` key exits the man page display. If you wish to search for text within the man page, the regular expressions can be used. An example is shown in the following with the characters used to locate the `option` string.

```
/option
```

Locating man pages

Linux man pages are stored on the system. The `MANPATH` variable contains the location of these files. The man pages are stored by default in the following locations.

- ♦ `/usr/man/man1`
- ♦ `/usr/man/man2`
- ♦ `/usr/man/man3`
- ♦ `/usr/man/man4`
- ♦ `/usr/man/man5`
- ♦ `/usr/man/man6`
- ♦ `/usr/man/man7`
- ♦ `/usr/man/man8`
- ♦ `/usr/man/man9`

The significance of the numbers is discussed in the following section of the chapter, “Searching man page sections.”



Be sure to know the default location of the man page source files. This will likely appear as an exam question.

The user can specify a different `MANPATH`. This allows a different set of man pages to be used. This is useful because some commands may not store their man pages in standard locations. Additionally, several options can be used with the `man` command, and an alternate path can be specified using an option as well. The options used by the `man` command are shown in Table 7-1.

Table 7-1
Options Used with man

Option	Use
<code>-C config-file</code>	Specifies a config file other than the default of <code>/etc/man.conf</code> .
<code>-M path</code>	Specifies directories to search for man pages.
<code>-P pager</code>	Specifies the pager, or the program used to format and display the man pages. The default pager is specified via the <code>PAGER</code> environment variable. The <code>more</code> and <code>less</code> pagers are frequently used.
<code>-S section-list</code>	Specifies a colon-delimited list of man sections to search.
<code>-a</code>	Specifies that all matching entries are to be displayed, not just the first.
<code>-c</code>	Specifies that the source page is to be reformatted.
<code>-d</code>	Specifies that debugging information is to be displayed instead of the actual man pages.
<code>-f</code>	Specifies that <code>man</code> is to behave like the <code>whatis</code> program (discussed later in this chapter).
<code>-h</code>	Displays help information for the <code>man</code> command.
<code>-k</code>	Specifies that <code>man</code> is to behave like the <code>apropos</code> program (discussed later in this chapter).
<code>-K</code>	Searches the man page sources for the specified string. The user is prompted as to whether he or she wishes to view each entry found.
<code>-m system</code>	Specifies an alternate set of man pages to search based on the system name specified.
<code>-w</code>	Specifies that the path to man pages is displayed instead of the actual man page.

An example of the use of the `-a` option follows. This causes the matching pages to be displayed in the order they are located. The user is first presented with the `crontab` entry in section one. When the user hits the `Q` key to exit this man page, the entry found in section five is displayed.

```
# man -a crontab
```

The `-w` option is useful for discovering the location of man page entries. Using this option for the `crontab` utility produces the following.

```
# man -w crontab
/usr/man/man1/crontab.1.gz
```



Be sure to know the search options and their function. These include `-a`, `-K`, and `-k`.

Searching man page sections

Linux man page information exists in a collection of files. These files are grouped into sections, with each section containing a specific type of information. Table 7-2 lists the sections and their uses.

Table 7-2
Man Page Sections

<i>Section</i>	<i>Use</i>
1	User commands and applications
2	System calls and kernel errors
3	Library calls
4	Device drivers and network protocols
5	Standard file formats
6	Games and demonstrations
7	Miscellaneous files and documents
8	System administration commands
9	Obscure kernel specs and interfaces

When an argument is passed to the `man` command, the sections are searched in a specific order and the first match is returned. The default search order is as follows: 1, 8, 2, 3, 4, 5, 6, 7, 9.

It is also possible to specify the man page section to search. If you wish to search section five for information on the `crontab` utility, you would use the following command:

```
# man 5 crontab
```

Using the `-a` option shown in Table 7-1, you can examine all of the matching man pages for the `crontab` utility. This is done using the following command.

```
# man -a crontab
```

Searching with `whatis`

The `whatis` utility is not explicitly mentioned in the exam objectives, but knowing its use can come in handy on the exam. The `whatis` utility is used to search the `whatis` database for matching entries. This database is created using the `/usr/bin/makewhatis` command. This database contains short descriptions found in the man pages of system commands. An example of its use is the following:

```
# whatis passwd
passwd          (1) - update a user's authentication
tokens(s)
passwd          (1ssl) - compute password hashes
passwd          (5) - password file
passwd.nntp [passwd] (5) - passwords for connecting to remote
Nntp servers
```

As you can see in this example, the `passwd` command has entries in sections one and five of the man pages. It is also found in section one of the `ssl` command man pages.

The `man -f` command searches this database for all entries matching the keyword entered. The command names listed in the database are searched for a matching entry. The following is an example of the output produced by this command.

```
# man -f passwd
passwd          (1) - update a user's authentication
tokens(s)
passwd          (1ssl) - compute password hashes
passwd          (5) - password file
passwd.nntp [passwd] (5) - passwords for connecting to remote
Nntp servers
```

These commands perform the same search. The commands and man page sections where they are located are displayed. This can be helpful for locating man page source sections and variants of commands.

Searching with `apropos`

Like `whatis`, the `apropos` command is not explicitly mentioned in the exam objectives, but is useful to know come exam time. And also like the `whatis` utility, the `apropos` command uses the `whatis` database. This command is used to search both the command names and the descriptions for the keyword specified. The following is an example of the `apropos` command:

```
# apropos password
chpasswd          (8) - update password file in batch
gpasswd          (1) - administer the /etc/group file
htpasswd         (1) - Create and update user
authentication files
nwpasswd        (1) - Change a user's password
passwd          (1) - update a user's authentication
tokens(s)
passwd          (1ssl) - compute password hashes
passwd          (5) - password file
passwd.nntp [passwd] (5) - passwords for connecting to remote
NNTTP servers
pg_passwd       (1) - Manipulate the flat password file
pwupdate        (8) - updates passwd and shadow NIS map
rpc.yppasswdd [rpc] (8) - NIS password update daemon
smbpasswd       (5) - The Samba encrypted password file
smbpasswd       (8) - change a users SMB password
ypchfn [yppasswd] (1) - change your password in the NIS
database
ypchsh [yppasswd] (1) - change your password in the NIS
database
yppasswd        (1) - change your password in the NIS
database
```

The following is an example of the `man -k` command:

```
# man -k password
chpasswd          (8) - update password file in batch
gpasswd          (1) - administer the /etc/group file
htpasswd         (1) - Create and update user
authentication files
nwpasswd        (1) - Change a user's password
passwd          (1) - update a user's authentication
tokens(s)
passwd          (1ssl) - compute password hashes
passwd          (5) - password file
passwd.nntp [passwd] (5) - passwords for connecting to remote
NNTTP servers
pg_passwd       (1) - Manipulate the flat password file
pwupdate        (8) - updates passwd and shadow NIS map
rpc.yppasswdd [rpc] (8) - NIS password update daemon
smbpasswd       (5) - The Samba encrypted password file
smbpasswd       (8) - change a users SMB password
ypchfn [yppasswd] (1) - change your password in the NIS
database
ypchsh [yppasswd] (1) - change your password in the NIS
database
yppasswd        (1) - change your password in the NIS
database
```

As you can see, these commands produce the same function. This can be particularly useful when searching for commands using keywords.

Configuring man page access

As was mentioned earlier in the chapter, the `/usr/man` directory is the default location for man page source files. The `MANPATH` environment variable can be used to change the default search path for man page source files. The `MANPATH` variable will overwrite the default search path for man pages, so including the path to existing man pages, if they will be needed, is important. Below is an example of a `MANPATH` variable setting added to a `/home/user/.profile` file.

```
Export MANPATH=/usr/local/man:/usr/man/preformat:/usr/man:/usr/X11R6/man
```

Most of the documents stored in `/usr/man` are compressed and unformatted. The `man` command uses the `/etc/man.config` file for information on the proper display of these files. This file contains `MANPATH` information as well as compression, formatting, and pager settings. Using the option `-C` shown in Table 7-1, a different configuration file can be specified.

The `man` command is located in `/usr/bin`. This directory needs to be located in the `PATH` environment variable, or the command must be run using the absolute path `/usr/bin/man`.

Using Documentation Stored in `/usr/doc`

Objective

1.8 Documentation

- **Use and Manage Local System Documentation.** Use and administer the man facility and the material in `/usr/doc/`. Includes finding relevant man pages, searching man page sections, finding commands and man pages related to one, configuring access to man sources and the man system, using system documentation stored in `/usr/doc/` and related places, determining what documentation to keep in `/usr/doc/`.

Along with the man pages, a variety of other documentation can be stored on the local system. Table 7-3 shows some common locations of documentation and the data stored there.

Table 7-3
Documentation Locations

<i>Location</i>	<i>Documentation</i>
<code>/usr/doc/program_name</code>	Documentation for <i>program_name</i>
<code>/usr/doc/FAQ</code>	Frequently Asked Questions

Location	Documentation
<code>/usr/doc/HTML</code>	Documentation in HTML format
<code>/usr/doc/HOWTO</code>	HOWTO files
<code>/usr/info</code>	Documentation presented using the <code>info</code> command

The contents of these directories will vary depending on the utilities and applications that are installed. These directories are excellent locations to check when looking for various documentation. The `/usr/doc/program_name` directory should be used to store documentation on installed applications accessed by users on the system.

Documentation on the Internet

Objective

1.8 Documentation

- **Find Linux documentation on the Internet.** Find and use Linux documentation at sources such as the Linux Documentation Project, vendor and third-party websites, newsgroups, newsgroup archives, mailing lists.

The Internet is also an excellent source of up-to-date and thorough documentation. Linux is used by a variety of people with a wide range of hardware and software. If you have a question about something, chances are the answer can be found by looking online. In fact, so much information is available online that it helps to know where to look when you have a question.

Linux Documentation Project

The Linux Documentation Project works to develop reliable documentation and to collaborate on all aspects of Linux documentation. This group is comprised of volunteers who work to provide free, high quality documentation available on the Internet. This project uses HOWTOs, guides, FAQs, and man pages. All of this information can be located at <http://www.linuxdoc.org/> and is searchable and downloadable from this site.

HOWTOs

The HOWTO and mini-HOWTO documents are written to provide “how to” information on specific subjects. These subjects can range from how to install and configure Apache to how to configure 3Dfx adapters. These documents can be of great help whenever you need instruction on a specific subject.

Guides

Guides are written to provide detailed information on various aspects of Linux. Some of the guides available include the following:

- ♦ Install Guide
- ♦ Network Guide
- ♦ Programmer's Guide
- ♦ Systems Administrator's Guide
- ♦ User's Guide

These guides cover a variety of tasks related to that specific aspect of Linux systems. Many of these guides are regularly updated whenever necessary.

Vendor sites

Many vendors also maintain documentation and support on their sites. All of the major distributions provide a searchable support system for locating information on a range of topics. Hardware vendors with Linux support also maintain information such as drivers and HOWTOs on their sites.

Newsgroups

There are many newsgroups available via Usenet that are dedicated to Linux topics. If you have a specific question, someone else probably has the same question and you may find the answer by checking the newsgroups. The site <http://groups.google.com> is useful for searching Usenet postings using keywords.

When using Usenet it is important to follow proper netiquette. Research a question before you ask. Check FAQs and read other postings, because asking a question that has already been answered isn't likely to produce a good response.

Mailing lists

There are many mailing lists devoted to Linux. These lists range from generic Linux information to very specific task- or application-related lists. When using mailing lists, the rules of netiquette apply as well.



You can expect to see at least one test question concerning where to search online for specific information.

Creating Documentation

Objective**1.8 Documentation**

- **Write System Documentation.** Write documentation and maintain logs for local conventions, procedures, configuration and configuration changes, file locations, applications, and shell scripts.

One area that is often neglected by systems administrators is documentation. Such neglect is unfortunate because documentation is an important task that should not be considered optional. Documentation should be created so that in the event that the system is destroyed any knowledgeable systems administrator can rebuild it. All details concerning hardware, software installations, and configurations should be maintained within the system logs. Passwords and special user accounts should also be stored here.

The system log should be updated whenever a change is made to the system. Be sure to include any problems and their resolution within the system log. Keeping records in this way will make it easy to track problems that occur on the system.

One consideration to examine when creating the system log is whether the log should be in electronic format or available on paper. If the documentation is stored in electronic format, it should exist in multiple locations and be accessible if the system is not functioning. Paper documentation is easily accessible but is more effort to organize effectively. Whichever format you choose be sure that the system log is maintained in a secure location.

As the systems administrator, you also may need to create documentation for users of the system. Be sure to write documentation for the level of the user. It is better to include too much detail and information than to provide not enough.

Creating documentation is an ongoing process. The documents will continually be in need of updates and revisions. Such revision is often overlooked because the benefits are not always immediately evident. However, this task often determines whether someone receives praise for a job well done or whether he or she is forced to look for a new job. Problems on the system are inevitable and quick recovery due to proper documentation can be a real lifesaver.

Providing Technical Support

Objective**1.8 Documentation**

- **Provide User Support.** Provide technical assistance to users via telephone, email, and personal contact.

Another important task for systems administrators is providing technical support. Working with users can be frustrating at times. However, remembering that without users there would be no need for systems administrators is important. Difficult situations are often caused by a lack of communication and knowledge. Documentation can be a great resource to use for educating users. Be sure they know the proper procedure to follow when a problem occurs.

Users can contact you for support in several different ways. Telephone and e-mail are two popular options. These methods can allow for initial troubleshooting and swift problem resolution. At times, however, a personal visit is more helpful. Such a visit provides a wonderful opportunity to observe the user and view the problem that is occurring.

Whichever method of contact you have with the user, be sure that you keep a few things in mind. The user should realize when it is appropriate to ask for help. Your time is valuable and not every problem should result in a call to you. Be sure to remain patient and listen to what the user is saying. Problems can be made much worse when miscommunications occur between the user and administrator. Be sure to maintain contact with the user when resolving a problem. Let the user know when they will hear from you and be sure that they do. User frustration rises when they are unsure of the status of their support request.

Although human relations can be trying, they also can be rewarding. A positive attitude combined with a professional approach is essential. When the user and administrator work together, their common goal is much easier to reach.

Key Point Summary

This chapter provides valuable information about locating more information. Knowing the available resources for support can save time and energy and can allow you to work more efficiently and be a better systems administrator. No one knows all of the answers about all software so knowing where to turn for help is essential to getting the job done correctly. The topics covered in this chapter are helpful in the real world and essential for the exam.

- ♦ Man pages contain information on commands, utilities, and applications on Linux systems.
- ♦ Man pages are accessed using the `man` command.
- ♦ The `MANPATH` and `PAGER` environment variables are both referenced when using the `man` command.
- ♦ The `/usr/man` directory is the default location for man pages while `/usr/doc` stores application-related documentation.
- ♦ The `whatis` and `apropos` commands search the `whatis` database for information on Linux commands.

- ♦ The Linux Documentation Project provides access to HOWTOs, FAQs, and guides that are useful for gaining more information about Linux and various tasks.
- ♦ Maintaining proper documentation is a key systems administration task.
- ♦ When you provide technical support, remaining patient and professional is important.



STUDY GUIDE

The following questions and exercises will allow you to review the information covered in this chapter. Take your time when completing this section of the chapter, carefully reviewing any questions that you may have. Simply answering the question correctly is not as important as understanding the answer, so review any material that you might still be unsure of. Being comfortable with the questions and answers presented here will help you be more prepared for the certification exam questions.

Assessment Questions

1. Which of the following would display all man page section numbers with the `ls` command in their description?
 - A. `man -w ls`
 - B. `man -a ls`
 - C. `man -f ls`
 - D. `man -K ls`
2. Which of the following commands would display all man page entries for the `ls` command?
 - A. `man -w ls`
 - B. `man -a ls`
 - C. `man -f ls`
 - D. `man -K ls`
3. Which of the following commands would display the path to the man page for the `ls` command?
 - A. `man -w ls`
 - B. `man -a ls`
 - C. `man -f ls`
 - D. `man -K ls`

4. Which of the following would search the content of all man pages for the `ls` command?
- A. `man -k ls`
 - B. `man -a ls`
 - C. `man -f ls`
 - D. `man -K ls`
5. The _____ command functions like the `man -f` command.
6. The _____ command functions like the `man -k` command.
7. Which section of man page documents contains information on kernel errors?
- A. 1
 - B. 2
 - C. 5
 - D. 8
8. Which section of man page documents contains information on standard file formats?
- A. 1
 - B. 2
 - C. 5
 - D. 8
9. Which section of man page documents contains information on user commands?
- A. 1
 - B. 2
 - C. 5
 - D. 8
10. Which section of man page documents contains information on administrative commands?
- A. 1
 - B. 2
 - C. 5
 - D. 8

11. Which of the following locations is used to store FAQs on the local system?
 - A. `/usr/doc/info`
 - B. `/usr/doc`
 - C. `/usr/doc/FAQ`
 - D. `/usr/doc/HTML`

12. Which type of document contains instructions on performing certain tasks?
 - A. HOWTO
 - B. FAQ
 - C. Guides
 - D. Mailing lists

Scenarios

1. When using the command `man -f passwd`, no entry is located. What might you examine to resolve this problem?
2. You are having trouble configuring the new scanner on your computer. Where would you search for information on this topic?

Answers to Chapter Questions

Chapter Pre-Test

1. The default location of Linux man page documents is `/usr/man/man#`.
2. Information on system administrative commands is found in the man page section 8.
3. The `apropos` and `whatis` commands search the `whatis` database.
4. The `http://groups.google.com` Web site is useful for searching Usenet postings.
5. Task-specific information is stored in HOWTO documents.
6. The `man -k` utility searches command man pages for specific words.
7. Documentation for installed applications is found in `/usr/doc/programname`.
8. The file `/etc/man.conf` contains configuration information used by the `man` command.
9. The `PAGER` environment variable is used for displaying man pages.

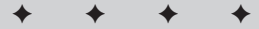
10. The `-w` option displays the man pages sections that contain information for the command specified.

Assessment Questions

1. **A.** The `man -w` command is used to search the descriptions of a man page document for a specific string. See the section “Locating man pages” for more information.
2. **B.** The `man -a` command displays all matching man page entries for the command specified. See the section “Locating man pages” for more information.
3. **A.** The `man -w` command displays the path to the man page located. See the section “Locating man pages” for more information.
4. **D.** The `man -K` command searches all man page entries for the string specified. See the section “Locating man pages” for more information.
5. **whatis.** The `whatis` command functions like `man -f`. See the section “Searching with `whatis`” for more information.
6. **apropos.** The `apropos` command functions like `man -k`. See the section “Searching with `apropos`” for more information.
7. **B.** Man page section 2 contains information on kernel errors. See the section “Searching man page sections” for more information.
8. **C.** Man page section 5 contains information on standard file formats. See the section “Searching man page sections” for more information.
9. **A.** Man page section 1 contains information on user commands. See the section “Searching man page sections” for more information.
10. **D.** Man page section 8 contains information on administrative commands. See the section “Searching man page sections” for more information.
11. **C.** The path to the FAQs stored on the local system is `/usr/doc/FAQ`. More information can be found in the section “Using documentation stored in `/usr/doc`”
12. **A.** The HOWTO documents provide step-by-step details on performing a variety of specific tasks. See the section “Documentation on the Internet” for more details.

Scenarios

1. Ensure that the man page paths are configured correctly. Run the `makewhatis` command to build the `whatis` database.
2. Search the Linux HOWTO documents. Web searches using `http://groups.google.com` will display Usenet posts on the topic as well.



Understanding the Boot Process

EXAM OBJECTIVES

Exam 101 ♦ General Linux, Part 1

2.6 Boot, Initialization, Shutdown, Run Levels

- **Boot the system.** Guide the system through the booting process, including giving options to the kernel at boot time, and check the events in the log files. Involves using the commands: `dmesg` (`lilo`). Involves reviewing the files: `/var/log/messages`, `/etc/lilo.conf`, `/etc/conf.modules` | `/etc/modules.conf`
- **Change runlevels and shutdown or reboot system.** Securely change the runlevel of the system, specifically to single user mode, halt (shutdown) or reboot. Make sure to alert users beforehand, and properly terminate processes. Involves using the commands: `shutdown`, `init`

Exam 102 ♦ General Linux, Part 2

2.2 Linux Installation and Package Management

- **Install a boot manager.** Select, install and configure a boot loader at an appropriate disk location. Provide alternative and backup boot options (like a boot floppy disk). Involves using the command: `lilo`. Involves editing the file: `/etc/lilo.conf`.

CHAPTER PRE-TEST

1. Which process always has a process ID of 1?
2. Which type of startup system do Red Hat and Debian use?
3. Which commands are used to change the runlevel?
4. Which file configures the `init` process?
5. Which file is displayed to console users before the login prompt?
6. How would you tell `init` to not run any scripts when the system starts?
7. What does it mean when the LILO prompt displays only `LIL`?
8. Which boot parameter tells the kernel the amount of RAM installed in the system?
9. Which file is used to configure LILO?
10. Which runlevel is used to reboot the system?

This chapter covers the installation of the LILO boot manager and the processes the system goes through during startup. These processes set up the system and allow you to customize what happens during booting. Troubleshooting information is also covered that will help should the system not boot.

Using LILO

Objective

2.6 Boot, Initialization, Shutdown, Run Levels

- **Boot the system.** Guide the system through the booting process, including giving options to the kernel at boot time, and check the events in the log files. Involves using the commands: `dmesg (lilo)`. Involves reviewing the files: `/var/log/messages`, `/etc/lilo.conf`, `/etc/conf.modules` | `/etc/modules.conf`

2.2 Linux Installation and Package Management

- **Install a boot manager.** Select, install and configure a boot loader at an appropriate disk location. Provide alternative and backup boot options (like a boot floppy disk). Involves using the command: `lilo`. Involves editing the file: `/etc/lilo.conf`.

LILO, the Linux Loader, is the most popular boot manager and has been used for many years. Other boot managers have come out that provide other features, but LILO is still the standard that almost all distributions use.

LILO is usually installed to the master boot record (MBR) of a hard disk. This is the place on a hard disk where the system's BIOS looks for the operating system's boot information. LILO can also be installed to the boot sector of a partition, instead of the MBR for the entire drive. This is done when another primary boot manager is used, which will then execute LILO when Linux is booted.

When the system boots and executes LILO, you will see a boot prompt that looks like this:

```
LILO:
```

Pressing the Tab key will list the possible boot selections. Pressing the Enter key will boot the default selection that was configured. You can pass parameters to the kernel by appending them after the boot selection name. If a parameter is not interpreted by the kernel, it is then checked as an environment variable. Environment variables should be in the form of `variable=value`. Finally, if the parameter is not recognized by the kernel and is not in the form of an environment variable, it is passed on to the boot process, which is usually `init`. Table 8-1 lists the possible kernel parameters.



If Linux does not see all the RAM installed in your system, tell it the correct amount using the `mem=` boot parameter.

Table 8-1
Kernel Parameters

<i>Parameter</i>	<i>Function</i>
<code>init=</code>	Sets the initial command to be run by the kernel. By default this is <code>init</code> .
<code>nfsaddr=</code>	Sets the NFS boot address, which is used for network booting.
<code>nfsroot=</code>	Sets the NFS root name, which is used for network booting.
<code>no387</code>	Disables the use of the math coprocessor on the CPU. Math Coprocessor Emulation must be enabled in the kernel for this to work.
<code>no-hlt</code>	Instructs Linux to not use the <code>hlt</code> CPU instruction, which is broken on some older chips.
<code>root=</code>	Changes the root file system device to the one specified. For example, to mount a floppy disk as the root file system you would use <code>root=/dev/fd0</code> .
<code>ro</code>	Mounts the root file system read-only so that <code>fsck</code> can fix any errors on the disk.
<code>rw</code>	Mounts the root file system read/write. This is the default.
<code>reserve=</code>	Stops a driver from probing the specified IO port region. Some hardware reacts badly to this scanning.
<code>mem=</code>	Tells the kernel the amount of RAM installed in the system.
<code>panic=N</code>	Tells the kernel to reboot <i>N</i> seconds after a kernel panic. A kernel panic occurs when a nonrecoverable error happens in the kernel and the entire machine crashes.
<code>reboot=[warm cold] [, [bios hard]]</code>	Sets the type of reboot used when the system is instructed to restart.
<code>nosmp</code>	Disables SMP support.
<code>maxcpus=N</code>	Sets the maximum number of CPUs used to <i>N</i> .

Configuring LILO

LILO is easily configurable through its configuration file, `/etc/lilo.conf`. The `lilo.conf` file has many different entries to dictate where and how LILO is installed and how it accesses the operating systems installed on the system. Table 8-2 lists the global entries and Table 8-3 lists the per-image entries.

Table 8-2
Global lilo.conf Entries

Entry	Function
<code>backup=<filename></code>	Copies the original boot sector to the specified backup file.
<code>boot=<boot device></code>	Specifies the name of the disk device or partition that the new boot record is written to. If this entry is omitted, the new boot sector will be written to the current root partition.
<code>change-rules</code>	Specifies boot-time changes to the partition type numbers. See the <code>lilo.conf</code> man page for more information.
<code>compact</code>	Causes LILO to try to merge read requests for adjacent disk sectors into a single request. This can greatly speed up booting from a floppy disk.
<code>default=<name></code>	Defines the default boot image to use. If this is omitted, the first boot entry listed will be used as the default.
<code>delay=<tenth of seconds></code>	Causes LILO to wait the specified number of tenths of seconds before booting the default image.
<code>disk=<device name></code>	Lets you specify nonstandard parameters for the specified disk.
<code>disktab=<disktab file></code>	Specifies the location of the disk parameter file. This file was once used to hold disk geometry information. The use of disk tabs is now discouraged.
<code>fix-table</code>	Lets LILO adjust the sector/head/cylinder relationship to the linear addresses on the disk drive. This is sometimes used with other operating systems that do not convert the information the same way as LILO.
<code>force-backup=<backup file></code>	Has the same function as the <code>backup</code> entry, but overwrites an existing file if it exists.
<code>ignore-table</code>	Instructs LILO to ignore corrupt partition table information.
<code>install=<boot loader></code>	Tells LILO which boot loader to install, either the text mode or the menu mode.
<code>lba32</code>	Generates 32-bit logical block addresses instead of sector/head/cylinder addresses. This allows booting from disks with more than 1024 cylinders.

Continued

Table 8-2 (continued)

Entry	Function
linear	Generates linear addresses instead of sector/head/cylinder addresses. This is required for some SCSI disks.
lock	Tells LILO to remember the last boot command line and use it as the default until another is manually entered.
map=<map file>	Specifies which map file to use. By default this is /boot/map.
menu-title=<menu title>	Specifies the title for the boot menu.
menu-scheme=<color scheme>	Specifies the boot menu color scheme.
message=<message file>	Specifies a text file with a message that is displayed before the boot prompt.
nowarn	Disables warnings about possible problems.
prompt	Instructs LILO to show the boot prompt, instead of booting the default image right away.
serial=<parameters>	Allows you to boot the system from a serial interface instead of a normal monitor.
timeout=<tenths of seconds>	Tells LILO to wait the specified number of tenths of seconds for keyboard input at the boot prompt before booting the default image.
verbose=<number>	Enables verbose messages. The level of verbosity can be set from 1 to 5.

Table 8-3
Per-Image lilo.conf Entries

Entry	Function
image=<path name>	The path to the device or file containing the kernel boot image.
other=<device name>	Used to boot another operating system, such as Windows. The device name should point to the partition or disk of the other operating system.
label=<name>	The name of the boot image as shown in the boot menu.
alias=<name>	A second name for a boot image.
lock	Same as the global entry, but applies only to this image.

Entry	Function
password=<password>	Password-protects this image at boot.
restricted	Requires the password option above to be used only if a kernel parameter is passed at boot. This will stop someone from changing the root device at boot or booting to single-user mode.
append=<string>	Passes the specified string to the kernel during boot, just as if you had entered it at the boot prompt.
initrd=<name>	Specifies the initial RAM disk image to use at boot.
literal=<string>	Like append, but removes all other options.
ramdisk=<size>	Specifies the size of the optional RAM disk.
read-only	Specifies that the root file system should be mounted read-only, so that it can be checked with <code>fsck</code> .
read-write	Specifies that the root file system should be mounted read/write.
root=<root device>	Specifies the device to be used as the root file system.
vga=<mode>	Enables the use of the VGA graphics mode during boot.
optional	Does not use the image if it is not available when LILO is installed. Useful for testing new kernels that may not always be there.

The following is an example of a `/etc/lilo.conf` file, with explanations.

```
# Support LBA for large hard disks.
#
lba32
```

This line enables LBA mode, which allows LILO to boot from a hard drive with more than 1024 cylinders.

```
boot=/dev/hda
```

This line tells LILO to install itself to the master boot record of `/dev/hda`. If a partition is specified, such as `/dev/hda3`, it will install itself into the boot block of the partition.

```
root=/dev/hda3
```

This line specifies that the root volume is to be mounted from `/dev/hda3`.

```
install=/boot/boot.b
```

This entry causes LILO to install the specified file to the boot sector.

```
prompt
delay=100
timeout=100
```

These lines tell LILO to display a boot prompt and wait 10 seconds (100 tenths) before booting to the default image. If someone presses a key, it will cancel the delay option. If a person does not press a key for 10 seconds after that, the `timeout` option will boot the default image.

```
vga=normal
```

This line tells LILO to boot in normal VGA mode, which is 80 columns by 25 lines.

```
default=Linux2.4
```

This line sets the default boot image, which will be configured in the next section.

```
image=/boot/vmlinuz-2.2.17-reiser
label=Linux
read-only
append="mem=256M hdd=ide-scsi"
```

This is the first image block. It instructs LILO to boot the kernel file named `/boot/vmlinuz-2.2.17-reiser`. This block is named “Linux” in the boot menu. The root file system will be mounted read-only when the system first boots, so that it can be checked with `fsck`. It will later be remounted as read/write by a startup script. The string `"mem=256M hdd=ide-scsi"` is passed to the kernel at boot. This string tells the kernel that the system has 256MB of RAM installed and configures `/dev/hdd` as an IDE drive using SCSI emulation, which is required for IDE CD-R drives.

```
image=/boot/vmlinuz-2.4.0-test10
label=Linux2.4
read-only
append="hdd=ide-scsi"
```

This is another image block, which is used to test a new kernel. The rest of the options are the same as the previous example.

```
other=/dev/hda4
label=Windows
restricted
```

This block is used to boot another operating system, and in this case, Windows. The `other` directive tells LILO to use the boot information contained in the `/dev/hda4` boot block. The `restricted` entry tells LILO to not accept any boot parameters on this entry.

Installing and updating LILO

Any time that you modify the `lilo.conf` file or install a new kernel, you must reinstall the LILO boot sector. To simply reinstall LILO using the existing `lilo.conf` file, just type **lilo** at the prompt. Many of the entries in the `lilo.conf` file can be specified on the `lilo` command line. Table 8-4 lists all of the possible command-line options for `lilo`.



Whenever a change is made to `/etc/lilo.conf`, you must remember to rerun `lilo` to install the new boot sector.

Table 8-4
lilo Options

Option	Function
<code>-c</code>	Enables map compaction so that read requests from adjacent sectors are combined into one.
<code>-C <config file></code>	Uses a different configuration file instead of the standard <code>/etc/lilo.conf</code> .
<code>-d <delay></code>	Specifies the amount of time in tenths of seconds to wait before booting the default kernel image.
<code>-D <label></code>	Uses the specified kernel label as the default image.
<code>-f <disk-tab file></code>	Uses a different disk-tab file instead of the default <code>/etc/disktab</code> .
<code>-i <boot loader file></code>	Uses a different boot loader file instead of the default <code>/boot/boot.b</code> .
<code>-I <label></code>	Prints the kernel image path for the requested label.
<code>-l</code>	Generates linear addresses instead of sector/head/cylinder addresses.
<code>-L</code>	Generates 32-bit LBA addresses, so that a drive with more than 1024 cylinders can be accessed.
<code>-m <map file></code>	Uses the specified map file, instead of <code>/boot/map</code> .
<code>-P {fix ignore}</code>	Tells LILO whether to fix or ignore corrupt partition table information.

Continued

Table 8-4 (continued)

Option	Function
-q	Prints information contained in the map file telling which images LILO will boot, as well as other information from <code>lilo.conf</code> .
-r <root directory>	Changes the perceived root when installing LILO in a troubleshooting situation.
-R <command line>	Specifies a command to run the next time the boot loader is executed, after which the command is erased. Used with reboot scripts.
-s <save file>	Saves the existing boot sector to the specified filename.
-S <save file>	Saves the existing boot sector to the specified filename, and overwrites an existing file if it exists.
-t	Enables test mode. Use with the <code>-v</code> parameter to display what LILO would do if installed.
-T <option>	Prints out system information for the requested option. See the <code>lilo</code> man page for a list of the options.
-u <device name>	Uninstalls LILO from the specified device.
-U <device name>	Uninstalls LILO from the specified device, but does not perform a time stamp check.
-v	Increases the verbosity of information displayed. Up to five <code>-v</code> parameters can be added to further increase the amount of information displayed.
-V	Displays the LILO version number.

Viewing boot messages

During the boot process the messages displayed to the console are saved to a log file, usually named `/var/log/dmesg`. To view this log you can either display the log file or use the `dmesg` command. Below is an example of the result of the use of this command:

```
Linux version 2.2.16-22 (root@porky.devel.redhat.com) (gcc version egcs-2.91.66
19990314/Linux (egcs-1.1.2 release)) #1 Tue Aug 22 16:49:06 EDT 2000
Detected 579006 kHz processor.
Console: colour VGA+ 80x25
Calibrating delay loop... 1048.58 BogoMIPS
Memory: 46712k/49152k available (1048k kernel code, 412k reserved, 916k data, 64
k init, 0k bigmem)
Dentry hash table entries: 262144 (order 9, 2048k)
Buffer cache hash table entries: 65536 (order 6, 256k)
```

```
Page cache hash table entries: 16384 (order 4, 64k)
VFS: Diskquotas version dquot_6.4.0 initialized
CPU: Intel Pentium III (Coppermine) stepping 03
Checking 386/387 coupling... OK, FPU using exception 16 error reporting.
Checking 'hlt' instruction... OK.
POSIX conformance testing by UNIFIX
mtrr: v1.35a (19990819) Richard Gooch (rgooch@atnf.csiro.au)
PCI: PCI BIOS revision 2.10 entry at 0xfd9de
PCI: Using configuration type 1
PCI: Probing PCI hardware
PCI: Enabling memory for device 00:78
PCI: Enabling memory for device 00:80
Linux NET4.0 for Linux 2.2
```

Understanding Runlevels and `init`

Objective

2.6 Boot, Initialization, Shutdown, Run Levels

- **Change runlevels and shutdown or reboot system.** Securely change the runlevel of the system, specifically to single user mode, halt (shutdown) or reboot. Make sure to alert users beforehand, and properly terminate processes. Involves using the commands: `shutdown`, `init`

The Linux boot process is very customizable, but can differ between distributions in the details. To understand the boot process and how to troubleshoot and debug it, you need to be familiar with the concept of runlevels and the `init` process.

Using runlevels

A Linux system can be configured to have several different configurations or states. These different states are known as *runlevels*, and there are seven of them, numbered 0 through 6. Runlevels 7 through 9 are valid, but unused because traditional UNIX operating systems did not support them. To see which runlevel the system is currently in, use the `runlevel` command. For example:

```
debian~# runlevel
N 2
```

The *N* in this example specifies that the system was booted to this runlevel and that there was no previous runlevel. On most distributions the runlevel dictates the configuration of the system. One runlevel may be for text mode and another for GUI mode. Separate runlevels are sometimes used to have network and nonnetwork configurations. Table 8-5 lists the standard runlevel uses for Red Hat and its variants.



To see which runlevel the system is currently in, just type **runlevel**.

Table 8-5
Red Hat Runlevels

<i>Runlevel</i>	<i>Function</i>
0	Changing to this runlevel causes the system to halt.
1	Takes the system to single-user mode, which is normally used for troubleshooting.
2	Multuser, but no networking support.
3	The standard runlevel for networked multuser text login.
4	Undefined.
5	The standard runlevel for multuser GUI login.
6	Reboots the system.



The runlevel can be changed using either `init` or `telinit`.

Debian has the same configuration for runlevels 0, 1, and 6. But, by default the normal runlevel used to boot the system for text or GUI is 2, and the rest are undefined. The undefined runlevels are free for the user to set up as they wish. Any of the runlevels can be modified as well. The runlevel of the system can be changed with the `init` and `telinit` commands. For example, to change to runlevel 2 you would use either of the following:

```
debian:~# init 2
```

or

```
debian:~# telinit 2
```



The `telinit` command is actually a link to `/sbin/init`, to provide for compatibility with other UNIX systems.

The `shutdown` and `reboot` commands are also used to change the runlevels. They take the system to runlevel 0, 1, or 6. Table 8-6 shows the command-line options for `shutdown`. The syntax for the `shutdown` command is as follows:

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

Table 8-6
shutdown Options

Option	Function
-a	Uses <code>/etc/shutdown.allow</code> , which lets nonroot users shut down the system.
-t <seconds>	Tells <code>init</code> to wait the specified number of seconds before sending the <code>warning</code> and <code>kill</code> commands to processes.
-k	Does not actually shut down the system, but only sends the warning.
-r	Reboots the system after shutdown.
-h	Halts the system after shutdown.
-f	Skips the <code>fsck</code> check on reboot.
-F	Forces a <code>fsck</code> check on reboot.
-c	Cancels a shutdown.
<time>	The time to shut the system down. This can be given in HH:MM format, +N for the number of minutes from now to execute, or the keyword <code>now</code> which means +0.
<warning message>	The warning message to send to all users.

The `halt`, `reboot`, and `poweroff` commands are used to stop, restart, or power off the system. The syntaxes for these commands are as follows:

```
/sbin/halt [-n] [-w] [-d] [-f] [-i] [-p]
/sbin/reboot [-n] [-w] [-d] [-f] [-i]
/sbin/poweroff [-n] [-w] [-d] [-f] [-i]
```

Table 8-7 lists the supported command-line options. The `reboot` and `poweroff` commands are actually symbolic links to `halt`. The functionality depends on the link called. If the system is not in runlevel 0 or 6 before these commands are called, the `shutdown` command will be used instead.

Table 8-7
halt/reboot/poweroff Options

Option	Function
-n	Do not sync the hard disks with the cache before rebooting or halting the system. This is not advised.
-w	Do not actually halt or reboot the system, only write the entry to <code>/var/log/wtmp</code> so it appears the system was halted or rebooted.
-d	Do not write the record to <code>/var/log/wtmp</code> .
-f	Force the halt or reboot without first calling <code>shutdown</code> .
-i	Shut down all network interfaces before doing the halt or reboot.
-p	Power off the system after halting.

Configuring the init process

After being started by LILO, the Linux kernel executes the first process, which is `init`. The `init` process always has the process ID of 1, as shown below.

```
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.2  1216   520 ?        S     Jan12   0:04 init [2]
```

All other processes on the system are child processes of `init`. When `init` is executed, it gets its configuration from the `/etc/inittab` file. Each entry in the `/etc/inittab` file has the following syntax:

```
id:runlevels:action:process
```

Lines that begin with `#` symbols are comments. Table 8-8 lists the function of each field in the entry. The possible entries for the action field are shown in Table 8-9.

Table 8-8
inittab Entry Fields

Field	Function
<code>id</code>	A string of one to four unique characters that identifies an entry in the <code>inittab</code> file.
<code>runlevels</code>	The list of runlevels for which the action specified in this entry should be taken.
<code>action</code>	Specifies the action to be taken.
<code>process</code>	The process to be executed.

Table 8-9
inittab action Entries

Action	Function
respawn	The process will be restarted whenever it is terminated. You will see this used with the <code>getty</code> login processes, which need to be regenerated each time.
wait	The process is started once, and then <code>init</code> will wait until it is terminated.
once	The process will be executed only once, when the runlevel is entered.
boot	The process will be executed at boot, and any runlevel field entries will be ignored.
bootwait	The process is started at boot, and <code>init</code> will wait for it to complete before continuing.
initdefault	Specifies the runlevel to enter when the system is booted.
sysinit	This entry will be executed at boot, before any other <code>boot</code> or <code>bootwait</code> entries. The runlevel field is ignored.
powerwait	This entry will be processed when the power to the system goes down, as reported by a UPS. <code>init</code> will wait for this to complete before continuing.
powerfail	The same as <code>powerwait</code> , but <code>init</code> does not wait for this process to complete.
powerokwait	This process is executed when the power has been restored to the system. <code>init</code> will wait for its completion.
powerfailnow	This process is executed when the UPS signals to the system that the battery is almost exhausted.
ctrlaltdel	This entry will be processed when a user presses the Ctrl-Alt-Delete key combination.
kbrequest	Execute the specified process when the defined special keyboard combination is pressed.

The following is an example of the `/etc/inittab` file with explanations.

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

Tells the system the default runlevel to start up in. As you can see, this system is probably a Red Hat system booting up in to multiuser text mode.



The `initdefault` entry defines the runlevel that the system will boot into.

```
# System initialization.
si::sysinit:/etc/rc.d/rc.sysinit
```

These lines run the `/etc/rc.d/rc.sysinit` script during system initialization. This script is executed before any others, and `init` will wait until it is completed to move on.

```
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
15:5:wait:/etc/rc.d/rc 5
16:6:wait:/etc/rc.d/rc 6
```

These lines tell `init` which command to run for each runlevel. For example, when the system is changed to runlevel 5, the command `/etc/rc.d/rc 5` is executed. The `/etc/rc.d/rc` script starts the system, and by passing it the parameter of 5, it starts in runlevel 5. `init` will also wait until the specified script is completed before continuing.

```
# Things to run in every runlevel.
ud::once:/sbin/update
```

This line causes the `/sbin/update` command to be executed once during system boot for all runlevels.

```
# Trap CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now
```

This line causes the system to execute the `/sbin/shutdown -t3 -r now` command whenever the Ctrl-Alt-Delete key combination is entered. This is useful for trapping the signal and gracefully rebooting the system.

```
pf::powerfail:/sbin/shutdown -f -h +2 "Power Failure; System
Shutting Down"
```

This line causes the system to shut down gracefully should the Uninterruptible Power Supply (UPS) signal that it has lost power.

```
pr:12345:powerokwait:/sbin/shutdown -c "Power Restored;
Shutdown Cancelled"
```

This line cancels the restart if the UPS detects that power has been restored.

```
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
```

The `/sbin/mingetty` processes are started for runlevels 2 through 5. These provide the standard login consoles and are respawned each time they are terminated.

```
x:5:respawn:/etc/X11/prefdm -nodaemon
```

This line starts the graphical login processes for runlevel 5.

Customizing the Boot Process

The process that `init` follows when it brings the system up is laid out exactly in the `/etc/inittab` file. But this file does not tell the whole story; it only launches other scripts that handle most of the tasks. Most Linux distributions use many different script files to start the services and processes on the system. There are two different types of startup procedures used today: BSD and Sys V.

BSD startup

A couple of Linux distributions use the older BSD style of initialization; with the most popular one being Slackware. With BSD initialization the system uses a few long `rc` (run control) script files to set up and configure the system. These are usually named similar to `/etc/rc` and `/etc/rc.local`. The benefit of this setup is that you know where to look for most things, and all of the information is stored in the same place. The problem with BSD-style initialization is that it is harder to quickly customize.



As the name implies, the BSD variants such as FreeBSD and OpenBSD use this type of startup.

Sys V startup

Red Hat, its variants, and Debian all use System V initialization, known as Sys V `init`. This style of startup uses many small script files in a hierarchical directory structure separated into runlevels. The root of this directory tree in Red Hat is `/etc/rc.d`, while in Debian the root is just in `/etc`.

Off of the root of the tree is a directory named `init.d`. Inside this directory are many different script files. Each of these scripts serves a specific purpose, whether it is to start a server, bring up the networking subsystem, or check files. These scripts can be used to manage the services on the system. To see which commands can be passed to a script, just execute the script with no parameters. For example:

```
[root@redhat init.d]# ./lpd
Usage: lpd {start|stop|restart|reload|status}
```



The startup scripts are covered in more detail in Chapter 11.

All scripts will take the `start`, `stop`, and `restart` commands at a minimum. These are used to gracefully manage the service and should be used instead of running the service executable manually. Many services provide the `status` command, which can show useful information, such as the following:

```
[root@redhat init.d]# ./network status
Configured devices:
lo eth0
Devices that are down:
```

```
Devices with modified configuration:
```

Off `/etc` or `/etc/rc.d` are directories for each runlevel named `rc0.d` through `rc6.d`. Inside the runlevel directories are link files to the scripts contained in the `init.d` directory. The names of the link files in the runlevel directories are not the same as the script they link to. The first letter is either an `S` or a `K`. When the system executes the scripts in the runlevel directory, it passes the `start` command to the scripts starting with `S`, and passes the `stop` command to scripts starting with `K`. This way you can limit which processes stay running when changing runlevels. After the first letter is a number that specifies the order the scripts are started. This way you can say that `S30Syslog` starts before `S50inet`. After the priority number is a name for the script, which is just descriptive information.

Since the runlevel directories contain only links to the scripts in `init.d`, it is very easy to change the services and functions that happen when a new runlevel is entered. If you want to remove a service, just delete the link file. If you want to add a new service to a runlevel, you just make a symbolic link to the script file in `init.d`, with an `S` as the first letter and a priority number. Also, since all the runlevels point to the same scripts, you need to make modifications to only one file to make a change.

Red Hat startup

When Red Hat starts up, the first script run by `init` is the `/etc/rc.d/rc.sysinit`. It is run by the following line in the `/etc/inittab` file.

```
si::sysinit:/etc/rc.d/rc.sysinit
```

This script sets up some of the most basic system configuration including host-name, time, and file systems and loads needed kernel modules. The following is a complete list of the tasks that this script performs.

1. Set the default system `PATH` variable.
2. Check to see if the `/etc/sysconfig/network` file exists, and if so, set up networking. If the file does not exist, disable networking.
3. Print the welcome banner shown at boot, and prompt the user to enter Interactive boot mode.
4. Enable logging with `dmesg`.
5. Mount the `/proc` file system.
6. Configure several kernel parameters by using the `sysctl` tool.
7. Set the system clock.
8. Load the default keymap.
9. Execute `/sbin/setsysfont` to configure the default system font.
10. Activate the swap partitions with the `swap -a` command.
11. Set the system's hostname.
12. Configure NIS, if used.
13. Check to see if `fsck` needs to be run on a volume, and if a serious error is detected, prompt the user to log in and perform the tests manually.
14. Enable quotas on the root file system.
15. Configure Plug-and-Play devices using `isapnp`.
16. The root file system is then remounted read/write.
17. Set up the `/etc/mtab` file, which lists mounted file systems.
18. Set up the `/etc/HOSTNAME` file.
19. Run `depmod -a` and set up the loadable kernel modules.
20. Load any sound modules configured.
21. Load any software RAID devices configured in `/etc/raidtab`.
22. Check other nonroot file systems with `fsck`.
23. Mount all other nonroot file systems.
24. Enable quotas on the other nonroot file systems.
25. Clean up various temporary files that are no longer needed.
26. Execute `/etc/rc.d/rc.serial` to configure the serial devices.
27. Load SCSI tape drive modules if detected.

28. Generate the header file named `/boot/kernel.h`.
29. Create the symbolic links for `/boot/System.map-kernel_version`.
30. Dump the boot information gathered in to `/var/log/dmesg`.

Once `init` has processed the `rc.sysinit` script, it will execute any other processes configured to run at boot and then enter the default runlevel. The runlevel is entered by running the `/etc/rc.d/rc` script with a parameter of the runlevel. For example, to enter runlevel 3, `init` would execute the following:

```
/etc/rc.d/rc 3
```

The `/etc/rc.d/rc` script then goes to the correct runlevel directory, in this example `/etc/rc.d/rc3.d`, and runs the specified script files. In runlevels 2, 3, and 5 the final script to run is `S99local`, which is a link to `/etc/rc.d/rc.local`. This script runs last and is used to do any tasks that don't really need a separate script in `init.d`. The default script performs the following tasks.

1. Gather system information in to several variables.
2. Create the `/etc/issue` file.
3. Create the `/etc/issue.net` file.

The `/etc/issue` file is displayed when a user logs in to the local console. For example:

```
Red Hat Linux release 6.2 (Zoot)
Kernel 2.2.16-3 on an i686
```

The `/etc/issue.net` file is displayed when a user logs in remotely over the network. By default it is just a copy of the `/etc/issue` file. Making a new version of this file that does not give out as much useful information to those who may want to break in to your system is a common practice.



If you make a new `/etc/issue.net` file, remember that it will be overwritten when the system is rebooted unless you modify the `/etc/rc.d/rc.local` script.

Red Hat provides a couple of tools to help manage the runlevel directories, without having to manually make link file changes. The first is a quick tool known as `chkconfig`. It takes its directives from the command line. The supported parameters are shown in Table 8-10. For example, to see which runlevels the `inet` service is started in, you would use the following:

```
[root@redhat rc.d]# chkconfig --list inet
inet      0:off  1:off  2:off  3:on   4:on   5:on   6:off
```

Table 8-10
chkconfig Options

<i>Option</i>	<i>Function</i>
<code>--level <levels></code>	A string of numbers that specifies which runlevels the requested operation effects. For example, <code>--level 35</code> would specify runlevels 3 and 5.
<code>--add <name></code>	Adds the service to the specified runlevels.
<code>--del <name></code>	Removes the service from the specified runlevels.
<code>--list <name></code>	Lists which runlevels do and do not start this service. If no name is specified all services are reported.

Another useful tool for changing runlevel information is `ntsysv`. It provides an easy-to-use text mode menu interface. By default it edits the current runlevel, but this can be overridden with the `--level` parameter. For example, to edit runlevel 5 you would use the following:

```
[root@europa rc.d]# ntsysv --level 5
```

Debian startup

Debian uses a similar startup routine to Red Hat and its variants, but with slightly different scripts and file placement. As we said before, Debian puts the runlevel directories `rc0.d` through `rc6.d` along with `init.d` off of `/etc`. Once `init` has been started by the kernel, it executes the first line in the `inittab`, which is the following:

```
si::sysinit:/etc/init.d/rcS
```

Instead of using a large script file to set up the base system, Debian uses the `/etc/init.d/rcS` script to read smaller scripts from `/etc/rcS.d`. The only other function of this script is to set the default path and `umask`. The scripts that run from `rcS.d` are shown below, in order.

1. `S05keymaps-1ct.sh`. This script loads the default key maps.
2. `S10checkroot.sh`. This checks the root file system's constancy and then remounts it as read/write. Finally, it builds the `/etc/mtab`.
3. `S15isapnp`. This script configures any ISA Plug-and-Play devices.
4. `S20modutils`. This executes `depmod -a` and runs `modprobe` to load all needed modules.
5. `S30checkfs.sh`. This script checks the other nonroot file systems.
6. `S30procps.sh`. This script configures the kernel using the `sysctl` tool.

7. `S30setserial`. This script configures the serial port devices on the system.
8. `S35devpts.sh`. This script sets up the `/dev/pts` devices.
9. `S35mountall.sh`. This script mounts all the other file systems specified in `/etc/fstab`.
10. `S39dns-clean`. This script cleans up some DNS items that may be left over from a PPP session.
11. `S39ifupdown`. This script removes the `/etc/network/ifstate` file.
12. `S40hostname.sh`. This script creates the `/etc/hostname` file.
13. `S40pump`. This script configures the network interfaces that use DHCP.
14. `S41portmap`. This script starts up the `rpc/portmap` services.
15. `S45mountnfs.sh`. This script checks the `/etc/fstab` file for NFS drives and then mounts them.
16. `S48console-screen.sh`. This script sets up some of the console defaults, including font, font map, and character set maps.
17. `S50hwclock.sh`. This script sets and adjusts the BIOS clock.
18. `S55bootmisc.sh`. This script performs several miscellaneous actions during boot, including the following: creates a `/etc/nologin` file so users do not log in before the system completes booting, clears the `/tmp` directory, clears up any locks in `/var/lock`, updates the `/etc/motd` file, and saves the kernel messages to `/var/log/dmesg`.
19. `S55urandom`. This script saves the random seed from `/dev/urandom` between reboots.
20. `S70nviboot`. This script recovers NVI editing sessions.
21. Any scripts in `/etc/rc.boot` are executed to maintain backward compatibility.

The system then moves on to run the scripts in the directory for the default run-level, as specified in `/etc/inittab`.

Troubleshooting the Boot Process

Objective

2.2 Linux Installation and Package Management

- **Install a boot manager.** Select, install and configure a boot loader at an appropriate disk location. Provide alternative and backup boot options (like a boot floppy disk). Involves using the command: `lilo`. Involves editing the file: `/etc/lilo.conf`.

One of the benefits of understanding the entire Linux boot process is that you can then begin to troubleshoot it. The three main problems that occur during boot are the following:

- ♦ Corrupt LILO install, or bad configuration
- ♦ Misconfigured new kernel installed
- ♦ Corrupted system files

The following sections cover the steps to go through to recover from a nonbooting Linux system.

Troubleshooting LILO

LILO displays error information when it is installed and when LILO loads at boot. The errors displayed when installing LILO are usually self-explanatory and provide enough information to fix the problem. On the other hand, the information displayed when LILO loads at boot can be cryptic. When LILO loads it displays a letter for each phase it goes through. The phases are explained in Table 8-11. Using this information you can narrow down the problem and resolve it.

The `-r` parameter for `lilo` is useful in troubleshooting situations. It changes the perceived root volume for LILO. If you cannot boot your system for some reason, you may boot from a repair disk or CD-ROM. If this is the case the root file system will be the repair media, not the normal root file system. In this case the `/etc` and `/boot` directories are not the actual ones you want to use. To fix this you need to use the `-r` option. For example, if you have booted to a repair disk and have the normal root file system mounted to `/mnt/oldroot`, you would use the following command to reinstall LILO.

```
debian:~# lilo -r /mnt/oldroot
```

This causes LILO to use the `/mnt/oldroot/etc` and `/mnt/oldroot/boot` directories.



The `-r` parameter for `lilo` is used in troubleshooting situations.

Table 8-11
LILO Error Codes

Letter	Meaning
<code><nothing></code>	No part of LILO was loaded.
<code>L<error code></code>	The first stage boot loader was loaded, but an error was encountered while loading the second stage. Table 8-12 lists the possible first stage error codes.

Continued

Table 8-11 (continued)

Letter	Meaning
LI	The first stage boot loader was able to load the second stage boot loader, but has failed to execute it. This can be caused either by a geometry mismatch or by moving <code>/boot/boot.b</code> without running the map installer.
LIL	The second stage boot loader has been started, but it can't load the descriptor table from the map file. This is typically caused by a media failure or by a geometry mismatch.
LIL?	The second stage boot loader has been loaded at an incorrect address. This is typically caused by a subtle geometry mismatch or by moving <code>/boot/boot.b</code> without running the map installer.
LIL-	The descriptor table is corrupt. This can be caused either by a geometry mismatch or by moving <code>/boot/map</code> without running the map installer.
LILO	LILO was loaded successfully.

Table 8-12
First Stage Error Codes

Error Code	Meaning
0x00	Internal error. This could be from a corrupted LILO install or from trying to access a cylinder over 1024 with the linear option.
0x01	Illegal command. This usually happens when a disk is accessed but is not supported by the BIOS.
0x02	Address mark not found. This is indicative of a media error. Try to boot the system again.
0x03	Write-protected disk.
0x04	Sector not found. This is usually caused by a geometry mismatch.
0x06	Change line active. This is a transient error, and you should try booting the system again.
0x07	Invalid initialization. The BIOS failed to initialize the disk controller.
0x08	DMA overrun. The LILO manual states this error "should not happen." The manual suggests booting again.

Error Code	Meaning
0x09	DMA attempt across 64K barrier. This is sometimes caused by a disk geometry mismatch. Try removing the compact entry from the <code>lilo.conf</code> file.
0x0C	Invalid media. This is caused by a media error.
0x10	CRC error. A bad spot on the media causes this error. Moving the boot files to other locations may fix it.
0x11	ECC correction successful. An error occurred while reading data, but it was corrected.
0x20	Controller error.
0x40	Seek failure. The disk media may be bad.
0x80	Disk timeout. The media may be bad, or the disk may not be spinning. If this is a floppy disk make sure the drive door is closed.
0xBB	BIOS error. If the problem persists after a reboot, try removing the compact entry. If that still does not fix it, try the LBA32 or linear settings.

Booting to single-user mode

If the system will not boot due to a hanging process or a corrupted driver, you can boot the system in single-user or emergency mode. This is done at the LILO command prompt. For example:

```
LIL0: linux S
```

This would boot the system in to single-user mode. In single-user mode the system is booted to runlevel 1, which provides basic support for a user logging in to the console.



Tip

You can also use the boot parameter `single` instead of `S` to boot to single-user mode.

The following boots the system in emergency mode:

```
LIL0: linux emergency
```

Emergency mode runs no scripts and puts the user directly in to the system. The root password is required for either option.

Creating a boot disk

If for some reason the kernel becomes corrupt, or you install a new kernel and it fails to boot, you can use a boot disk to gain access to the system. On Red Hat systems, boot disks are created using the `mkbootdisk` command. Table 8-13 lists the available command-line options for `mkbootdisk`.

Table 8-13
mkbootdisk Options

<i>Option</i>	<i>Function</i>
<code>--device <device file></code>	Specifies the device where the boot disk is to be created. If this option is omitted, <code>/dev/fd0</code> will be used.
<code>--mkinitrdargs <arguments></code>	Allows you to pass any needed arguments to the <code>mkinitrd</code> command that is used to create the <code>initrd</code> image for the boot floppy.
<code>--noprompt</code>	Disables the standard prompt for the user to insert a floppy disk.
<code>--verbose</code>	Enables verbose mode.
<code>--version</code>	Displays version information.

For example, to create a boot disk on the second floppy drive on the system with a disk already in the drive, you would use:

```
mkbootdisk --device /dev/fd1 --noprompt --verbose
```

Debian systems use the `mkboot` command to create a boot disk. To use the kernel `/boot/vmlinuz-2.4.0` as the boot image, you would enter:

```
mkboot /boot/vmlinuz-2.4.0
```

Creating repair disks

Creating a kernel boot disk is fine, as long as the root file system is intact and not corrupted so much that the system won't boot. What happens if the problems are worse? Most Linux distributions include repair disks to boot from that will enable you to fix this situation. The first disk is a *boot disk*, which contains a kernel image. The second disk is a *repair or root disk*, which contains the image of a basic root file system that gets expanded to a RAM disk. Once you boot to this new root file system, you can mount the old file systems and repair them.

Red Hat uses the images named `boot.img` and `rescue.img`, while Debian uses `rescue.bin` and `root.bin`. These images can be placed on a floppy using the `dd` command. For example:

```
dd if=<path>/<image name> of=/dev/fd0 bs=1024
```

Do this for the two images and then boot from them. Of course, it is best to make these disks ahead of time, but in a jam you can create them from another workstation. If you only have access to a Windows system, you can use the `rawrite.exe` utility instead of `dd`.

Once the system is booted in rescue mode you can mount the damaged file systems and repair them. The rescue disks include the basic commands you should need to edit files, check disk integrity, and move files.



Tip

Linuxcare provides a rescue CD-ROM image that is excellent! They also distribute the image on business card sized CD-ROMs that fit in your wallet. For more information see their site at <http://open-projects.linuxcare.com/BBC/>

Key Point Summary

The Linux boot process can be complicated, but once you understand the process it is easy to customize and change. Several methods to troubleshoot a nonbooting system exist, even if the root file system is corrupted.

- ♦ LILO is the most popular boot manager for Linux.
- ♦ LILO is usually installed to the master boot record (MBR) of a hard disk, or to the boot block of a partition.
- ♦ Kernel parameters, environment variables, and boot commands can be entered at the LILO boot prompt.
- ♦ LILO is configured via the `/etc/lilo.conf` file.
- ♦ Any time the `lilo.conf` file is changed you must rerun `lilo`.
- ♦ The `-r` option for `lilo` can be used to change the perceived root.
- ♦ Runlevels are used to give the Linux system multiple configurations.
- ♦ Runlevels 0 through 6 are used.
- ♦ Red Hat uses runlevel 3 for text mode and 6 for GUI mode.
- ♦ Debian uses runlevel 2 for text and GUI mode.
- ♦ Runlevel 0 halts the system, 1 is for single-user mode, and 6 reboots the system.

- ♦ The `init` and `telinit` tools are used to change the runlevel.
- ♦ `halt`, `poweroff`, and `reboot` are tools used to control the system.
- ♦ The Linux kernel starts the `init` process, which always has PID 1.
- ♦ `init` starts several other processes, which are configured in `/etc/inittab`.
- ♦ Red Hat, its variants, and Debian use Sys V startup.
- ♦ Slackware uses BSD-style startup.
- ♦ Individual script files are stored in the `init.d` directory.
- ♦ The first script executed in Red Hat is `rc.sysinit`, and in Debian it is `rcS`.
- ♦ `chkconfig` and `ntsysv` can be used to change the services started in different runlevels.
- ♦ LILO provides many predefined error codes it displays when a problem occurs at boot.
- ♦ To boot in single-user mode pass an `S` to the startup script at the LILO prompt.
- ♦ To boot in to emergency mode pass `emergency` to `init` at the LILO prompt.
- ♦ Boot disks are created using the `mkbootdisk` and `mkboot` commands.
- ♦ Most distributions supply repair and rescue disks for when the root partition is too corrupted to boot.



STUDY GUIDE

The following questions and exercises will allow you to review the information covered in this chapter. Take your time when completing this section of the chapter, carefully reviewing any questions that you may have. Simply answering the question correctly is not as important as understanding the answer, so review any material that you might still be unsure of. Being comfortable with the questions and answers presented here will help you be more prepared for the certification exam questions.

Assessment Questions

1. Which entry in `lilo.conf` restricts the user from adding boot parameters without a password?
 - A. `lock`
 - B. `restricted`
 - C. `noparams`
 - D. `password`
2. If Linux is the only operating system on your computer, where should you tell LILO to place the boot information?
 - A. MBR
 - B. Partition boot block
 - C. Floppy disk
 - D. BIOS
3. Which boot parameter would you use to have a different root file system mounted?
 - A. `root=/mnt/`
 - B. `root=/dev/hda1 /`
 - C. `root=/dev/hda1`
 - D. `root=/dev/hda`

4. Which `lilo.conf` entry gives a boot image a second name?
 - A. `alias`
 - B. `label`
 - C. `name`
 - D. `tag`

5. Which command will change a Red Hat system to GUI login mode?
 - A. `startx`
 - B. `runlevel 5`
 - C. `init 3`
 - D. `telinit 5`

6. To reboot the system right now with the `shutdown` command, you would enter the command _____.

7. The `reboot` and `poweroff` commands are actually symbolic links to _____.

8. Which line in `inittab` would you change to change the runlevel that the system boots up to?
 - A. `defaultinit`
 - B. `defrunlevel`
 - C. `runlevel`
 - D. `initdefault`

9. When a user Telnets into the system, the _____ file is displayed to him or her before the login prompt.

10. Which command would list the runlevel information about the `inetd` service?
 - A. `chkconfig --list inet`
 - B. `ntsysv --list inet`
 - C. `/etc/rc.d/init.d/inet status`
 - D. `ls -l /etc/rc.d/init.d/inet`

11. Which command would you use in a troubleshooting configuration if the old root volume is mounted under `/mnt/root`?
- A. `lilo -p /mnt/root`
 - B. `lilo -r /mnt/root/boot`
 - C. `lilo -r /mnt/root`
 - D. `lilo -p /mnt/root/boot`
12. To boot the system without running any startup scripts, you would use the boot parameter of _____.
13. To create a boot disk on the second floppy drive without prompting the user, you would use which command?
- A. `mkbootdisk --device /dev/fd1 --noprompt`
 - B. `mkboot --device /dev/fd1 --noprompt`
 - C. `mkbootdisk --device /dev/fd0`
 - D. `mkboot --device /dev/fd0`
14. The `lilo.conf` entry _____ is used to enable booting from disks with more than 1024 cylinders.
- A. `lba`
 - B. `lba32`
 - C. `bigdisk`
 - D. `linear`
15. When you reboot the system it first changes to runlevel _____.
16. Which process always has the PID of 1?
- A. `init`
 - B. `kernel`
 - C. `sync`
 - D. The first driver loaded
17. Which entry in `inittab` tells `init` to wait for the requested process to finish before continuing?
- A. `hold`
 - B. `pause`
 - C. `once`
 - D. `wait`

18. Which type of startup uses a few large script files?
 - A. UNIX
 - B. BSD
 - C. Sys V
 - D. Red Hat
19. To see information on the `inetd` process, you would enter the _____ command (no path).
20. What does LILO display when it cannot load the second stage of the boot loader?
 - A. LI
 - B. LIL
 - C. SECOND STAGE NOT FOUND!
 - D. LIL-

Scenarios

1. While you are booting your Linux workstation, it hangs while starting a daemon. What steps could you take to fix the situation?
2. You have a Linux server that currently boots up in to the X Window system. The graphical interface is using up needed resources on the system. How do you fix the problem?
3. After compiling and installing a new kernel, you get a kernel panic when rebooting. Unfortunately, you were so sure about the new kernel you did not keep the older one around. What can you do to fix this problem?

Lab Exercises

Lab 8-1 Checking boot services

Many Linux distributions install and start services that you do not need. To check your boot services, take the following steps.

1. First check the runlevel that your system boots to by default. This is done by examining the `/etc/inittab` file.

```
[root@rh7 /etc]# more inittab
#
# 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:3:initdefault:
```

2. As you can see, this system is set to boot to runlevel 3 by default. Next, see which services are set to start at boot.

```
[root@rh7 rc3.d]# ls
K01pppoe      S08ipchains  S16apmd      S45pcmcia    S80isdn      S95anacron
K20nfs        S10network   S20random    S55sshd      S80sendmail  S97rhnsd
K20rwhod      S12syslog    S25netfs     S56rawdevices S85gpm        S99linuxconf
K45arpwatch   S13portmap   S35identd    S60lpd        S90crond     S99local
S05kudzu      S14nfslock   S40atd       S75keytable  S90xfs
```

3. As you can see, since the script `S80sendmail` is in the `/etc/rc3.d` directory, the service will start. (If your system does not start `sendmail`, pick another script, such as `gpm` or `crond`.) To confirm the service, you type: `[root@rh7 rc3.d]# ps auxw | grep sendmail`

```
root          532  0.0  0.7  3232  336 ?        S      Mar14
0:01 sendmail: accepting connections
```

4. To set this service not to start at the next boot you can either remove the script link from the directory or just change its name to something that does not start with S or K. For example:

```
[root@rh7 rc3.d]# mv S80sendmail H80sendmail
```

5. Reboot the system and check to make sure the service isn't running.

Answers to Chapter Questions

Chapter Pre-Test

1. The `init` process is started by the kernel and has the PID of 1.
2. Sys V
3. `init`, `telinit`, `shutdown`, and `halt`
4. `/etc/inittab`
5. `/etc/issue.net` is displayed to network logins.
6. Pass the emergency boot parameter at the LILO prompt.
7. The media may be bad or there may be a disk geometry translation problem.
8. `ram=`
9. `/etc/lilo.conf`
10. 6

Assessment Questions

1. **B.** The `restricted` parameter stops someone at the console from issuing boot parameters without knowing the preset password. The `password` option is used to specify the password to enter. The other options are invalid. For more information see the “Configuring LILO” section.
2. **A.** The LILO boot sector should be placed in the master boot record of the drive instead of a partition. The partition boot block can be used when multiple operating systems are installed. For more information see the “Using LILO” section.
3. **C.** The `root=` parameter tells the system which volume to use as the root volume. The device name should point to a partition. No mount point is needed. For more information see the “Using LILO” section.
4. **A.** Use the `alias` entry to have a second name for your boot image. The label entry gives the boot entry a name that is displayed at the LILO menu. For more information see the “Configuring LILO” section.
5. **D.** The command `telinit 5` command changes the system to runlevel 5, which is the GUI mode in Red Hat. The `startx` command is used to start the X Window System from a command line, but not at boot. The `init 3` command would change the system to multiuser text mode. Choice B is invalid. For more information see the “Using runlevels” section.

6. `shutdown -r now`. The `-r` option tells the system to reboot, and the `now` option causes the system to do it immediately. For more information see the “Using runlevels” section.
7. `/sbin/halt`. For more information see the “Using runlevels” section.
8. **D**. The other entries are invalid. For more information see the “Configuring the init process” section.
9. `/etc/issue.net`. Local users will receive the `/etc/issue` file. For more information see the “Customizing the Boot Process” section.
10. **A**. The `chkconfig` tool is used from the command line while `ntsysv` provides a menu interface. The other commands do not provide runlevel information. For more information see the “Customizing the Boot Process” section.
11. **C**. The `-r` parameter changes the perceived root path. It should point to the root volume. For more information see the “Troubleshooting LILO” section.
12. `emergency`. The `S` parameter still runs the system initialization script. For more information see the “Booting to single-user mode” section.
13. **A**. The `--device` parameter specifies the floppy drive, which starts with number 0. The `mkboot` tool does not accept these parameters. For more information see the “Creating a boot disk” section.
14. **B**. The `linear` option is required for some SCSI disks. The other options are invalid. For more information see the “Configuring LILO” section.
15. **6**. Runlevel 0 halts the system, and runlevel 1 is used for single-user mode. For more information see the “Using runlevels” section.
16. **A**. The `init` process is the only process that has a runlevel of 1. For more information see the “Configuring the init process” section.
17. **D**. The `once` parameter tells `init` to start the process once and not to restart it again. The other options are invalid. For more information see the “Configuring the init process” section.
18. **B**. The BSD style of initialization uses a few large script files, while Sys V uses many smaller ones in a directory hierarchy. For more information see the “Customizing the Boot Process” section.
19. `inet status`. The `inet` startup script in `/etc/init.d` or `/etc/rc.d/init.d` can be queried for the process status. For more information see the “Sys V startup” section.
20. **A**. The `LIL` error is caused by a media failure or geometry mismatch. The `LIL-` is caused by a corrupt descriptor table. For more information see the “Troubleshooting the Boot Process” section.

Scenarios

1. You could boot the system in single-user or emergency mode. Find the script that starts the daemon that is hanging and either troubleshoot it or just remove it from that runlevel.
2. Change the system to boot to text mode by editing the `/etc/inittab` file. To unload X and change to text mode without rebooting the system, you could execute `telinit 3`.
3. If you have already made boot disks, you could use those to boot the system. If not, you can create boot disks on another Linux system or download boot and rescue images from your distribution's site.

Using X

9

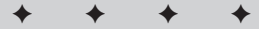
CHAPTER

EXAM OBJECTIVES

Exam 102 ♦ General Linux, Part 2

2.10 X

- **Install & Configure XFree86.** Verify that the video card and monitor are supported by an X server, install the correct X server, configure the X server, install an X font server, install required fonts for X (may require a manual edit of `/etc/X11/XF86Config` in the “Files” section), customize and tune X for videocard and monitor. Commands: `XF86Setup`, `xf86config`. Files: `/etc/X11/XF86Config`, `.xresources`.
- **Setup XDM.** Turn `xdm` on and off, change the `xdm` greeting, change default bitplanes for `xdm`, set-up `xdm` for use by X-stations
- **Identify and terminate runaway X applications.** Identify and kill X applications that won't die after user ends an X-session. Example: `netscape`, `tkrat`, etc.
- **Install & Customize a Window Manager Environment.** Select and customize a system-wide default window manager and/or desktop environment, demonstrate an understanding of customization procedures for window manager menus, configure menus for the window manager, select and configure the desired x-terminal (`xterm`, `rxvt`, `aterm` etc.), verify and resolve library dependency issues for X applications, export an X-display to a client workstation. Commands: Files: `.xinitrc`, `.Xdefaults`, various `.rc` files.



CHAPTER PRE-TEST

1. Which command starts the X Window System from the command line?
2. Which file is used to configure the X server?
3. Which executable file does the `startx` script run?
4. What piece of software manages window borders and desktop icons?
5. Which section in the `XF86Config` file configures fonts?
6. Which environment variable controls where an X application is displayed?
7. Which file is used to customize a user's X applications?
8. Which service is used to allow users to log in at a graphical prompt?
9. Which file in a user's home directory is executed by `xinit`?
10. What is the oldest graphical terminal emulator?

The XFree86 system (also called X Window or X) was designed to give Linux and UNIX systems an easy-to-use graphical interface. With more people moving to Linux from Windows and Macintosh, the need for X is greater than ever. Users from the other systems will be amazed at the level of customization and amount of choices available.

Overview of the X Window System

Unlike other operating systems such as Windows 2000 and MacOS, the X Window System is not an integrated part of Linux. It is a separate application that runs on top of the base operating system. This allows you to run a GUI interface if you want, but to free up memory and CPU resources if they are needed elsewhere.

History of X

X was originally developed at the Massachusetts Institute of Technology and released in 1984. It was derived from the project known as Athena, which was a group effort between MIT, IBM, and Digital Equipment to design a graphical environment for education. The goal was to provide a graphical networked environment that was standardized so multiple vendors' equipment could work together.

The lead for the main development project was Robert Scheifler. X also owes many of its early ideas to the “W” windows package, developed by Paul Asente at Stanford. In 1987 MIT released the first version of X that we know today, known as X11. As of X11R2 (X11 Release 2) the control of X passed from MIT to the X Consortium, which was formed in 1988. The current release of X11 is X11R6.5.1.

The version of X that is used with Linux is XFree86. The name comes from the fact that originally the package was meant for the Intel x86 processor, but it now runs on many different platforms and systems. More information on this project is available at <http://www.xfree86.org>.

Architecture overview

The X Window System was designed to have a very powerful and flexible client-server architecture. X applications can run on the user's local workstation or from a large network server with a fast processor. The X protocol handles the remote and local communications. The biggest module is the X server, which is responsible for displaying the applications on the user's system. X servers exist for almost every graphical computing platform used today. Applications may be run on a different computing platform and displayed back over the network to an X server running on a different platform.

Some people quickly get confused between client and server when discussing the X Window System, and we can't blame them. Most people consider their workstations to be the client, but in the case of X, their workstations are actually the servers. The X server provides the graphical interface on the workstation, and clients, in the form of applications, connect to the X server from either the local workstation or from across the network and get displayed.



The X server runs on the system with the monitor, while the clients (applications) can run remotely or locally. They connect to the X server software for display.

Window managers

The X server just allows client applications to connect and be displayed. It does not provide any mechanism for controlling the windows or customizing the display. This is where a *window manager* comes in. The window manager provides functionality such as window borders, menus, icons, virtual desktops, toolbars, and wallpaper control. Basically, the window manager provides the look and feel of your X desktop. Once you get the window manager that you want to use, the next step is to get a *theme*. The desktop theme specifies the detailed look and feel of the window manager. Normally a theme sets the color, window buttons, task bars, and possibly wallpapers. A popular site for window manager themes is <http://www.themes.org>.

The following is a list of the most popular window managers.

- ♦ **AfterStep**—AfterStep is available from <http://www.afterstep.org/>. It is based on the look and feel of the NeXTStep interface, which is the desktop for the NeXT computer. It has undergone a number of enhancements based on user feedback. Another benefit is the small memory size, advertised at less than 1MB.
- ♦ **Blackbox**—Blackbox is available from <http://blackbox.alug.org/>. It offers a small code base written in C++. It has a fast and simple interface that uses little resources. It is a very popular window manager due to its small size and speed.
- ♦ **Enlightenment**—Enlightenment is available from <http://www.enlightenment.org>. Often called E for short, Enlightenment is a very customizable window manager with excellent look and feel. The down side is that it can be resource intensive and usually runs best on higher-end systems; however, it can be worth it. E used to be the default window manager with GNOME (covered in the next section).
- ♦ **FVWM**—FVWM is available from <http://www.fvwm.org>. Most window managers today have their roots in the older FVWM. No one is sure what the F stands for, but the rest stands for Virtual Window Manager. FVWM is still being developed and worked on, but is not nearly as popular as it once was.

- ♦ **IceWM**—IceWM is available from <http://www.icewm.org>. It is a nice window manager designed for speed, usability, and consistency. It is able to emulate the look of Motif, OS/2, and Windows and allows you to have a customizable look using pixmaps.
- ♦ **Sawfish**—Sawfish is available from <http://sawmill.sourceforge.net/>. It is an extensible window manager using a LISP-based scripting language. All window decorations are configurable and all user-interface policy is controlled through the extension language. Sawfish is now the default window manager used with GNOME.
- ♦ **Window Maker**—Window Maker is available from <http://www.windowmaker.org/>. It is an X11 window manager designed to emulate the look and feel of the NeXTStep GUI. It is relatively fast, feature rich, and easy to configure and use.

Desktop environments

Window managers provide a good look and feel for your desktop, but until recently there were no real rules on getting applications to look and work the same. However, the new desktop environments coming on to the computing scene are changing this. A desktop environment includes a suite of applications, file manager, control panel, window manager, and many other components. They are designed to work together with the same look and feel. The two most popular desktop environments are KDE and GNOME. Every new user to Linux owes it to himself or herself to try out both environments and decide which he or she likes best.

KDE

KDE is available from <http://www.kde.org>. The first large-scale desktop environment for Linux was KDE. It is a mature project that offers an easy-to-use environment bundled with many applications, a window manager, and an easy-to-use control panel. KDE is built around the Qt libraries from Trolltech. Developers can write other applications using these libraries and have them fit together and work very well with KDE. The Qt libraries were once under a restrictive license that caused it some acceptance problems, but that situation has recently changed.

GNOME

The Ximian GNOME distribution is available from <http://www.ximian.com>. GNOME, GNU Network Object Model Environment, was originally started as an alternative to KDE, since the Qt libraries that KDE are based on were not under a free license. GNOME was originally an independent project, but a company named Helix Code was started by the core developers as a way to promote and fund the project. Helix Code has since changed their name to Ximian. They provide an excellent GNOME distribution that includes many applications, the sawfish window manager, a control panel, and many desktop themes. It also includes an easy-to-use updating tool that lets you keep your system current by automatically downloading new updates.

While you can get GNOME without the Ximian packaging enhancements, we recommend that you include the enhancements because of the extra tools and installer provided.

Installing X

Objective**2.10 X**

- **Install & Configure XFree86.** Verify that the video card and monitor are supported by an X server, install the correct X server, configure the X server, install an X font server, install required fonts for X (may require a manual edit of `/etc/X11/XF86Config` in the “Files” section), customize and tune X for videocard and monitor. Commands: `XF86Setup`, `xf86config`. Files: `/etc/X11/XF86Config`, `.xresources`.

The easiest way to install X is to do it during the main installation of the distribution. This way all of the needed packages are installed and configured up front. But, if you install a system without X and later decide to add it, you can do that, too. The installation method depends on the packaging system used by your distributions.

Installing with RPMs

If your system uses RPMs, it is recommended to install the necessary RPM packages from the distribution CD-ROM or site. This is because most distributions tweak and customize the X Window install. Some distributions add special user tools, such as automatic menu creation.

Installing on Debian

Debian users can easily add X to a system by typing either of the following:

```
apt-get install task-x-window-system-core
```

or

```
apt-get install task-x-window-system
```

These tasks will download all needed packages, install them, and walk you through the configuration of X. The core task downloads only the basic X window System, while the system task downloads X as well as a window manager and other supporting applications.

Installing with binary packages

The XFree86 group provides binary tar files with an installation script for those users who do not want or cannot use another packaging system. These packages are available from their site at <http://www.xfree86.org>. A README file is provided with instructions on installation. The steps below will walk you through installing XFree86 with the tar files. Be sure to change into a temporary directory first. In the example that follows you are in the /root/X directory.

**Tip**

The Release Notes distributed with every version of XFree86 tell you which hardware devices are supported.

1. FTP to `ftp.xfree86.org` and log in as anonymous.

```
[root@redhat ~/X]# ftp ftp.xfree86.org
Connected to ftp.xfree86.org.
220 public.xfree86.org FTP server (Version wu-2.6.1(1) Tue
Jan 2 12:12:36 PST 20 01) ready.
Name (ftp.xfree86.org:root): anonymous
331 Guest login ok, send your complete e-mail address as
password.
Password:
230-
230- Welcome to ftp.xfree86.org, the master server for
XFree86 distribution.
230-
```

2. Change to the `/pub/XFree86/3.3.6/binaries/Linux-ix86-glibc21` directory.

3. Change the FTP client to binary mode and turn off prompting.

```
ftp> binary
200 Type set to I.
ftp> prompt
Interactive mode off.
```

4. Download all of the files in this directory by typing `mget all`.

5. Change to the Servers directory by typing `cd servers`.

6. List the available servers by typing `ls`.

7. Choose the correct server for your video card and download it using `get server_filename`.

8. Exit out of the FTP client.

9. Make the two install scripts executable by typing `chmod +x *.sh`.

10. Create a directory named `/usr/X11R6` by typing `mkdir /usr/X11R6`.

11. Change into the `/usr/X11R6` by typing `cd /usr/X11R6`.

12. Execute the `preinst.sh` script from the temporary directory.
13. Make the `extract` script executable. For example:

```
[root@redhat /usr/X11R6]# chmod +x /root/X/extract
```
14. Execute the `extract` script with a parameter of directory with the downloaded files.

```
[root@redhat /usr/X11R6]# /root/X/extract /root/X/*.tgz
```
15. Execute the `postinst.sh` script.

```
[root@redhat /usr/X11R6]# /root/X/postinst.sh
```
16. Answer “yes” to the prompted link question.
17. The final step is to run a configuration tool to create the needed `XF86Config` file.



The `XF86Config` configuration tools are discussed later in this chapter in the “Configuring X” section.

Versions of XFree86



2.10 X

- **Install & Configure XFree86.** Verify that the video card and monitor are supported by an X server, install the correct X server, configure the X server, install an X font server, install required fonts for X (may require a manual edit of `/etc/X11/XF86Config` in the “Files” section), customize and tune X for videocard and monitor. Commands: `XF86Setup`, `xf86config`. Files: `/etc/X11/XF86Config`, `.xresources`.

There are two major versions of XFree86 being used today. Some distributions are still shipping XFree v3 while others have moved up to the new XFree v4. The LPI exam was written with XFree v3 in mind, so it will be the focus here, but we try to mention important differences to XFree v4.

The biggest changes that users will notice between the two versions are the following:

- ♦ Consolidated X server
- ♦ Slightly different `XF86Config` format
- ♦ Speed

With XFree v3 the X servers were separated by video chipset. You had to make sure to install the correct server for your hardware. Table 9-1 lists the available X

servers. If you install X during the distribution setup, the setup program will usually detect the correct video chipset and install the right server. If you install X after the initial setup, you will have to manually install the correct X server. For example, to install the S3 server on a Red Hat system, you would install the XFree86-S3-3.3.6-33.i386.rpm package.

Table 9-1
XFree v3 X Servers

<i>X Server</i>	<i>Chipset Supported</i>
X3DL	Server for 3Dlabs video cards.
X8514	Server for 8514-based video cards.
XAGX	Server for AGX-based video cards.
XFB	Frame buffer driver, which can be used for some cards that do not have a specific X server.
XI128	Server for Number Nine Imagine 128 video cards.
XMach32	Server for ATI Mach 32 video cards.
XMach64	Server for ATI Mach 64 video cards.
XMach8	Server for ATI Mach 8 video cards.
XMono	Server for monochrome video displays.
XP9K	Server for P9000-based video cards.
XS3	Server for S3-based video cards.
XS3V	Server for S3 Virge-based video cards.
XSVGA	Server for VESA Super VGA video cards.
XVGA16	VGA 16-color server. Many cards that do not have their own X server can use this.
XW32	Server for the ET4000/W32-based video cards.



When installing XFree v3 you need to install the correct X server for your video chipset.

Starting with XFree v4 the X server is now consolidated. Instead of installing a specific server for your video chipset, you need to install only the main server and specify which chipset driver to use in the X configuration file. The downside is that currently XFree v4 does not support all of the video chipsets that XFree v3 supports.

In either version of X, the file `/etc/X11/X` is actually a link to the X server you want to run. Consider this example:

```
[root@redhat X11]# ls -l X
lrwxrwxrwx 1 root root 29 Aug 21 15:32 X ->
../../usr/X11R6/bin/XF86_SVGA
```

This shows the `/etc/X11/X` file is linked to `/usr/X11R6/bin/XF86_SVGA`, which is the SVGA X server. Since XFree v4 does not have separate X servers, the X file is linked to the same server, as shown here:

```
debian:/etc/X11# ls -l X
lrwxrwxrwx 1 root root 20 Dec 11 10:55 X ->
/usr/bin/X11/XFree86
```

The file `/usr/bin/X11/XFree86` is the X server for v4 and will always be the destination of the link.

The main configuration file for the X server is the `XF86Config` file. The format for this file is similar between XFree v3 and v4, but some slight differences exist. If you use one of the provided tools to create this file, you do not need to worry about the differences.

One of the best enhancements in XFree v4 is speed. XFree v4 noticeably increases performance over XFree v3. The new version also supports greater hardware acceleration for games and graphic applications.

Configuring X

Objective

2.10 X

- **Install & Configure XFree86.** Verify that the video card and monitor are supported by an X server, install the correct X server, configure the X server, install an X font server, install required fonts for X (may require a manual edit of `/etc/X11/XF86Config` in the “Files” section), customize and tune X for videocard and monitor. Commands: `XF86Setup`, `xf86config`. Files: `/etc/X11/XF86Config`, `.xresources`.

Configuring the X Window System can stop a new Linux user in their tracks. It can turn into a very complicated problem with odd error messages and strange results. Since most distributions now do an excellent job of detecting the hardware and monitor on a system, users do not need to go through all of the procedures that used to be required. But, if you install X manually or have an unusual setup, you may still be faced with configuring X yourself. The two configuration tools covered here can be used with any distribution. Further, each distribution commonly supplies its own distribution-specific tool. For example, Red Hat uses `Xconfigurator`, Mandrake uses `DrakX`, and Debian uses `anXious` and `DaX`.



The configuration file for the X server is `XF86Config`.

Manually configuring the XF86Config file

The main X Window configuration file is `/etc/X11/XF86Config`. While you are not required to build this file by hand any more, you should still be familiar with its layout in case there is a problem or you need to make a simple change. The file is broken up into several sections that have the following syntax:

```
Section "Section Name"
    This is a command
EndSection
```



Caution

XFree86 v4 uses a configuration file named `XF86Config-4`.

The following sections are used in the `XF86Config` file, and usually in this order:

- ♦ Files
- ♦ ServerFlags
- ♦ Keyboard
- ♦ Pointer
- ♦ Monitor
- ♦ Device
- ♦ Screen

Files

The `Files` section is used for RGB (red-green-blue) settings and font paths. Multiple font paths can be specified and will all be searched. If a useable font cannot be found, the X server will crash. The following is an example of the `Files` section.



Even if you use a font server for all of your fonts, you usually need a path to the `misc` font directory. This path provides the basic system fonts.

```
# *****
# Files section. This allows default font and rgb paths to be set
# *****

Section "Files"

# 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.

    RgbPath      "/usr/X11R6/lib/X11/rgb"

# 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.

    FontPath     "/usr/X11R6/lib/X11/fonts/TrueType"
    FontPath     "unix/:-1"

EndSection
```

This example uses two font paths. The second path defines the font server:

```
FontPath      "unix/:-1"
```

A font server can be run either on the local system or across the network. It can provide fonts for many systems at one time. The line specified above dictates the use of a local font server, since the port number is `-1`.

The only other possible entry in this section is `ModulePath`, which is used to load dynamic modules. This is used extensively in XFree v4.



The most important part of the `Files` section to know for the exam is the font configuration.

ServerFlags

This section sets several server settings. The possible entries for this section are shown in Table 9-2. Many times this section is blank or totally commented out. As you can see, several miscellaneous settings go here.

Table 9-2
ServerFlags Entries

<i>Entry</i>	<i>Function</i>
<code>NoTrapSignals</code>	Tells the X server to not unexpectedly trap signals and exit cleanly.
<code>DontZap</code>	Disables the use of the Ctrl-Alt-Backspace key combination to kill the X server.
<code>DontZoom</code>	Disables the use of the Ctrl-Alt-Keypad Plus and Ctrl-Alt-Keypad Minus, which normally changes the screen resolutions and acts like a zoom.

Entry	Function
AllowNonLocalXvidtune	Allows the <code>xvidtune</code> client to connect from a remote system. The <code>xvidtune</code> client is discussed in more detail later in the chapter.
DisableVidMode	Disables the parts of the <code>VidMode</code> extensions used by <code>xvidtune</code> .
AllowNonLocalModInDev	Lets a user from a remote host connect and change the keyboard and mouse settings in the running environment.
DisableModInDev	Disables the extensions that allow remote changing of the input settings.
AllowMouseOpenFail	Normally the X server will not start if the mouse device cannot be opened. This lets the X server start in that situation.

Keyboard

As you would expect, this section configures the keyboard device. It has many different configuration entries that deal with the keymap settings. Meta keys can also be defined here. Meta keys are those keys that change the function of the next key pressed. For example, `Ctrl` and `Alt` are normally meta keys. The following is an example of this section.

Section "Keyboard"

```

Protocol      "Standard"

# when using XQUEUE, comment out the above line, and uncomment the
# following line
#Protocol     "Xqueue"

AutoRepeat   500 5

# Let the server do the NumLock processing.  This should only be
# required when using pre-R6 clients
#ServerNumLock

# Specify which keyboard LEDs can be user-controlled (eg, with xset(1))
#Xleds       1 2 3

#To set the LeftAlt to Meta, RightAlt key to ModeShift,
#RightCtl key to Compose, and ScrollLock key to ModeLock:

LeftAlt      Meta
RightAlt     Meta

```



```

ScrollLock      Compose
RightCtl       Control
XkbKeycodes    "xfree86"
XkbTypes       "default"
XkbCompat      "default"
XkbSymbols     "us(pc101)"
XkbGeometry    "pc"
XkbRules       "xfree86"
XkbModel       "pc101"
XkbLayout      "us"
EndSection

```

Pointer

This section configures the pointing device. If this is not configured correctly, the X server may not start, or the mouse may act erratically. Table 9-3 shows the available configuration entries for this section. The following is an example:

```

# *****
# Pointer section
# *****

Section "Pointer"
    Protocol      "IMPS/2"
    Device        "/dev/mouse"
    ZAxisMapping  4 5
EndSection

```

Table 9-3
Pointer Section Entries

Entry	Function
Protocol " <i>Protocol Name</i> "	Specifies the protocol that the mouse device uses. Check the <code>XF86Config</code> man page for the complete list of available protocols.
Device " <i>Device Path</i> "	Specifies the device to use to communicate with the mouse. For example, <code>/dev/mouse</code> .
Port " <i>Device Path</i> "	This is an alternative to the <code>Device</code> entry.
BaudRate <i><baud rate></i>	Sets the baud rate for a serial mouse.
Buttons <i>N</i>	Specifies the pointing device has <i>N</i> number of buttons.
Emulate3Buttons	Allows a two-button mouse to emulate a three-button mouse by registering the third button when both buttons are pressed.

Entry	Function
<code>ChordMiddle</code>	Tells the X server that the mouse sends left and right clicks when the middle button is pressed. Common with Logitech mice.
<code>SampleRate <rate></code>	This defines how many updates per second the mouse sends.
<code>Resolution <count></code>	Sets the resolution supported by the mouse in counts per inch.
<code>ClearDTR</code>	Causes the DTR (Data Terminal Ready) line to be cleared when using a serial mouse.
<code>ClearRTS</code>	Causes the RTS (Ready To Send) line to be cleared when using a serial mouse.
<code>ZaxisMapping N M</code>	Used with wheel mice. Maps button <i>N</i> to negative movement, and button <i>M</i> to positive movement.

Monitor

The `Monitor` section defines the specifications for the monitor and lists the possible display modes. Table 9-4 shows the possible entries in this section. The most complicated entries in this section are the `mode` lines that specify the refresh and sync rates for your monitor. The following is an example:

```
ModeLine "1024x768i" 45 1024 1048 1208 1264 768 776 784 817 Interlace
Mode "1024x768i"
    DotClock      45
    HTimings      1024 1048 1208 1264
    VTimings      768 776 784 817
    Flags         "Interlace"
EndMode
```

The two entries are equivalent. It is best to let a configuration tool create the correct mode lines for your monitor, or you risk damaging it.



Tip

Many sites on the Internet have `mode` lines for popular monitors. Before trying to create your own, do a quick search to see if someone has already done the work for you.

Table 9-4
Monitor Section Entries

Entry	Function
Identifier " <i>ID String</i> "	A user-specified string that will be referenced later in the Screen section. Each Monitor section should have a unique Identifier string.
VendorName " <i>Vendor</i> "	An optional entry that specifies the monitor's vendor.
ModelName " <i>Model</i> "	An optional entry that specifies the monitor's model.
HorizSync < <i>horizontal sync range</i> >	A comma-separated list of discrete values that define the horizontal sync range supported by the monitor.
VertRefresh < <i>vertical refresh range</i> >	A comma-separated list of discrete values that define the vertical refresh rates supported by the monitor.
Gamma < <i>gamma value(s)</i> >	An optional entry that can be used to change the gamma of your monitor. Not all X servers support it. It is specified as either a single value or three separate RGB values.
Mode " <i>name</i> "	Starts a multiline video mode definition.
Modeline " <i>name</i> " < <i>mode description</i> >	Specifies a single line video mode definition.

Device

The Device section defines the video card(s) in your system. Multiple Device sections are allowed in the XF86Config file. The following is an example of this section:

```
Section "Device"
    Identifier      "Nvidia TNT2"
    VendorName     "Nvidia"
    BoardName      "TNT2"
    Chipset        "svga"
EndSection
```

Table 9-5 lists the possible entries for this section. Some other options that are only used by a few X servers do exist, and these entries are listed in the man page for the XF86Config file. Modern PCI and AGP video cards require very few entries in this field, as they can be probed by the X server for the correct information. If you use an old video card, you may need to specify all of the information manually.



Almost all PCI and AGP video cards can be probed by the X server for the needed hardware information.

Table 9-5
Device Section Entries

Entry	Function
Identifier " <i>ID String</i> "	Specifies a string that will be referenced later in the Screen section.
VendorName " <i>Vendor</i> "	An optional entry that specifies the video card vendor's name.
BoardName " <i>Model</i> "	An optional entry that specifies the video card's model.
Chipset " <i>Chipset Type</i> "	An optional entry that specifies the chipset type on the video board. If this entry is omitted, the X server will probe the card for the type.
Ramdac " <i>RAMDAC Type</i> "	An optional entry that specifies the type of RAMDAC installed on the video board. If this entry is omitted, the X server will probe for type.
DacSpeed < <i>speed</i> >	An optional entry that specifies the RAMDAC speed in MHz. This is usually either printed on the chip or in the video card's documentation.
Clocks < <i>clock</i> > ...	An optional entry that specifies the clock entries used by some video cards. This is only for older cards, and if a programmable clockchip is used, you should not use this entry.
ClockChip " <i>ClockChip Type</i> "	An optional entry that specifies the programmable clockchip type. Again, this is used only on older video cards.
Option " <i>Option String</i> "	Some X servers and drivers allow the user to define special settings. This entry is used to pass these settings to the X server or driver.
VideoRam < <i>memory</i> >	Sets the amount of memory installed on the video card in K.
BIOSBase < <i>Base Address</i> >	An optional entry that specifies the base BIOS address for the video card. The default is 0xC0000. If your video card uses an alternate address, specify it here.
MemBase < <i>Base Address</i> >	This entry is used differently by different X servers. It is used to specify the video card's base memory address. Check the X server documentation for your card for more information.
IOBase < <i>Base Address</i> >	Specifies the I/O base memory address. It is used by only a few X servers.

Screen

The XF86Config file lets you specify multiple video cards and monitors. The Screen section is used to match up the different video cards to the monitor definitions. Table 9-6 shows the possible entries for this section. The following is an example:

```
Section "Screen"
    Driver      "accel"
    Device      "My Video Card"
    Monitor     "Gateway EV700"
    Subsection "Display"
        Depth      32
        Modes      "1152x864"
        ViewPort   0 0
    EndSubsection
EndSection
```



If you receive an error when starting X saying that no screens are useable, make sure the specified monitor is capable of using the requested resolution with the defined video card.

Table 9-6
Screen Section Entries

Entry	Function
Driver " <i>Driver Name</i> "	Each Screen section begins with this entry. It tells the system which X server, or which driver in an X server, to load for this screen.
Device " <i>Device ID</i> "	Specifies which video card identifier string to use for this screen. This is how the video card is matched up to the monitor.
Monitor " <i>Monitor ID</i> "	Specifies which monitor identifier string to use for this screen. This is how the monitor is matched up to the video card.
DefaultColorDepth < <i>bpp number</i> >	Specifies the color depth to use for this screen, in bpp (bits per pixel).
ScreenNo < <i>Screen Number</i> >	Used in a multimonitor configuration to override the default screen numbering.
BlankTime < <i>Time</i> >	Specifies the amount of idle time to wait before blanking the screen, in minutes. The default is 10.
StandByTime < <i>Time</i> >	Specifies the timeout for the monitor to enter Stand By power-saving mode, in minutes. The default is 20.

Entry	Function
SuspendTime <Time>	Specifies the timeout for the monitor to enter Suspend power-saving mode, in minutes. The default is 30.
OffTime <Time>	Specifies the timeout for the monitor to power off, in minutes. The default is 40.
SubSection "Display"	Begins a block that configures display specific information. Table 9-7 shows the entries available in this subsection.

Table 9-7
Display Subsection Entries

Entry	Function
Depth <bpp>	Specifies the default color depth that the X server will run.
Weight <RGB>	An optional entry that specifies the RGB weight to use when in 16 bpp (bits per pixel) mode.
Virtual <x dimensions> <y dimensions>	Defines the virtual desktop dimensions. This is used to give the user a larger desktop than his or her monitor can handle.
ViewPort <x0> <y0>	Sets the starting location of the upper left corner when a virtual desktop is used. If this entry is omitted, the display is centered in the virtual desktop.
Modes "Mode Name" ...	Specifies a list of mode entries to use for this display.
Option "Option String"	Allows you to send special settings to the driver or X server.

Using XF86Setup

XF86Setup is a graphical configuration tool. It uses the 16-color VGA X server, which should work on almost all video cards. Figure 9-1 shows the main configuration screen. XF86Setup uses a tabbed type interface, with these screens:

- ♦ Mouse
- ♦ Keyboard
- ♦ Card

- ♦ Monitor
- ♦ Modeselection
- ♦ Other

To start the configuration, just type **XF86Setup**. Once you have gone through all of the tabs click Done, and XF86Setup will test the X server with your settings. The following sections describe each of the tabs.

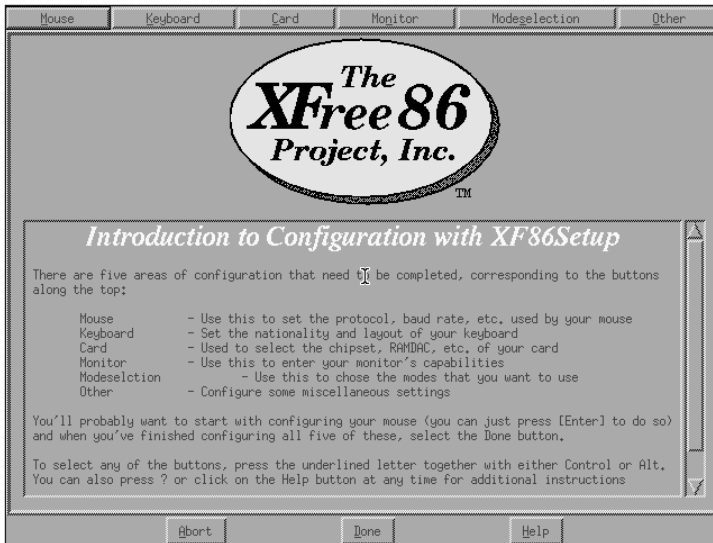


Figure 9-1: XF86Setup tool

Mouse

The Mouse tab allows you to set the options for your pointing device. This screen is shown in Figure 9-2. When you first go to this tab, you are presented with a screen that shows the keyboard shortcuts to move around. This way you can navigate around to set up your mouse before the mouse begins to work. Choose the correct mouse type, device, and number of buttons. If you have a two-button mouse be sure to enable three-button emulation.

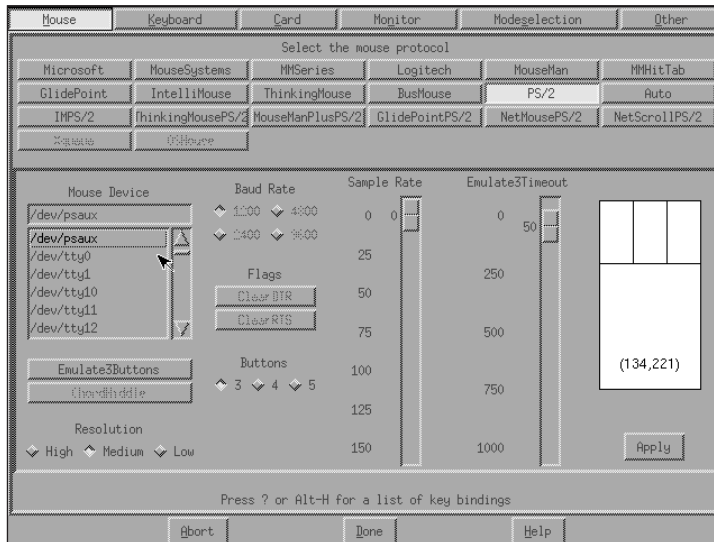


Figure 9-2: Mouse tab

Keyboard

The Keyboard tab allows you to set the options for your keyboard. This screen is shown in Figure 9-3. Choose the correct keyboard type, and the Shift and Control key functions.

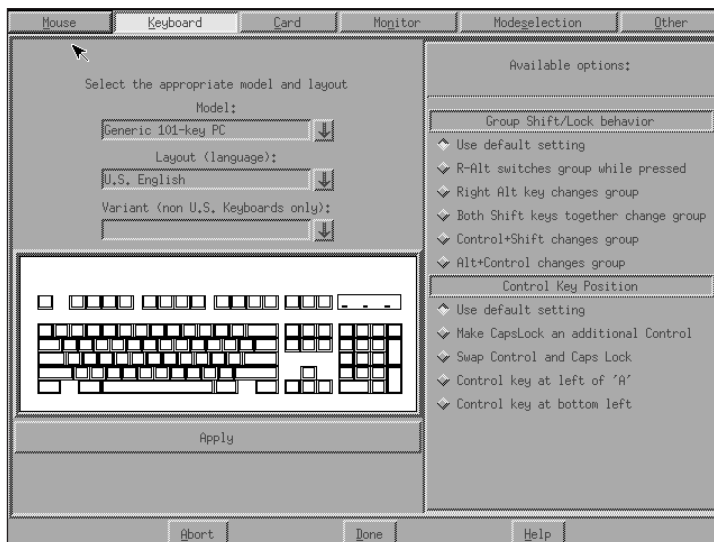


Figure 9-3: Keyboard tab

Card

The Card tab allows you to set the options for your video card. This screen is shown in Figure 9-4. Choose the correct video card for your system. If your card is not listed, try to find another listing for the video chipset it uses or pick another card that uses the same chipset. The Detailed Setup option lets you define manual entries to the X configuration file and specify the amount of video RAM on your card.

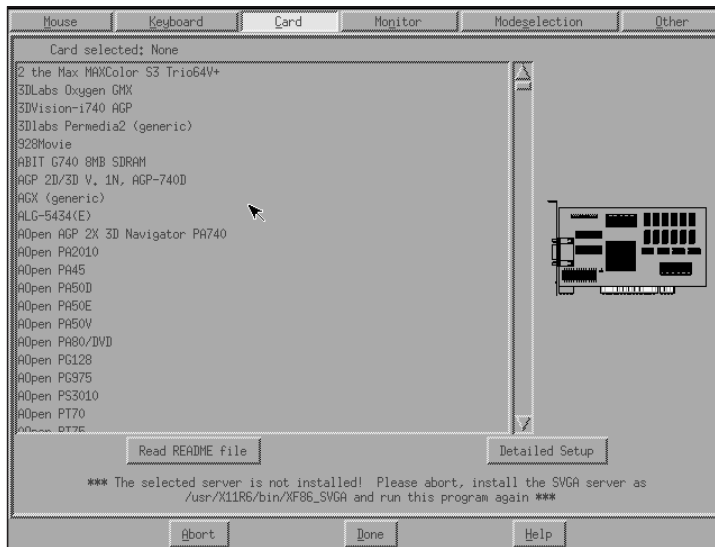


Figure 9-4: Card tab

Monitor

The Monitor tab allows you to set the resolution options for your monitor. This screen is shown in Figure 9-5. Choose the resolution and refresh rates that match your monitor. If none of the options match, you can enter the horizontal and vertical sync rates for your monitor. These settings should be available in your monitor manual.

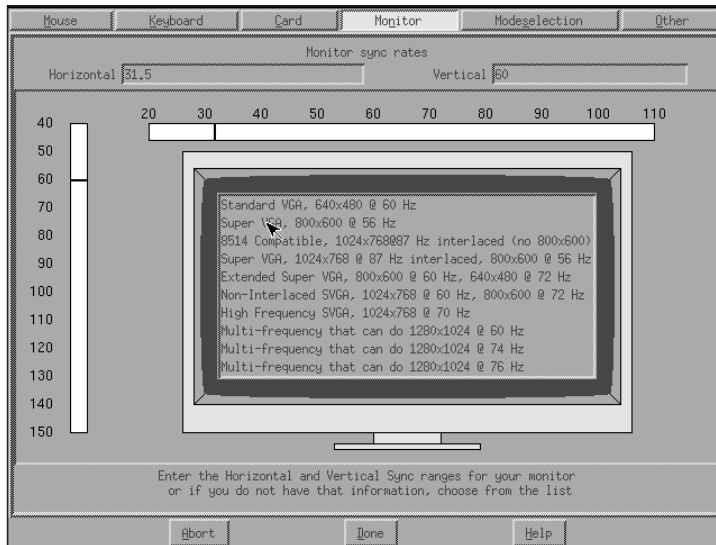


Figure 9-5: Monitor tab

Modeselection

The Modeselection tab allows you to set the resolutions and color depths to run while in X. This screen is shown in Figure 9-6. Choose the resolutions that you want to have available while in X, as well as the default color depth. Table 9-8 lists the color depths. Not all X servers support all color depths.

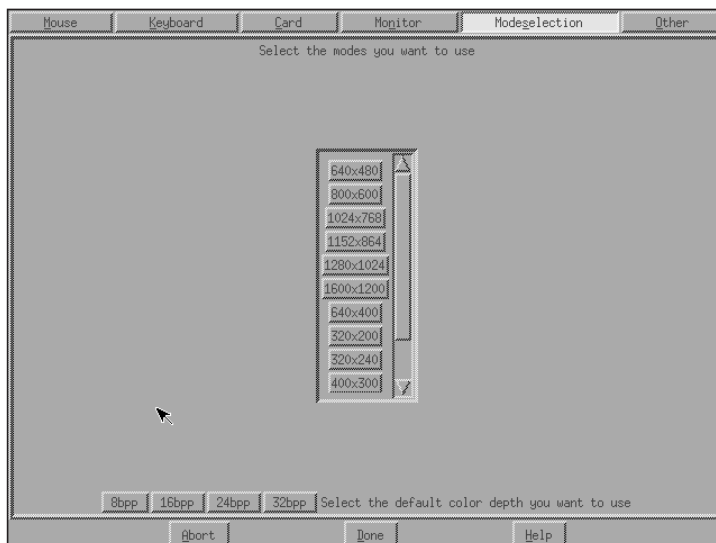


Figure 9-6: Modeselection tab

Table 9-8
Color Depths

<i>Bit Depth</i>	<i>Colors</i>
8bpp	256 colors
16bpp	65,535 colors
24bpp	16.7 million colors
32bpp	16.7 million colors (faster on some video cards)

Other

The last tab is the Other tab, which has miscellaneous settings. This screen is shown in Figure 9-7. This screen lets you set several small options, such as how to kill the server, whether clients can change the video mode, and who can change keyboard and mouse settings.

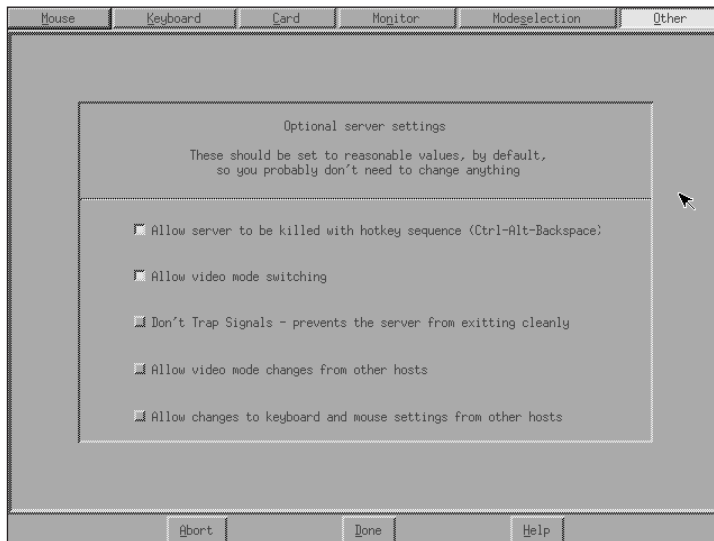


Figure 9-7: Other tab



XF86Setup provides an easy-to-use graphical interface for configuring your X server.

Using xf86config

Another XFree86 configuration tool is `xf86config`. It is a text mode tool that uses a series of menus to gather input from the user. It is not as pretty as `XF86Setup`, but it does not require the 16-color VGA server to be installed.



Tip

Do not get confused between `XF86Config` and `xf86config`. `XF86Config` is the configuration file, and `xf86config` is the tool that can be used to create it.

Figure 9-8 shows an example of `xf86config`. This tool is much more cumbersome to use than `XF86Setup` and requires more intimate knowledge of the hardware present in your system. We don't recommend that you use this tool unless you have problems with the others. The following steps walk you through the configuration process.

1. The first few screens configure your mouse. Choose the correct type of mouse you have or the closest option offered.
2. Choose whether to enable three-button emulation for two-button mice.
3. Enter the path to the mouse device. Most distributions link the file `/dev/mouse` to the actual interface device. So if you are unsure, try `/dev/mouse`.
4. The next couple of screens explain the keyboard maps and prompt you to choose the correct keymap. Choose the correct one for your system. Most users should choose option 1 or 2.
5. The next few screens configure the monitor. Choose the synchronization rates for your monitor. If there is no entry that matches your monitor, enter the rates listed in your monitor manual.
6. Next, you will be prompted for three text strings. These are the identifier, vendor, and model of your monitor. These are used to keep different monitor configurations separate in the X configuration file. There are no wrong answers, so type in the information requested in case you need to reference it later in the X configuration file.
7. Next, you will be prompted for your video card or chipset. A long list of available configurations is displayed. Scroll through the list and enter the number that closest matches the video card or video chipset in your system.
8. Choose the correct X server for your video card. The last option presented tells `xf86config` to install the X server specified in the video card configuration you chose. This is usually the best option to take.
9. Specify the amount of video memory installed in your system.
10. Again, you will be prompted for three strings making up the identifier, vendor, and model. Enter information that is appropriate for your configuration should you need to reference it later in the X configuration file.

11. Some video cards require a RAMDAC setting. Check with your video card manufacturer or look on the card itself for the RAMDAC chipmaker. This is usually a large chip near the video output connector.
12. Some video cards have a programmable clockchip that requires configuration. Newer video cards do not use this. Unless you know your video card requires this setting, hit Enter to go on without making a selection.
13. Older video cards required clock probes to get the correct clock information. This could either be done during setup and put in the configuration file or be done every time the X server started. Unless your video card is several years old it is best to not do a probe.
14. The next several screens let you set the supported resolutions and color depths. The default settings are displayed, but you have the option to change them if you want.
15. Virtual desktops allow you to have a resolution that is larger than your monitor can support. This way you can scroll around the larger desktop and have more workspace. When prompted, tell `xf86config` whether you want to use virtual desktops or not.
16. The final step is to have `xf86config` write the new `XF86Config` file.

```

Now you must determine which server to run. Refer to the manpages and other
documentation. The following servers are available (they may not all be
installed on your system):

 1 The XF86_Mono server. This a monochrome server that should work on any
   UGA-compatible card, in 640x480 (more on some SUGA chipsets).
 2 The XF86_UGA16 server. This is a 16-color UGA server that should work on
   any UGA-compatible card.
 3 The XF86_SUGA server. This is a 256 color SUGA server that supports
   a number of SUGA chipsets. On some chipsets it is accelerated or
   supports higher color depths.
 4 The accelerated servers. These include XF86_S3, XF86_Mach32, XF86_Mach8,
   XF86_8514, XF86_P9000, XF86_AGX, XF86_M32, XF86_Mach64, XF86_I128,
   XF86_S3U, and XF86_3DLabs.

These four server types correspond to the four different "Screen" sections in
XF86Config (vga2, vga16, svga, accel).

 5 Choose the server from the card definition, XF86_S3.

Which one of these screen types do you intend to run by default (1-5)? █

```

Figure 9-8: `xf86config`



`xf86config` is a text mode configuration tool that is harder to use than `XF86Setup`.

Detecting video hardware

The `SuperProbe` tool can be used to probe the video hardware in a system and report back information. This information can be useful for determining which video card to choose with the other `XF86Config` creation tools. Be careful though; this tool can cause the system to lock up, requiring you to reboot. The following is an example of its output:



The SuperProbe utility could cause your system to lock up and stop responding.

```
SuperProbe Version 2.21 (12 October 1999)
(c) Copyright 1993,1994 by David Wexelblat <dwex@xfree86.org>
(c) Copyright 1994-1998 by The XFree86 Project, Inc
```

```
This work is derived from the 'vgadoc2.zip' and
'vgadoc3.zip' documentation packages produced by Finn
Thoegersen, and released with all appropriate permissions
having been obtained. Additional information obtained from
'Programmer's Guide to the EGA and VGA, 2nd ed', by Richard
Ferraro, and from manufacturer's data books
```

```
Bug reports are welcome, and should be sent to XFree86@XFree86.org.
In particular, reports of chipsets that this program fails to
correctly detect are appreciated.
```

```
Before submitting a report, please make sure that you have the
latest version of SuperProbe (see http://www.xfree86.org/FAQ).
```

```
WARNING - THIS SOFTWARE COULD HANG YOUR MACHINE.
          READ THE SuperProbe.1 MANUAL PAGE BEFORE
          RUNNING THIS PROGRAM.
```

```
          INTERRUPT WITHIN FIVE SECONDS TO ABORT!
```

```
First video: Super-VGA
Chipset: Cirrus CL-GD5446 (PCI Probed)
Memory: 2048 Kbytes
RAMDAC: Cirrus Logic Built-in 8-bit pseudo-color DAC
          (with 6-bit wide lookup tables (or in 6-bit mode))
```



SuperProbe works with most ISA, VLB, and PCI video cards. Very new cards will probably not be detected.

Fine tuning video

The `xvidtune` tool can be used to fine-tune the placement of the image on the monitor. It can be used to center the display on the monitor or make other adjustments. It outputs mode lines that can be used in the `XF86Config` file. Figure 9-9 shows an example of the `xvidtune` interface.

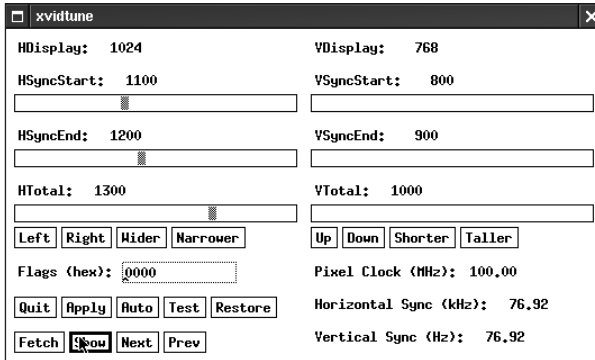


Figure 9-9: xvidtune

Configuring fonts

The X font server keeps track of all of the fonts installed on a system and lets a client use them, either locally or from across the network. Usually the font server does its work in the background, but when new fonts are added you need to make sure the font server is notified.

The X server knows to get fonts from the font server with the following entry in the XF86Config file:

```
FontPath    "unix/:-1"
```

The syntax for this is as follows:

```
FontPath    "<transport>/<hostname>:<port number>"
```

This example tells the X server to use the font server on the local system, since the `unix` transport, no hostname, and the `-1` port is used. If you use a network font server, you would need to change this line. For example, suppose you have a font server named `font.ournetwork.com` listening on port 7100. You would use this directive:

```
FontPath    "tcp/font.ournetwork.com:7100"
```

Font files are stored in the `/usr/X11R6/lib/X11/fonts` directory tree. Each subdirectory off this tree contains fonts and a couple of files that tell the X server about the fonts installed. If you install new fonts, you need to run `mkfontdir` to create these files. The syntax for `mkfontdir` is as follows:

```
mkfontdir <font subdirectory>
```

For example, to create the necessary files for `/usr/X11R6/lib/X11/fonts/newfonts`, you would run the following:

```
mkfontdir /usr/X11R6/lib/X11/fonts/newfonts
```

When this command runs, it creates the `fonts.dir` file. The following is an example of this file:

```
UTI__14.pcf.gz -adobe-utopia-regular-i-normal--15-140-75-75-p-79-iso8859-1
UTI__24.pcf.gz -adobe-utopia-regular-i-normal--25-240-75-75-p-133-iso8859-1
cour008.pcf.gz -adobe-courier-medium-o-normal--8-80-75-75-m-50-iso8859-1
```

Another file that is used in the font directories is `fonts.alias`. It is a hand-created file that consists of two columns separated by a white space such as a space or tab. The first column contains the alias and the second column contains a font name or pattern to match.

Once the new font files are created, you need to instruct the running X server to reread the font directories. This is done with the `xset` command. To have it reread the font directories you would use the following:

```
xset fp rehash
```



The `mkfontdir` command must be run whenever new fonts are added to your system.

Starting X

Objective

2.10 X

- **Setup XDM.** Turn `xdm` on and off, change the `xdm` greeting, change default bit-planes for `xdm`, set-up `xdm` for use by X-stations

You can run X on your system in two ways. The first method is to invoke it manually from the command line after login. The second way is to have the system boot into a graphical mode with a GUI login.

Starting X manually

The X Window System can be manually started with the `startx` command. The next several sections follow the flow of scripts that begins when the user executes `startx`. The problem is that distributions vary on the exact path they take through the startup scripts and on what each script calls. It is recommended that you walk through the scripts for your distribution and version to know exactly what happens. Plus, it's excellent experience!

startx

This `startx` command is actually a script that sets several variables and finally calls the `xinit` application. The following is an example of a `startx` script with comments.

```
#!/bin/sh
#
# (c) 1999 Red Hat Software, Inc.
bindir=/usr/X11R6/bin
userclientrc=$HOME/.xinitrc
userserverrc=$HOME/.xserverrc
sysclientrc=/etc/X11/xinit/xinitrc
sysserverrc=/etc/X11/xinit/xserverrc
clientargs=""
serverargs=""
```

This sets several variables that point to other script files that get called later.

```
if [ -f $userclientrc ]; then
    clientargs=$userclientrc
else if [ -f $sysclientrc ]; then
    clientargs=$sysclientrc
fi
fi

if [ -f $userserverrc ]; then
    server=$userserverrc
else if [ -f $sysserverrc ]; then
    server=$sysserverrc
fi
fi
```

These lines do a test `-f` check to see if the files in the previous variables exist. The next section, omitted here, executes the script files stored in the variables. These are the `xinitrc` and `xserverrc` files.

```
# set up default Xauth info for this machine
mcookie=`mcookie`
serverargs="$serverargs -auth $HOME/.Xauthority"
xauth add $display . $mcookie
xauth add `hostname -f`$display . $mcookie
```

This block sets up the X security information by executing `xauth`. The `xauth` tool is covered in more detail later in this chapter.

```
xinit $clientargs -- $server $display $serverargs
```

Finally, the `startx` script executes `xinit`.



The `startx` command is a script that executes the `xinit` process.

The `xinitrc` and `.xinitrc` files

A script file is executed whenever the `xinit` process starts up. Each user can create his or her own personal `.xinitrc` in their home directory. If this file does not exist, the system-wide `xinitrc` stored in `/etc/X11/xinit` or `/usr/X11/xinit/xinitrc` will be used. If you put multiple tasks in your `.xinitrc` file, the last one should not be sent to the background or the X server may exit.



A user's `.xinitrc` script is run when a user starts X with the `startx` command. If no `.xinitrc` is found, the `xinit` process runs the system-wide `xinitrc` script.

The `Xclients` and `.Xclients` files

Some distributions have the `xinitrc` file call the `Xclients` script file. Red Hat uses this file to test and see which window managers are installed and to run the one that you have configured as your preference. A user may have his or her own `Xclients` file in `~/.Xclients`.

Using XDM

An alternative to booting to text mode and manually running `startx` is to run a display manager. There are several display managers available, but XFree comes with XDM. It is shown in Figure 9-10. XDM lets you boot up to a GUI login screen and go directly into X. When you exit out of X you are brought back to the GUI login screen.

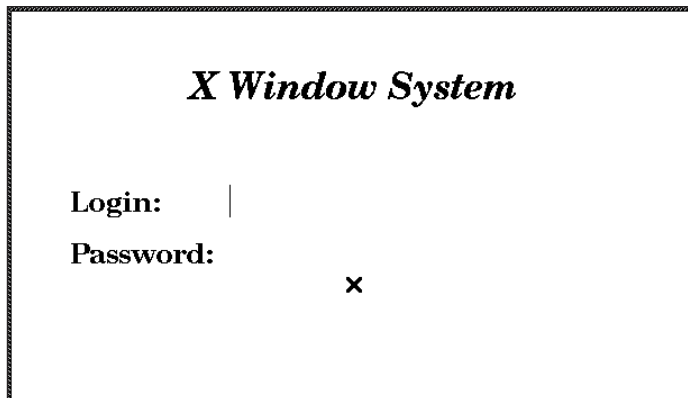


Figure 9-10: XDM

**Tip**

Both KDE and GNOME provide their own XDM replacements, which look better and provide more functionality. The new display managers are used by default when the environments are installed.

Enabling and disabling XDM

When enabled, the `xdm` application is started at boot by a runlevel script. Debian just adds the `S99XDM` script to runlevel 2. With Red Hat, the default runlevel is changed from 3 to 5 to enable `xdm`.

To disable XDM you need either to remove the startup script in Debian or to change the runlevel from 5 to 3 in Red Hat. This can quickly be done with a `telinit 3` command.

**Exam Tip**

The `xdm` process is started from a runlevel script in distributions that use Sys V `init`.

Configuring XDM

XDM can be difficult and complicated to configure, since it has several files that must be maintained. The following sections cover the most important files to look at when dealing with XDM.

`xdm-config`

The main configuration file for XDM is `xdm-config`. This file sets the basic operation of `xdm`. For most installations it should never be changed, but if you configure your system to allow remote logins, you may need to make some adjustments. That is covered in more detail in the next section. The following is a sample of an `xdm-config` file.

```
! $XFree86: xc/programs/xdm/config/xdm-conf.cpp,v 1.1.1.2.4.2 1999/10/12 18:33:2
9 hohndel Exp $
!
! $XConsortium: xdm-conf.cpp /main/3 1996/01/15 15:17:26 gildea $
DisplayManager.errorLogFile:    /var/log/xdm-error.log
DisplayManager.pidFile:         /var/run/xdm.pid
DisplayManager.keyFile:         /etc/X11/xdm/xdm-keys
DisplayManager.servers:         /etc/X11/xdm/Xservers
DisplayManager.accessFile:      /etc/X11/xdm/Xaccess
! All displays should use authorization, but we cannot be sure
! X terminals will be configured that way, so by default
! use authorization only for local displays :0, :1, etc.
DisplayManager._0.authorize:    true
DisplayManager._1.authorize:    true
! The following three resources set up display :0 as the console.
DisplayManager._0.setup:        /etc/X11/xdm/Xsetup_0
DisplayManager._0.startup:      /etc/X11/xdm/GiveConsole
DisplayManager._0.reset:        /etc/X11/xdm/TakeConsole
!
```

```

DisplayManager*resources:      /etc/X11/xdm/Xresources
DisplayManager*session:      /etc/X11/xdm/Xsession
DisplayManager*authComplain:  false ! SECURITY: do not listen for XDMCP or
Chooser requests
! Comment out this line if you want to manage X terminals with xdm
DisplayManager.requestPort:   0

```

Xsession

The XDM application uses the `Xsession` script file to know which desktop to give a user after they log in. It has many of the same commands as the `xinitrc` file and performs a similar function for those starting X with XDM. The following is an example of `Xsession` taken from a Red Hat system, with comments.



The `Xsession` file decides which window manager to use when logging in with XDM.

```

#!/bin/bash -login
# (c) 1999, 2000 Red Hat, Inc.

# redirect errors to a file in user's home directory if we can
for errfile in "$HOME/.xsession-errors" "${TMPDIR-/tmp}/xses-$USER" "/tmp/xses-$USER"
do
    if ( cp /dev/null "$errfile" 2> /dev/null )
    then
        chmod 600 "$errfile"
        exec > "$errfile" 2>&1
        break
    fi
done

```

This section creates a file named `.xsession-errors` in the user's home directory, which is used to log any errors.

```
xsetroot -solid '#356390'
```

This sets the desktop background to a solid color.

```

# clean up after xbanner
if [ -x /usr/X11R6/bin/freetemp ]; then
    /usr/X11R6/bin/freetemp
fi
userresources=$HOME/.Xresources
usermodmap=$HOME/.Xmodmap
sysresources=/etc/X11/xinit/Xresources
sysmodmap=/etc/X11/xinit/Xmodmap

# backward compatibility
oldsysresources=/etc/X11/xinit/.Xresources
oldsysmodmap=/etc/X11/xinit/.Xmodmap

```

These lines are similar to those in the `xinitrc` file and specify the files for several variables. Notice the path that this script uses for the `Xresources` file, in case you need to customize it.

```
# merge in defaults
if [ -f "$oldsysresources" ]; then
    xrdp -merge "$oldsysresources"
fi

if [ -f "$sysresources" ]; then
    xrdp -merge "$sysresources"
fi

if [ -f "$userresources" ]; then
    xrdp -merge "$userresources"
fi
# merge in keymaps
if [ -f "$sysxkbmap" ]; then
    setxkbmap `cat "$sysxkbmap" `
    XKB_IN_USE=yes
fi

if [ -f "$userxkbmap" ]; then
    setxkbmap `cat "$userxkbmap" `
    XKB_IN_USE=yes
fi

if [ -z $XKB_IN_USE -a ! -L /etc/X11/X ]; then
    if grep '^exec.*Xsun' /etc/X11/X > /dev/null 2>&1 && [ -f /etc/X11/XF86Conf
ig ]; then
        xkbsymbols=`sed -n -e 's/[      ]*XkbSymbols[      ]*"(.*)".*$/\1/p' /etc
/X11/XF86Config`
        if [ -n "$xkbsymbols" ]; then
            setxkbmap -symbols "$xkbsymbols"
            XKB_IN_USE=yes
        fi
    fi
fi # xkb and xmodmap don't play nice together
if [ -z $XKB_IN_USE ]; then
    if [ -f "$oldsysmodmap" ]; then
        xmodmap "$oldsysmodmap"
    fi

    if [ -f "$sysmodmap" ]; then
        xmodmap "$sysmodmap"
    fi

    if [ -f "$usermodmap" ]; then
        xmodmap "$usermodmap"
    fi
fi

unset XKB_IN_USE
```

This large section combines several variables and loads the correct keymaps.

```
# run all system xinitrc shell scripts.
for i in /etc/X11/xinit/xinitrc.d/* ; do
    if [ -x "$i" ]; then
        . "$i"    fi
done

# now, we see if xdm/gdm/kdm has asked for a specific environment
case $# in
1)
    case $1 in
    failsafe)
        exec xterm -geometry 80x24-0-0
        ;;
    gnome)
        exec gnome-session
        ;;
    kde)
        exec startkde
        ;;
    anotherlevel)
        # we assume that switchdesk is installed.
        exec /usr/share/apps/switchdesk/Xclients.anotherlevel
        ;;
    esac
esac
# otherwise, take default action
if [ -x "$HOME/.xsession" ]; then
    exec "$HOME/.xsession"
elif [ -x "$HOME/.Xclients" ]; then
    exec "$HOME/.Xclients"
elif [ -x /etc/X11/xinit/Xclients ]; then
    exec /etc/X11/xinit/Xclients
else
    # should never get here; failsafe fallback
    exec xsm
fi
```

Finally, this section starts the appropriate desktop interface. If the system does not use one of the specified desktops, the `xsm` command is issued, which starts a basic X session.

Xresources

The XDM login screen can be customized with the `Xresources` file that is referenced in the `Xsession` file. This file uses the standard `Xresource` format, which is discussed in more detail in the “Customizing X applications” section later in the chapter.

```
! $XConsortium: Xresources /main/8 1996/11/11 09:24:46 swick $
xlogin*login.translations: #override\
    Ctrl<Key>R: abort-display()\n\
```

```

        <Key>F1: set-session-argument(failsafe) finish-field()\n\
        Ctrl<Key>Return: set-session-argument(failsafe) finish-field()\n\
        <Key>Return: set-session-argument() finish-field()
xlogin*borderWidth: 3
xlogin*greeting: CLIENTHOST
xlogin*namePrompt: login:\040
xlogin*fail: Login incorrect
#ifdef COLOR
xlogin*greetColor: CadetBlue
xlogin*failColor: red
*Foreground: black
*Background: #ffffff0
#else
xlogin*Foreground: black
xlogin*Background: white
#endif

XConsole.text.geometry: 480x130
XConsole.verbose:      true
XConsole*iconic:      true
XConsole*font:        fixed

Chooser*geometry:      700x500+300+200
Chooser*allowShellResize: false
Chooser*viewport.forceBars: true
Chooser*label.font:    *-new century schoolbook-bold-i-normal-*-240-*
Chooser*label.label:   XDMCP Host Menu from CLIENTHOST
Chooser*list.font:     -*-*medium-r-normal-*-*230-*-*c-*iso8859-1
Chooser*Command.font:  *-new century schoolbook-bold-r-normal-*-180-*

```

Almost all of the entries are self-explanatory and allow you to change the color, font, text size, or other aesthetic features of the XDM login display.



XDM uses an `Xresources` file to configure the display properties of the login screen.

Using X

Objective

2.10 X

- **Identify and terminate runaway X applications.** Identify and kill X applications that won't die after user ends an X-session. Example: netscape, tkat, etc.
- **Install & Customize a Window Manager Environment.** Select and customize a system-wide default window manager and/or desktop environment, demonstrate an understanding of customization procedures for window manager menus, configure menus for the window manager, select and configure the desired x-terminal (xterm, rxvt, aterm etc.), verify and resolve library dependency issues for X applications, export an X-display to a client workstation. Commands: Files: `.xinitrc`, `.Xdefaults`, various `.rc` files.

After fighting the battle of installing and configuring X, you now get the payoff of using it! The X environment is extremely customizable, shockingly so to some new users of Linux and X.

Choosing a window manager or environment

As mentioned earlier in this chapter, a variety of window managers are available. And that list is only the most popular; many other less-well-known window managers are available as well. Many people just run a window manager while others run an entire desktop environment such as KDE or GNOME, which combines a window manager along with a suite of applications and tools that integrate well.

The chosen window manager or desktop environment is usually started from the `Xclient`, `Xsession`, or `.xinitrc` file, depending on the distribution and user's choice. Red Hat uses the `Xclient` file while Debian uses a system-wide `Xsession` file for local and remote users. If you want to run a window manager that these files do not know about, you can manually put it in your `.xinitrc` file in your home directory. This script gets executed whenever the `xinit` process starts.

Once you have chosen your X environment, you can then work to customize the system. One method is to use themes, which change the overall look and feel of the system. A popular site for window manager themes is <http://www.themes.org>. While window managers used to require the user to manipulate configuration files by hand to change even simple things like menus, current window managers often come with control panel-like tools to do this easily. Figure 9-11 shows an example of the GNOME menu editor.

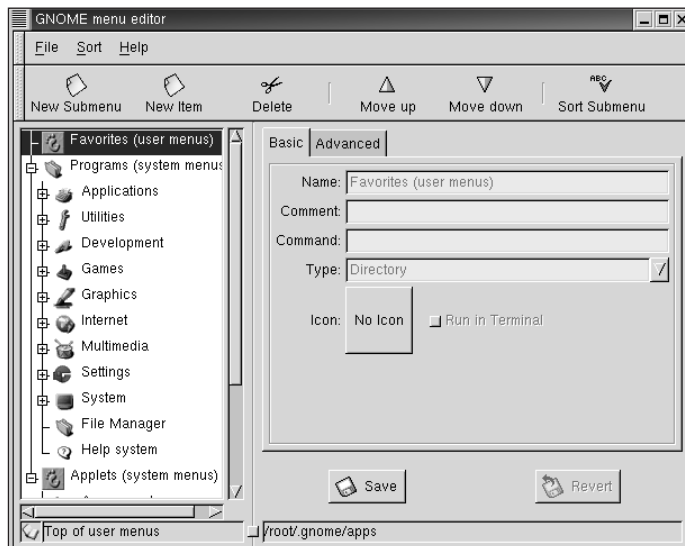


Figure 9-11: GNOME menu editor

Using X clients

Several command-line parameters apply to almost all X client applications. These let you set the startup configuration using a standard syntax. They are known as the *X Toolkit options*, and while they may be cumbersome to type each time, they are useful to use in a script or menu. Table 9-9 shows these parameters.

Table 9-9
X Toolkit Client Options

<i>Option</i>	<i>Function</i>
<code>-bg <color> or -background <color></code>	Sets the default background color of the application.
<code>-fg <color> or -foreground <color></code>	Sets the default text color for the application.
<code>-bd <color> or -bordercolor <color></code>	Sets the default color of the window border.
<code>-bw <number> or -borderwidth <number></code>	Specifies the width in pixels of the window border.
<code>-display hostname:display.screen</code>	Specifies the host, display number, and screen number to display the application to.
<code>-fn or -font </code>	Specifies the font to use for text.
<code>-geometry <width>x<height>+<x>+<y></code>	Specifies the initial size and location of the window.
<code>-iconic</code>	Tells the application to start in an icon state, if possible.
<code>-name <name></code>	Specifies the name under which resources for the applications should be found.
<code>-rv or -reverse</code>	Tells the application to simulate reverse video, if possible.
<code>+rv</code>	Tells the application to not attempt to simulate reverse video.

Using a terminal emulator

Many users run X as a graphical interface to multiple terminal emulators on a single screen simultaneously. A number of terminal emulators exist to satisfy everyone's needs. The following list several popular terminal emulators.

xterm

The “standard” terminal emulator, `xterm` has been around for a long time. It provides DEC VT102/VT220 and Tektronix 4104 emulation for applications. `xterm` supports a great number of command-line options to configure the emulator at runtime, but it also supports the standard X Toolkit options listed in the previous section.

rxvt

For those users who do not need all of the functionality of `xterm`, there is `rxvt`. `rxvt` does not emulate as many systems as `xterm` and does not support the X Toolkit command-line options. The benefit is a smaller memory footprint, as it was designed to be used on systems with a lot of terminal emulators open simultaneously.

aterm

`aterm` is similar to `rxvt` in that it does not support as many terminal emulations as `xterm`. It also does not support the X Toolkit options. It was designed to work with the AfterStep window manager, but it is not required. `aterm` offers several other options over `xterm`, including a transparency mode where the desktop background is seen through the window.

Eterm

If you want the best-looking, most-configurable terminal emulator, `Eterm` is the one. It was designed to work with the Enlightenment window manager, and it shows. It requires more memory than other terminal emulators require, but can be changed and customized to look however you want. `Eterm` also supports themes.

gnome-terminal

The GNOME desktop environment comes with its own GNOME-compliant terminal emulator, `gnome-terminal`. It is written using the GTK (GIMP Tool Kit) libraries so it will follow the look and feel of the GNOME theme you install. It is a full-featured terminal client that is a good option if you use GNOME.

konsole

Not to be outdone, the KDE group includes a terminal emulator named `konsole` with their desktop environment. It is a full-featured emulator that follows the look and feel of your KDE themes.

Customizing X applications

Again, the X environment is very customizable. Applications that are written to use desktop environments such as GNOME and KDE follow the look and feel you set up in that environment, but what about the other applications? Many applications were not written for a desktop environment and use only the standard X Toolkit libraries, but still give you methods to customize their look.

Applications that use the X Toolkit library can be customized using the Xresource format. Many sample Xresource files are stored in the `/usr/X11R6/lib/X11/app-defaults` or `/etc/X11/app-defaults`. They are named after the application they reference. The following is an example from the Xman file, which is for the xman application that displays man pages.

```
*manualBrowser.Title:      Manual Page
*manualBrowser.IconName:   Manual Page
*manualBrowser.geometry:   600x600
```

This shows the default title, icon name, and window geometry. The actual syntax for the entries is very complex, but the simple ones that you are most likely to change follow the format above, which is the application name followed by the property to change.

By changing a setting in a file from `apps-default`, you will change the default behavior for all users. To change the behavior of the application for only one user, you should place the Xresource entries in a file named `.Xdefaults` in their home directory.



The `.Xdefaults` file stores a user's customizations to X applications.

Using special keys

There are several special key combinations to use within X. Users used to switching virtual consoles in Linux with the Alt-Function key combo may be surprised when it does not work in X. Table 9-10 lists these special combinations.

Table 9-10
Key Combinations

<i>Combination</i>	<i>Function</i>
Ctrl-Alt-<Keypad +>	Changes to a higher video resolution, if configured.
Ctrl-Alt-<Keypad ->	Changes to a lower video resolution, if configured.
Ctrl-Alt-Backspace	Exits out of X quickly, unless disabled in <code>XF86Config</code> .
Ctrl-Alt-<F1 through F6>	Switches to the text mode console.
Ctrl-Alt-<F7>	Switches back to graphical mode from a text console.

Managing bad applications

Because of the graphical nature of X applications, they are usually more complex and harder to develop than text mode applications. Because of this many X

applications have gotten a bad reputation for not being stable or causing problems. The biggest one that most people have run into is the Netscape Navigator Web browser, but it is not the only one to blame.

Some applications just crash, while others do not actually close when told to exit. Instead, they stay open and continue to use resources that could be better used elsewhere. If you notice your system running slower than usual or hear the hard drive as memory is swapped to the swap partition, you need to check your process list. Make sure that no old applications that you thought were closed are still open and running in the background.



Many unstable applications run in X. The exam objectives specifically mention Netscape Navigator and `tkrat` (a graphical mail agent), so you should remember those.

Running X and Clients Remotely

Objective

2.10 X

- **Setup XDM.** Turn `xdm` on and off, change the `xdm` greeting, change default bit-planes for `xdm`, set-up `xdm` for use by X-stations
- **Install & Customize a Window Manager Environment.** Select and customize a system-wide default window manager and/or desktop environment, demonstrate an understanding of customization procedures for window manager menus, configure menus for the window manager, select and configure the desired x-terminal (`xterm`, `rxvt`, `aterm` etc.), verify and resolve library dependency issues for X applications, export an X-display to a client workstation. Commands: Files: `.xinitrc`, `.Xdefaults`, various `.rc` files.

A very nice feature of X is the ability to transparently display applications across a network. This can be used to set up inexpensive desktop computers on users' desks, while running the applications from a large network server. X provides the ability to either run individual client applications remotely or start the entire desktop from a remote system.

Configuring X security

X supports several different authentication mechanisms, some very complex. For most users, the basic host method works well and is easy to configure and manage. The host-based security is configured with the `xhost` command. The current authentication settings can be checked by just running `xhost` with no options.

```
[root@redhat /root]# xhost
access control enabled, only authorized clients can connect
INET:brain
INET:marvin
```

The `xhost` command can also be used to see if you are allowed to display clients on a remote system. This is done by changing the `DISPLAY` variable to that of the remote host and then running `xhost`. For example, to see if you can display applications on `redhat.the-nashes.net` you would type:

```
export DISPLAY=redhat.the-nashes.net:0.0
xhost
```

The `DISPLAY` variable is covered in detail in the next section.

The `xhost` command uses a few simple command-line options. To enable host-based authentication and clear any allowed hosts you would use `xhost -`. For example:

```
[root@redhat /root]# xhost -
access control enabled, only authorized clients can connect
```

To disable all authentication and allow anyone to connect to your system and display applications, you type `xhost +`. For example:

```
[root@redhat /root]# xhost +
access control disabled, clients can connect from any host
```



Running `xhost +` is a bad idea, since any user can connect to your X server.

Authorizing hosts to connect is done with the `+` option too. For example, to allow three hosts to connect you would type the following:

```
[root@redhat /root]# xhost +deedee marvin brain
deedee being added to access control list
marvin being added to access control list
brain being added to access control list
```

To remove a host from the allowed list you would use the `-` option. For example, to remove `deedee` you would type the following:

```
[root@redhat /root]# xhost -deedee
deedee being removed from access control list
```

Finally, to make sure your authentication is correct, just type `xhost`:

```
[root@redhat /root]# xhost
access control disabled, clients can connect from any host
INET:brain
INET:marvin
```



The host authentication method is configured using the `xhost` command.

Configuring remote clients

You can tell a client application to display on a remote system in two basic ways. The first involves using the `DISPLAY` environment variable. The format for the `DISPLAY` variable is as follows:

```
DISPLAY=hostname:display.screen
```

For example:

```
DISPLAY=marvin.the-nashes.net:0.0
```

The `hostname` field can either be the DNS name or an IP address. If only one user is using an X server on the remote system, the display will be 0. The screen number is used only in multiscreen environments and can be omitted if only one screen is being used. After changing the `DISPLAY` variable, any X client started will automatically be displayed on the system specified in the variable, if authentication allows it.

The second method for displaying clients remotely is to use the `-display` command-line parameter that most clients use. The syntax is as follows:

```
xclient -display hostname:display.screen [xclient arguments]
```

Notice the format is the same as the `DISPLAY` variable. This is useful if you want to display only one or two clients on another system.

Configuring remote login

Another very useful function of XDM is to let you log in remotely to another system and use the local system for display only. You can also configure a system to provide a client with a menu of possible login systems. The underlying functionality of this is provided and controlled by XDMCP (X Display Manager Control Protocol). This setup should work with any X clients and servers that support XDMCP.



The Linux Terminal Server Project has created a login system to be used with diskless workstations. You can find more information at <http://www.ltsp.org>.

Logging into remote systems

If a remote system is running XDM, it is easy to log into that system from within X and execute applications from it. The following syntax is used:

```
X -query <server name>
```

X also supports the ability to broadcast on the local network for any server running XDM. This is done with the following command:

```
X -broadcast
```

Linux systems can also be set up to provide a chooser, which is a list of hosts that you can log into. To request a chooser, you would use the following command:

```
X -indirect <server name>
```



Be sure to know the difference between these three commands for the exam.

Configuring XDM to provide a chooser

To configure XDM to provide a chooser you need to change the `/etc/X11/xdm/Xaccess` file. You need to add one or more chooser directives to the file. They have the following formats:

```
client CHOOSER server_list
```

The `client` entry defines which workstations will be provided with a chooser list, and the `server_list` specifies which servers will appear in the chooser. Wildcards can be specified in the client host lists. The following is an example:

```
*.the-nashes.net    CHOOSER    marvin brain
Norbert             CHOOSER    brain
!bugs
*                   CHOOSER    dag
```

If a host matches more than one line, the first line encountered is used. These lines define the following:

- ♦ The first line specifies that any host in the `the-nashes.net` domain is shown the `marvin` and `brain` servers in the chooser.
- ♦ The second line specifies that the host `norbert` will be shown the server `brain` in the chooser.
- ♦ An `!` before an entry denies the host from receiving a chooser.
- ♦ The last line specifies that any host will be shown the server `dag` in the chooser.

If you would prefer to dynamically generate the chooser list you can use the following directive:

```
*                   CHOOSER    BROADCAST
```

This causes the X server to generate the list displayed on the chooser for any host that asks. You can combine this with other directives discussed in this section.

The default installation of XDM is to ignore chooser requests. To enable chooser requests you must remove or comment out the following line from the `xdm-config` file.

```
! SECURITY: do not listen for XDMCP or Chooser requests
! Comment out this line if you want to manage X terminals with xdm
DisplayManager.requestPort: 0
```

Key Point Summary

The X Window System provides a very customizable graphical interface to Linux. While at times the system is complicated to install and configure, new configuration tools have helped this immensely. Keep in mind that the LPI exam covers the basics of X that are the same across distributions. Many distributions have their own ways to start up X and make detailed changes.

- ♦ The X server runs on a system and displays the desktop and applications.
- ♦ X clients are applications that connect to an X server and get displayed.
- ♦ Window managers handle window borders, icons, virtual desktops, toolbars, and so on.
- ♦ Desktop environments combine a window manager with other system tools, programming toolkits, applications, and configuration editors to create an entire environment.
- ♦ X can be installed from distribution-created package files or binary files distributed by the XFree organization.
- ♦ X v3 uses separate X servers for different video chipsets.
- ♦ The X server executable is linked to the `/etc/X11/X` file.
- ♦ The main configuration file for an X v3 server is `XF86Config`.
- ♦ The `XF86Config` file is broken up into several sections, including `Files`, `ServerFlags`, `Keyboard`, `Pointer`, `Monitor`, `Device`, and `Screen`.
- ♦ Font paths are listed in the `Files` section and point to directories containing font files, or a font server.
- ♦ `XF86Setup` is a graphical configuration tool for the `XF86Config` file.
- ♦ `xf86config` is a text-based configuration tool for `XF86Config`, but can be very confusing to use.
- ♦ `SuperProbe` can be used to detect some video hardware.
- ♦ `xvidtune` is a tool that lets you customize your video display and creates mode lines that are useable in the `XF86Config` file.
- ♦ The `mkfontdir` command should be run any time new fonts are installed.

- ♦ The `startx` command is script used to start X from a command line.
- ♦ After `startx` executes `xinit`, it looks for the user's `.xinitrc` file, and if one is not found, it executes the system's `xinitrc`.
- ♦ The XDM application is used to give users a graphical login screen at boot.
- ♦ XDM is configured through the `xdm-config` file.
- ♦ The XDM display can be customized by using its `Xresources` file.
- ♦ X applications that are written to use the X Toolkit library take standard command-line parameters to configure their options at startup.
- ♦ Many different terminal emulators are available for X, each with their own advantages and disadvantages.
- ♦ X application displays can be customized by putting their `Xresource` entries into a user's `.Xdefaults` file.
- ♦ Some complex X applications behave badly, do not exit when instructed, or crash unexpectedly.
- ♦ X can use host-based security, which is controlled with the `xhost` tool.
- ♦ An X client's display can be sent to another host by using the `DISPLAY` environment variable or by the `-display` command-line option.
- ♦ XDM can be used to log into a system remotely and run clients from the remote server, while being displayed on the local system.
- ♦ XDM can be configured to present a menu to a client prompting them for the system to log into.



STUDY GUIDE

The following questions and exercises will allow you to review the information covered in this chapter. Take your time when completing this section of the chapter, carefully reviewing any questions that you may have. Simply answering the question correctly is not as important as understanding the answer, so review any material that you might still be unsure of. Being comfortable with the questions and answers presented here will help you be more prepared for the certification exam questions.

Assessment Questions

1. Which tool may cause the system to lock up and stop responding?
 - A. SuperProbe
 - B. xvidtune
 - C. XF86Setup
 - D. startx
2. Which piece of software is different for different video chipsets?
 - A. Window manager
 - B. Desktop environment
 - C. xinitrc
 - D. X server
3. What needs to be done to change the X server from SVGA to S3?
 - A. `switchX S3`
 - B. `ln -sf /etc/X11/X /usr/X11R6/bin/XFree_S3`
 - C. `ln -sf /usr/X11R6/bin/XFree_S3 /etc/X11/X`
 - D. Edit the XF86Config file and change the Server directive.
4. Which section in the XF86Config file configures the mouse?
 - A. Mouse
 - B. Input
 - C. User
 - D. Pointer

5. Which of the following XF86Config configuration tools is easiest to use?
 - A. vi
 - B. XF86Setup
 - C. xf86config
 - D. SuperProbe

6. Which directive in the XF86Config file disables the Ctrl-Alt-Backspace key combination?
 - A. NoZap
 - B. DontZap
 - C. NoExit
 - D. DontExit

7. Which of the following XF86Config directives is required?
 - A. VendorName
 - B. ModelName
 - C. Identifier
 - D. Gamma

8. Which tool is used to detect most video hardware?
 - A. SuperProbe
 - B. xvidtune
 - C. XF86Setup
 - D. xf86config

9. You would enter the _____ command if you added fonts to the /usr/X11R6/lib/X11/fonts/myfonts directory.

10. If a user wants an application to load when they run startx, they should put it in their _____ file.

11. Which file must be changed to allow remote logins with XDM?
 - A. Xresources
 - B. Xsession
 - C. xdm-config
 - D. xdmrc

12. Which file would you change to alter the text shown at the XDM login screen?
- A. Xsession
 - B. Xresources
 - C. xdm-config
 - D. xdmrc
13. Which parameter would you use to send the display output of an application to the second screen on the first display of the host named `norbert`?
- A. `-display norbert:1.2`
 - B. `-display norbert:2.1`
 - C. `-display norbert:0.1`
 - D. `-display norbert:1.0`
14. Which terminal emulator was designed to work with the Enlightenment window manager?
- A. `rxvt`
 - B. `aterm`
 - C. `xterm`
 - D. `Eterm`
15. The _____ file stores a user's personal changes to X applications written with the X Toolkit.
16. Which key combination returns the user to the X environment from a text console?
- A. `Ctrl-Alt-F7`
 - B. `Ctrl-Alt-Backspace`
 - C. `Ctrl-Alt-F6`
 - D. `Ctrl-Alt+`
17. To allow the host `dag` to connect to your X server, you would enter the command _____.
18. Which command would you use to quickly log into the system named `norbert` that is running XDM?
- A. `X -broadcast`
 - B. `X -indirect norbert`
 - C. `X -query norbert`
 - D. `startx -broadcast`

19. Which entry in the `Xaccess` file would block the host `norbert` from using the chooser?

- A. `norbert NOCHOOUSER *`
- B. `!norbert CHOOUSER brain`
- C. `-norbert CHOOUSER brain`
- D. `* NOCHOOUSER brain`

20. Which entry in the `Xaccess` file needs to be modified to enable remote login for XDM?

- A. `EnableRemote.Request 0`
- B. `DisplayManager.Enable 0`
- C. `DisplayManager.request 0`
- D. `DisplayManager.requestport 0`

Scenarios

1. You have an X application that does order entry and tracking for your company. You do not want to spend a lot of money to put computers in your warehouses and retail counters. How could X help you solve this problem?
2. What do you need to do if you upgrade the video card in your system to one that does not use your old X server?

Lab Exercises

Lab 9-1 Remote X applications

In this lab you will practice displaying back X applications across the network. In this example we assume that the X server system has the IP of 192.168.0.1, and the system that will run the client application has 192.168.0.2. Adjust these addresses as necessary for your lab setup.

1. Start X on the X server system.
2. Open a terminal emulator window, such as `xterm`.
3. Run the command `xhost +192.168.0.2` to allow the client system to display applications on your X server.
4. Telnet to the client system.

5. Execute the command `export DISPLAY=192.168.0.1:0.0` to change the display system for X clients.
6. In the Telnet session, execute the command `xeyes`. This is a small toy application to display a pair of eyes that follow the cursor around. If you do not have this application, try running `xterm`.
7. Open another terminal window.
8. Execute the command `xhost -192.168.0.2` to disable the other system from connecting to your X server.
9. Try executing the client application again from the client system and see which error message you get.

Lab 9-2 Configuring XDM for remote login

In this lab you will practice configuring XDM for remote login. We use the same example IP addresses as in the previous lab. The client system will be logging into the server system across the network.

1. On the server system, make the `/etc/X11/xdm/Xaccess` file writeable by root by typing `chmod u+w /etc/X11/xdm/Xaccess`.
2. Open the `/etc/X11/xdm/Xaccess` file in an editor such as `vi`.
3. Add a line similar to the following to the end of your `Xaccess` file:

```
192.168.0.2      CHOOSEr 192.168.0.1
```
4. Save the file and exit.
5. Make the `/etc/X11/xdm/xdm-config` writeable by root by typing `chmod u+w /etc/X11/xdm/Xaccess`.
6. Open the `/etc/X11/xdm/xdm-config` file in an editor such as `vi`.
7. Comment out the line that looks like this:

```
DisplayManager.requestPort:      0
```
8. Log into the console of the client system.
9. Change the system to text login mode if needed.
10. Try logging into the remote system directly by typing:

```
X -query 192.168.0.1
```
11. Log out of that session.
12. Now try to bring up a chooser window by typing:

```
X -indirect 192.168.0.1
```
13. Log into the system.

Answers to Chapter Questions

Chapter Pre-Test

1. The `startx` command is used to start X from the command line.
2. The `XF86Config` file is used to configure the X server.
3. The `startx` script executes the `xinit` process.
4. The window manager is responsible for window borders and desktop icons.
5. Fonts are configured in the `Files` section of the `XF86Config` file.
6. The `DISPLAY` variable defines where X clients are displayed.
7. The `.Xdefaults` file is used to change a user's X application configurations.
8. The XDM server lets the system boot to a GUI login session.
9. The `xinit` process executes the `.xinitrc` script from a user's home directory.
10. The oldest terminal emulator for X is `xterm`.

Assessment Questions

1. **A.** When `SuperProbe` examines the video hardware, it may cause the system to stop responding. The `startx` command is used to start the X Window System. The other tools should not cause any stability problems. See the “Detecting video hardware” section for more information.
2. **D.** The X server installed varies between video chipsets. Window managers and desktop environments do not depend on a certain video chipset. See the “Versions of XFree86” section for more information.
3. **C.** The X server file is linked to `/etc/X11/X`. Answers A and D are invalid. See the “Versions of XFree86” section for more information.
4. **D.** The `Pointer` section contains the mouse configuration settings. The other options are invalid. See the “Pointer” section for more information.
5. **B.** `XF86Setup` provides an easy to navigate graphical interface for creating the `XF86Config` file. `SuperProbe` is not a configuration tool. `xf86config` is much harder to use with its hard-to-navigate interface. See the “Using XF86Setup” section for more information.
6. **B.** The other commands do not exist. See the “ServerFlags” section for more information.
7. **C.** `Identifier` is referenced from the `Screen` section and is required. The other answers are optional. See the “Monitor” section for more information.

8. **A.** SuperProbe will look for the video chipset type installed in the system. The other tools do not detect video hardware. See the “Detecting video hardware” section for more information.
9. `mkfontdir /usr/X11R6/lib/X11/fonts/myfonts`. The `mkfontdir` command must be run whenever new fonts are added to the system. See the “Configuring fonts” section for more information.
10. `.xinitrc`. This file is executed by `xinit`. See the “The `xinitrc` and `.xinitrc` files” section for more information.
11. **C.** The `xdm-config` file must be changed to tell XDM to listen for queries. See the “Configuring XDM” section for more information.
12. **B.** The XDM service uses an `Xresources` file to customize the login display. See the “Xresources” section for more information.
13. **C.** Remember that the display and screen numbers start at 0, and the order is display first, then screen. See the “Configuring remote clients” section for more information.
14. **D.** `Eterm` is the terminal emulator that goes with Enlightenment. See the “Using a terminal emulator” section for more information.
15. `.Xdefaults`. This file holds Xresource entries that customize X apps for that user. See the “Customizing X applications” section for more information.
16. **A.** Ctrl-Alt-F1 through Ctrl-Alt-F6 take you to various text consoles, and Ctrl-Alt-F7 brings you back to the X environment. See the “Using special keys” section for more information.
17. `xhost +dag`. Host authentication is configured using the `xhost` command. See the “Configuring X security” section for more information.
18. **C.** This command connects directly to the specified system, without going through the chooser. See the “Logging into remote systems” section for more information.
19. **B.** The `!` at the beginning of the line is used to block hosts. See the “Configuring XDM to provide a chooser” section for more information.
20. **D.** The other options are invalid. Without changing this option XDM will not allow connections. See the “Configuring XDM to provide a chooser” section for more information.

Scenarios

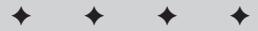
1. A good solution would be to deploy cheap workstations to the warehouses and retail counters. The order entry application could be served from a high-speed server to the cheap clients. Since the application is stored in a single location, the support costs would also be reduced.

2. The first step is to install the correct X server for the new card. This can either be done from the provided binary tar file from the XFree FTP site, or from a package file from your distribution. If you have more than one X server installed make sure that the `/etc/X11/X` file points to the correct one. Finally, make any changes to the Device section of the `XF86Config` file that may be needed for the new X server.

Administering Linux

The third part of this book covers the topics useful for administering Linux systems. This includes the various aspects of managing users as well as the tasks required on the system. Chapter 10 covers user and group administration including the files and utilities used to create users and configure options for the user's environment. Chapter 11 provides information on administering the system including system logging, jobs, backups, and process management. Chapter 12 introduces printing, the tools used to create and manage printers, and the files used when printing on a system. Kernel management is introduced in Chapter 13 along with the tools used to configure modules and the system kernel. This part ends with Chapter 14 in which shells and scripts are covered with details on the various variables and files used to work with shells and scripts.

P A R T



In This Part

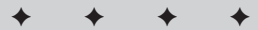
Chapter 10
Managing Users and Groups

Chapter 11
Administering the System

Chapter 12
Printing

Chapter 13
Working with the Kernel

Chapter 14
Using Shells and Scripts



Managing Users and Groups

EXAM OBJECTIVES

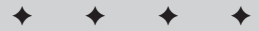
Exam 101 ♦ Linux Certification, Part 1

2.11 Administrative Tasks

- **Manage users and group accounts and related system files.** Add, remove, suspend user accounts, add and remove groups, change user/group info in passwd/group databases, create special purpose and limited accounts. Includes commands `useradd`, `userdel`, `groupadd`, `gpasswd`, `passwd`, and file `passwd`, `group`, `shadow`, and `gshadow`.
- **Tune the user environment and system environment variables.** Modify global and user profiles to set environment variable, maintain `skel` directories for new user accounts, place proper commands in `path`. Involves editing `/etc/profile` and `/etc/skel/`.

10

CHAPTER



CHAPTER PRE-TEST

1. Which file contains user names and home directories?
2. Which command(s) are used to create a new user account and assign a password?
3. Which rights are assigned to the file containing shadow passwords?
4. Where would you place files so that they would be copied to all users' home directories?
5. Global environment settings are made using which file?
6. Which user account has full access to the Linux system?
7. Which command and options are used to delete a user's account and home directory?
8. Which default user has no home directory or shell?
9. How can the shell be configured to look different for the root user?
10. What group is assigned to new user accounts when none is specified?

One key function of systems administration involves managing users and groups. Creating and managing these accounts is an essential task required when administering any system. Without users, no need for a systems administrator would exist. This chapter covers the special user accounts that are created by default and their uses. Also covered here are the tools used for creating, modifying, and deleting user and group accounts. Finally, the chapter discusses how to create global environment settings for all users of the system. The information in this chapter will definitely be used when working as a systems administrator and when taking the exam.

Special Users

During installation of a Linux system, several special user accounts are created automatically. These accounts are used for special functions on the system and should not be removed. The first example is the *root* account. This account has full access to the system and must be closely guarded. The *nobody* account is another example. This account is used for services and has no shell or home directory assignments. Finally we discuss the *bin* account, which has the home directory of */bin* with no shell assignment.

root

The root account, also known as superuser, is the most privileged account on a Linux system. This account gives you the ability to administer the system, including adding accounts, changing user passwords, examining log files, installing software, changing file permissions, and so on.

When using this account, you must be as careful as possible. The root account has no security restrictions imposed upon it. This means it is easy to perform administrative duties without hassle. However, the system assumes you know what you are doing and will do exactly what you request. Therefore, it is easy, with a mistyped command, to wipe out crucial system files.

When you are signed in as root, or acting as root, the shell prompt displays *#* as the last character if you are using *bash*. This is to serve as a warning to you of the absolute power of this account.

The rule of thumb is never sign in as root unless absolutely necessary. While root, type commands carefully and double-check them before pressing Return. Sign off from the root account as soon as you have accomplished the task you signed on for. Finally, be extra careful to keep the password secure!



The root account is required for many administrative functions on the system. It is an important practice not to log into the system as root, however. It is better to simply `su`, or switch user, to root when you need to perform these functions and then log out of this account as soon as they are complete.

nobody

The nobody user account is useful for running services on the system. It has no home directory or shell assignment. If this account is compromised, the services running the account will be affected but the rest of the system will be secure.

bin

The bin special account is assigned the home directory of `/bin`. It is used to secure essential binaries on the system. No shell is assigned to the bin special account. The account is created by default and should be left intact.

Manually Adding Users and Groups

Objective

2.11 Administrative Tasks

- **Manage users and group accounts and related system files.** Add, remove, suspend user accounts, add and remove groups, change user/group info in `passwd/group` databases, create special purpose and limited accounts. Includes commands `useradd`, `userdel`, `groupadd`, `gpasswd`, `passwd`, and file `passwd`, `group`, `shadow`, and `gshadow`.

The Linux kernel itself treats users as numbers. Each user is identified by a unique integer, the *user id* or *uid*, because numbers are faster and easier for a computer to process than textual names are. A separate database outside the kernel assigns a textual name, the *user name*, to each user id. The database contains additional information as well. Each user is assigned a user id number when the account is created. This number can be automatically generated, or it can be manually assigned. Along with the user id a group id is also assigned. Users are a member of one default group and may manually switch to other groups. These processes are covered later in the chapter.

To create a user, you need to add information about the new user to the user database and create the new user's home directory. It may also be necessary to educate the user and set up a suitable initial environment for him or her.

Most Linux distributions come with a program for creating accounts. Several such programs are available. Two command-line programs that can be used are `adduser` and `useradd`. Whatever the program, the result is that there is little, if any, manual

work to be done. Even if the details are many and intricate, these programs make everything seem trivial. In this chapter, we cover using the tools just mentioned as well as some of the manual processes of account creation.

Storing user information

The basic user database is stored in the `/etc/passwd` file, which lists all valid user names and their associated information. The file has one line per user name and is divided into seven colon-delimited fields:

- ♦ User name, up to eight characters maximum. The user name is case-sensitive and is usually all lower-case.
- ♦ Password, in encrypted form. If shadow passwords are used, an x will appear here.
- ♦ Numeric user id.
- ♦ Numeric group id. This usually matches the user id.
- ♦ Full name or other description of account.
- ♦ The user's home directory, which is usually `/home/username`.
- ♦ Login shell (program to run at login).

Any user on the system may read the password file, so that they can, for example, learn the name of another user. This means that the password (the second field) is also available to everyone. The password file encrypts the password, so in theory there is no problem. However, the encryption is breakable, especially if the password is weak (for example, it is short or it can be found in a dictionary). Therefore, it is not a good idea to have the password in the password file.

Linux systems have *shadow passwords* to help solve this problem. This is an alternative way of storing the password: the encrypted password is stored in a separate file, `/etc/shadow`, which only root can read. The `/etc/passwd` file only contains a special marker, displayed as an x, in the second field. Any program that needs to verify a user is `setuid`, which allows the program to run as a specified user id, and can therefore access the shadow password file. Normal programs, which only use the other fields in the password file, can't get at the password.



Shadow passwords are discussed in more detail later in this chapter and in Chapter 17.

The `/etc/passwd` file contains a number of user and service accounts. The service accounts are installed by default to allow the various services to function properly. Following is a sample `/etc/passwd` file:


```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/adm:
lp:x:4:7:lp:/var/spool/lpd:
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:
news:x:9:13:news:/var/spool/news:
uucp:x:10:14:uucp:/var/spool/uucp:
operator:x:11:0:operator:/root:
games:x:12:100:games:/usr/games:
gopher:x:13:30:gopher:/usr/lib/gopher-data:
ftp:x:14:50:FTP User:/home/ftp:
nobody:x:99:99:Nobody:/:
xfs:x:43:43:X Font Server:/etc/X11/fs:/bin/false
named:x:25:25:Named:/var/named:/bin/false
postgres:x:26:26:PostgreSQL Server:/var/lib/pgsql:/bin/bash
jason:x:500:500:Jason Nash:/home/jason:/bin/bash
angie:x:501:501:Angie Nash:/home/angie:/bin/bash2
```



Exam Tip Be sure you understand all of the various information stored in the `/etc/passwd` file. This file will appear on the exam in a variety of possible questions.

Storing group information

The `/etc/group` file is used to store the groups that are used on a Linux system. Along with each group, the password, group id number, and group members are stored in this file. This file is similar to the `/etc/passwd` file in appearance.

Picking numeric user and group ids

On most systems it doesn't matter what the numeric user and group ids are, but if you use the Network File System (NFS), you need to have the same uid and gid on all systems. This is because NFS also identifies users with the numeric uids. If you aren't using NFS, you can let your account creation tool pick them automatically.

However, you should try to avoid reusing numeric uids (and textual user names), because the new owner of the uid (or user name) may get access to the old owner's files (or mail, or whatever).

Creating a user by hand

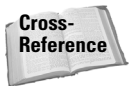
To create a new account manually, follow these steps:

1. Edit `/etc/passwd` and add a new line for the new account. Be careful with the syntax. You should make the password field be `*` so that it is impossible to log in.
2. Similarly, edit `/etc/group`, if you need to create a new group as well.
3. Create the home directory of the user with `mkdir`. This is often done using the `/home/username` structure.
4. Copy the files from `/etc/skel` to the new home directory.
5. Fix ownerships and permissions with `chown` and `chmod`. The `-R` option is most useful. The correct permissions vary a little from one site to another, but usually the following commands do the right thing:

```
cd /home/newusername
chown -R username.group .
chmod -R go=u,go-w .
chmod go= .
```

In this example the `go=` statements are used to set the group and owner permissions for the files.

6. Set the password with the `passwd` command. This command is used at the command line and allows passwords to be set and changed.



Information on the `chown` and `chmod` utilities is presented in Chapter 6.

After you set the password in the last step, the account will work. You shouldn't set it until everything else has been done, or else the user may inadvertently log in while you're still copying the files.

It is sometimes necessary to create dummy accounts that are not used by people. For example, to set up an anonymous FTP server (so that anyone can download files from it, without having to get an account first), you need to create an account called `ftp`. In such a case, it is usually not necessary to set the password (last step above). Indeed, it is better not to, so that no one can use the account unless they first become root, since root can become any user.

Managing Users and Groups

When managing user and group accounts, you need to perform several tasks: creating, modifying, and deleting accounts. Several tools are used for these tasks. Another important part of user and group management is password management. Both users and groups can be assigned passwords to ensure that only the authorized users can access these accounts. It is possible to make these passwords even more secure using shadow files, as discussed in later in this chapter, and in Chapter 17. These files use restricted access rights so that only the root user can access the files.

Managing User and Group Accounts

The first part of user management is user and group account management. Only the root user can create and modify accounts. It is important to understand the tools that are used for these tasks.

Adding a new user

The `useradd` command creates a new user account using the values specified on the command line and the default values from the system. The new user account will be entered into the system files as needed, the home directory will be created, and initial files copied, depending on the command line options. The version provided with Red Hat Linux will create a group for each user added to the system, unless `-n` option is given. The options that apply to the `useradd` command are shown in Table 10-1.

Table 10-1
Options Used with `useradd`

Option	Use
<code>-c comment</code>	The new user's password file comment field.
<code>-d /home_dir</code>	The new user will be created using <code>home_dir</code> as the value for the user's login directory. The default is to append the login name to the <code>default_home</code> assigned for the system and use that as the login directory name.
<code>-e date</code>	The date on which the user account will be disabled. The date is specified in the format YYYY-MM-DD.
<code>-f days</code>	The number of days after a password expires until the account is permanently disabled. A value of 0 disables the account as soon as the password has expired, and a value of -1 disables the feature. The default value is -1.
<code>-g group</code>	The group name or number of the initial login group.
<code>-G groups</code>	A list of groups that the user is also a member of.
<code>-k file</code>	Specifies alternative file to use in place of <code>/etc/skel</code> .
<code>-m</code>	Creates the user's home directory if it doesn't already exist.
<code>-n</code>	Prevents a group being created with the user's account name.
<code>-r</code>	Creates a system account with the special privileges assigned.
<code>-p passwd</code>	The password encrypted using the hash method.
<code>-s</code>	Specifies the user's shell account.
<code>-u uid</code>	The user's numeric user id.

Following is an example of the creation of a user account using some of the options covered in Table 10-1. In this example, the account is created with a user id of 722, a home directory of /angie, and a default group of nash.

```
# useradd angie -u 722 -d /angie -g nash
```

Deleting a user

The `userdel` command is used to remove the specified user account from the `/etc/passwd` and `/etc/shadow` files. The user can't be logged into the system when you delete the account. The `-r` option is used to delete the user's home directory and the files it contains. Other files on the system must be found and deleted manually.



There is only one option to know for the `userdel` command so be sure to know the option and its function.

The following is an example of using the `userdel` command to remove the user account `angie` and the associated home directory.

```
# userdel angie -r
```

Modifying a user account

The `usermod` command is used to modify the user's account in one of a number of ways. Several options are used to control the modifications that occur; these options are covered below in Table 10-2, and most are the same as the options used with `useradd`. As with `userdel`, the `usermod` command can't be used on an account if the user is logged in.

Table 10-2
Options Used with `usermod`

Option	Use
<code>-c comment</code>	The new user's password file comment field.
<code>-d /home_dir</code>	The new user will be created using <code>home_dir</code> as the value for the user's login directory. The default is to append the login name to <code>default_home</code> and use that as the login directory name.
<code>-e date</code>	The date on which the user account will be disabled. The date is specified in the format YYYY-MM-DD.
<code>-f days</code>	The number of days after a password expires until the account is permanently disabled. A value of 0 disables the account as soon as the password has expired, and a value of -1 disables the feature. The default value is -1.

Continued

Table 10-2 (continued)

Option	Use
-g <i>group</i>	The group name or number of the initial login group.
-G <i>groups</i>	A list of groups that the user is also a member of.
-l <i>name</i>	Changes the user's login name.
-m	Creates the user's home directory if it doesn't already exist. Will also move an existing home directory to a new location.
-s	Specifies the user's default shell.
-u <i>uid</i>	The user's numeric user id.
-L	Locks a user's password disabling it.
-U	Unlocks a user's password.

The following is an example of using the `usermod` command to unlock the password for the user `angie`.

```
# usermod angie -u
```

Adding a new group

The `groupadd` command is used to create new group accounts using the options covered in Table 10-3. These options are used to set the default values for the group created.

Table 10-3
Options Used with `groupadd`

Option	Use
-g <i>gid</i>	Specifies the group account id.
-r	Specifies that a system account be created.
-f	Forces the command to run without returning an error code even if an existing group with that name or id already exists.
-o	Allows duplicate group id numbers to be created.

The following is an example of the use of this command in which a group by the name of `linuxchix` is created with a `gid` of 721:

```
# groupadd -g 721 linuxchix
```

Deleting a group

Groups are removed using the `groupdel` command. This functions similarly to the `userdel` command and removes the group from the `/etc/group` file on the system. This command cannot be used to remove a member's primary group. It is important to verify that there are no members requiring access to a group before it is removed.

The following shows an example of the correct use of this command to remove the group `chicks`:

```
# groupdel chicks
```

Modifying a group

A group's settings can be changed using the `groupmod` command. This command is used with the `-g gid` option to change the group id number. The `groupmod` command functions similarly to the `usermod` command. Following is an example of the correct use of this command to change a group id number.

```
# groupmod nash -g 400
```

Viewing group membership

The `/etc/group` file contains information about which groups exist on the system along with which users are members of these groups. Users can view their group memberships using the `groups` command, and the root account can view any user's groups using this command. As you can see from the sample below, not all accounts are associated with groups. Some of these special accounts are used for various programs or services and don't require group membership. A sample `/etc/groups` file is shown here:

```
root:x:0:root
bin:x:1:root,bin,daemon
daemon:x:2:root,bin,daemon
sys:x:3:root,bin,adm
adm:x:4:root,adm,daemon
tty:x:5:
disk:x:6:root
lp:x:7:daemon,lp
mem:x:8:
kmem:x:9:
wheel:x:10:root
mail:x:12:mail
news:x:13:news
uucp:x:14:uucp
man:x:15:
games:x:20:
gopher:x:30:
dip:x:40:
ftp:x:50:
```

```
nobody:x:99:
users:x:100:
floppy:x:19:
xfs:x:43:
named:x:25:
console:x:31:
utmp:x:22:
pppusers:x:44:
popusers:x:45:
slipusers:x:46:
postgres:x:26:
slocate:x:21:
jason:x:500:
angie:x:501:
```

Assigning and Using Passwords

Password management is another essential key to user administration. Once passwords are created, they should be changed by the users on a regular basis. It is also possible to create more secure passwords using shadow passwords, as covered in Chapter 17. The password management tools are covered below.

Changing a password

To change a password on behalf of a user, first sign on or `su` to the root account. Then type, `passwd user` (where *user* is the user name for the password you are changing). The system will prompt you to enter a password. Passwords do not echo to the screen when you enter them.

You can also change your own password, by typing `passwd` (without specifying a user name). You will be prompted to enter your old password for verification and then a new password.

Using shadow passwords

As we mentioned earlier in the chapter, traditional Linux systems keep user account information, including one-way hashed passwords, in a text file called `/etc/passwd`. This file is used by many tools to display file ownerships and so on by matching user ids with the user's names, so the file needs to be world-readable. Consequently, this file can be something of a security risk.

To mitigate this risk, you can use another method of storing account information, the *shadow password* format. As with the traditional method, this method stores account information in the `/etc/passwd` file in a compatible format. However, the password is stored as a single `x` character. A second file, called `/etc/shadow`, contains encrypted passwords as well as other information such as account or password expiration values and so on. The `/etc/shadow` file is readable only by the root account and is therefore less of a security risk.



It is important to understand the permissions assigned to the `/etc/shadow` file. This will appear as a possible exam question.

As with the `passwd` file, each field in the `shadow` file is also separated with a colon. The fields are as follows:

- ♦ User name, up to eight characters. Case-sensitive, usually all lowercase. A direct match to the user name in the `/etc/passwd` file.
- ♦ Password, thirteen characters encrypted. A blank entry indicates a password is not required to log in, and a `*` entry indicates the account has been disabled.
- ♦ The number of days since the password was last changed.
- ♦ The number of days before the password may be changed (0 indicates it may be changed at any time).
- ♦ The number of days after which the password must be changed (99999 indicates the user can keep his or her password unchanged for many, many years).
- ♦ The number of days in advance to warn the user of an expiring password (7 for a full week).
- ♦ The number of days after the password expires that the account is disabled.
- ♦ The number of days since an account has been disabled.
- ♦ A reserved field for possible future use.

Following is a sample `/etc/shadow` file which has been modified with incorrect password information; as you can see, the encrypted password is shown with the user name:

```
root:aer9a9reau34iajfkjeri73qa97:11324:0:99999:7:::
bin*:11263:0:99999:7:::
daemon*:11263:0:99999:7:::
adm*:11263:0:99999:7:::
lp*:11263:0:99999:7:::
sync*:11263:0:99999:7:::
shutdown*:11263:0:99999:7:::
halt*:11263:0:99999:7:::
mail*:11263:0:99999:7:::
news*:11263:0:99999:7:::
uucp*:11263:0:99999:7:::
operator*:11263:0:99999:7:::
games*:11263:0:99999:7:::
gopher*:11263:0:99999:7:::
ftp*:11263:0:99999:7:::
```



```
nobody:!:11263:0:99999:7:::
xfs:!:11263:0:99999:7:::
named:!:11263:0:99999:7:::
postgres:!:11263:0:99999:7:::
jason:$WERAERREAELRAEREAFAEA83w:11324:0:99999:7:::
angie:lawekroiweriwoierowier:11324:0:99999:7:::
```

In this example, for the user `jason`, the first field in the shadow file is the user name. The second field is `jason`'s password in encrypted form. The third field is the day the password was set. This value is in the form of the number of days since January 1, 1970. The fourth field is the number of days the user must wait between password changes, in this case that is zero days. The fifth field is the number of days the password remains valid, which for `jason` is 99,999 days. The sixth field is the number of days before the password expires that the user will be warned to change the password. The user `jason` will be warned one week before the password expires. The last three fields are unused in this file. The seventh value contains the number of days after password expiration to wait before treating the login as expired. The eighth field specifies a number of days from January 1, 1970, to wait before expiring the login. The ninth field is reserved for future use.

Configuring Global and User Settings

Objective

2.11 Administrative Tasks

- Tune the user environment and system environment variables. Modify global and user profiles to set environment variable, maintain `skel` directories for new user accounts, place proper commands in `path`. Involves editing `/etc/profile` and `/etc/skel/`.

The files discussed in this section are extremely useful by allowing one location for settings and then having these settings replicated to all users that are created. Such a process greatly eases the administrative burden for configuring user accounts.

`/etc/profile`

The `etc/profile` file contains the global environment variable settings for users on the Linux system. This file sets the global variables that are defined in Chapter 2. Entries here are applied to all users unless they are overwritten by the user's individual profile. Following is a sample `/etc/profile` file:

```
# /etc/profile

# System wide environment and startup programs
# Functions and aliases go in /etc/bashrc
```

This first section introduces the file with the filename and a comment about its purpose.

```
PATH="$PATH:/usr/X11R6/bin"
```

This line appends the `/usr/X11R6/bin` directory to the `PATH` variable.

```
ulimit -c 1000000
if [ `id -gn` = `id -un` -a `id -u` -gt 14 ]; then
    umask 002
else
    umask 022
fi
```

This section configures the `ulimit` and the `umask` for the user.

```
USER=`id -un`
LOGNAME=$USER
MAIL="/var/spool/mail/$USER"

HOSTNAME=`bin/hostname`
HISTSIZ=1000

if [ -z "$INPUTRC" -a ! -f "$HOME/.inputrc" ]; then
    INPUTRC=/etc/inputrc
fi
```

This section configures variables for the user, such as the `USER`, `LOGNAME`, `MAIL`, `HOSTNAME`, `HISTSIZ`, and `INPUTRC` variables.

```
export PATH USER LOGNAME MAIL HOSTNAME HISTSIZ INPUTRC
```

The above line exports the variables so they become available to the user.



Both the `/etc/profile` file and the `/etc/skel` directory are extremely useful for configuring global settings and files for all users on the system. As an administrator you should use these whenever you wish to apply a setting to all users.

/etc/skel

When the home directory for a new user is created, it is initialized with files from the `/etc/skel` directory. The system administrator can store files in `/etc/skel` that will provide a nice default environment for users. For example, you might create a `/etc/skel/.profile` file that sets the `EDITOR` environment variable, the variable used to specify the default editor used by the system, to some editor that is friendly towards new users.

However, it is usually best to try to keep the `/etc/skel` file as small as possible, since it will be next to impossible to update existing users' files. For example, if the name of the friendly editor changes, all existing users would have to edit their

.profile files. The system administrator could try to do it automatically, with a script, but that is almost certain to break someone's file.

Whenever possible, it is better to put global configuration into global files, such as /etc/profile. This will ensure that there is no contradiction between the settings.

Following is an example of the contents of the /etc/skel directory.

```
[angie@redhat ~]$ ls -al /etc/skel
total 28
drwxr-xr-x  2 root  root    4096 Nov  1 19:01 .
drwxr-xr-x 34 root  root    4096 Jan 20 22:37 ..
-rw-r--r--  1 root  root      24 Jul 13 1994 .bash_logout
-rw-r--r--  1 root  root    230 Aug 22 1998 .bash_profile
-rw-r--r--  1 root  root    124 Aug 23 1995 .bashrc
-rwxr-xr-x  1 root  root    333 Feb 21 2000 .emacs
-rw-r--r--  1 root  root   3394 Mar  7 2000 .screenrc
```

Key Point Summary

Several essential tools and elements of user administration are covered in this chapter. Understanding the proper use of the tools and files covered here is essential. The most important of the concepts covered are as follows:

- ♦ The root account has full control of system functions.
- ♦ The nobody account has no home directory or shell assignment and is used for services on the system.
- ♦ The bin account has a home directory of /bin and no shell assignment.
- ♦ User accounts are created using the useradd command.
- ♦ User accounts are removed using the userdel command.
- ♦ User accounts are changed using the usermod command.
- ♦ The user's group can be viewed using the groups command.
- ♦ New groups are created using the groupadd command.
- ♦ The passwd command is used to assign and change user passwords.
- ♦ The /etc/passwd file contains the user account database.
- ♦ The /etc/profile file is used for global environment settings.
- ♦ The /etc/skel/ directory contains the files that are copied to all users' home directories.



STUDY GUIDE

The following questions and exercises will allow you to review the information covered in this chapter. Take your time when completing this section of the chapter, carefully reviewing any questions that you may have. Answering the question correctly is not as important as understanding the answer, so review any material that you might still be unsure of. Being comfortable with the questions and answers presented here will help you be more prepared for the certification exam questions.

Assessment Questions

1. Which of the following is not stored in the `/etc/passwd` file?
 - A. Home directory path
 - B. Shell assignment
 - C. User ID
 - D. Quotas
2. When shadow passwords are enabled the password in the `/etc/passwd` file is displayed as _____ .
3. Files located in which file are copied to users' home directories?
 - A. `/etc/group`
 - B. `/etc/passwd`
 - C. `/etc/skel`
 - D. `/etc/user`
4. Which option with the `useradd` command specifies that the home directory should be created if it doesn't already exist?
 - A. `-h`
 - B. `-m`
 - C. `-c`
 - D. `-d`

5. Which option is used with `userdel` to remove a user's home directory as the account is removed?
 - A. `-d`
 - B. `-h`
 - C. `-r`
 - D. `-e`

6. Which option with `usermod` is used to change a user's default shell?
 - A. `-d`
 - B. `-s`
 - C. `-b`
 - D. `-q`

7. Global environment settings are configured in the _____ file.

8. The default `umask` used on the system is configured in which file?
 - A. `/etc/bash`
 - B. `/etc/profile`
 - C. `/etc/passwd`
 - D. `/etc/files`

9. Which type of access is granted to all users for the `/etc/passwd` file?
 - A. Read
 - B. Execute
 - C. No access
 - D. Write

10. Which type of access is granted to other users for the `/etc/shadow` file?
 - A. Read
 - B. Execute
 - C. No access
 - D. Write

11. Which command allows users to view their group memberships?
 - A. groupadd
 - B. newgroup
 - C. groups
 - D. newgrp
12. The _____ account has full control of the system.

Scenarios

1. One of your users will be on medical leave for six months. How would you disable the user's account?
2. You need to examine all the user and group accounts on the system to ensure that there are no accounts for users who no longer need access to the system. What could you use to do this?

Lab Exercises

Lab 10-1 Manually create a new user account

In this lab you create a user account manually. This enables you to see all of the steps required when creating an account and familiarizes you with the files and directories used.

1. Edit `/etc/passwd` and add a new line for the new account. Be careful with the syntax. You should make the password field be `*`, so that it is impossible to log in.
2. Similarly, edit `/etc/group`, if you need to create a new group as well.
3. Create the home directory of the user with `mkdir`.
4. Copy the files from `/etc/skel` to the new home directory.
5. Fix ownerships and permissions with `chown` and `chmod`. The `-R` option is most useful. The correct permissions vary a little from one site to another, but usually the following commands do the right thing:

```
cd /home/newusername  
  
chown -R username.group .  
  
chmod -R go=u,go-w .  
  
chmod go=
```

6. Set the password with `passwd`.

Answers to Chapter Questions

Chapter Pre-Test

1. The `/etc/passwd` file contains the user names and home directories.
2. The `useradd` command is used to create a user account, and the `passwd` command is used to assign a password.
3. Only the root user has access to the shadow files.
4. The `/etc/skel` directory contains the files copied to all users' home directories.
5. The `/etc/profile` file contains global environment settings.
6. The root user has full access to the system.
7. The `userdel -r` command is used to delete a user's account and home directory.
8. The `nobody` user has no home directory or shell.
9. The bash prompt may be configured differently for the root user.
10. The default group assigned when none is specified is a group name the same as the user name.

Assessment Questions

1. **D.** Quota information is not stored in the `/etc/passwd` file. See the "Storing user information" section for more information.
2. **x.** Once shadow passwords are enabled the password field in the `/etc/passwd` file contains an `x` character to signify that the password is contained in the `/etc/shadow` file. See the "Storing user information" section for more information.
3. **C.** Files stored in the `/etc/skel` directory are copied to users' home directories. See the `/etc/skel` section for more information.
4. **B.** The `useradd -m` command specifies that the home directory is to be created with the user account. See the "Adding a new user" section for more information.
5. **C.** The `userdel -r` command is used to remove a user's home directory with the user account. See the "Deleting a user" section for more information.
6. **B.** The `usermod -s` command is used to change a user's default shell. See the "Modifying a user account" section for more information.

7. `/etc/profile`. Global environment settings are stored in the `/etc/profile` file. See the “`/etc/profile`” section for more information.
8. **B.** The default `umask` used on the system is configured in the `/etc/profile` file. See the “`/etc/profile`” section for more information.
9. **A.** All users are granted read access to the `/etc/passwd` file. See the “Using shadow passwords” section for more information.
10. **C.** All users are granted no access to the `/etc/shadow` file. See the “Using shadow passwords” section for more information.
11. **C.** Users can view their group memberships using the `groups` command. See the “Viewing group membership” section for more information.
12. **root.** The root account has full control of the system. See the “root” section for more information.

Scenarios

1. Lock the user’s password using the `usermod -L` command.
2. The `/etc/passwd` and `/etc/group` files can be used for viewing all user and group accounts on the system.

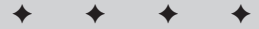
Administering the System

EXAM OBJECTIVES

Exam 101 ♦ General Linux, Part 1

2.11 Administrative Tasks

- **Configure and use system log files to meet administrative and security needs.** Configure the type and level of information logged, manually scan log files for notable activity, arrange for automatic rotation and archiving of logs, track down problems noted in logs. Involves editing `/etc/syslog.conf`
- **Automate system administration tasks by scheduling jobs to run in the future.** Use cron to run jobs at regular intervals, use `at` to run jobs at a specific time, manage cron and `at` jobs, configure user access to cron and `at` services.
- **Maintain an effective data backup strategy.** Plan a backup strategy, backup filesystems automatically to various media, perform partial and manual backups, verify the integrity of backup files, partially or fully restore backups.



CHAPTER PRE-TEST

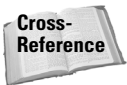
1. Which utility allows for jobs to be run on a recurring basis?
2. What signal name and number is used to kill a process?
3. Which utility allows you to run an automated job whenever system loads are low?
4. Which field in the `/etc/syslog.conf` file is used to represent the priority that is being logged?
5. Which command is used to display system messages?
6. Which type of backup is used to back up all data that is contained in `/home`?
7. How do you limit the size of `.core` files that can be created?
8. By default, who is mailed with the results of a completed `cron` job?
9. Which command would be used to create an archive file that is compressed and contains all of the files located on the `/home` partition?
10. Which command is used to restart the `cron` daemon?

This chapter covers several of the topics required for the Administrative Tasks section of the exam: log files, scheduled jobs, and backups. We discuss the configuration and usage strategies for each of these tasks, along with the utilities and parameters necessary to accomplish these tasks. Knowledge of these tools is necessary for the test and for the job of Linux systems administrator. We recommend that you try out each of these tools before taking the exam.

As a system administrator of a Linux network, you will be responsible for the maintenance of the systems. Several tools, which are useful for ensuring that the system runs as expected, can help you with this job, alerting you to potential problems, and helping you to recover your system from failures. The tools that are covered in this chapter include logging, automating jobs, and backups.

Starting and Stopping Daemons

Daemons, also known as servers, are programs that run in the background without any interactivity. They simply wait around for clients to send them requests that they respond to based on their configuration and functionality. They usually spend far more time waiting for connections than actually doing work. But, while waiting they use little resources. Daemons are usually identified by having a *d* at the end of their program name; examples are `ftpd` and `httpd`. Daemons are usually either configured to start when the computer boots or they can be started manually.



Starting daemons at boot is covered in Chapter 8.

Using the `/etc/rc.d` scripts

Daemons often use initialization scripts that are used to start, stop, and restart the server. The location of these scripts can vary according to where the program installs them. The most common location for these scripts on Red Hat–based systems is the `/etc/rc.d/init.d` directory. Usually these scripts have the same name as the daemon they control. For example, the `httpd` server may be started using the `/etc/rc.d/init.d/httpd` script.

Before trying to start the daemon manually, you should verify that it is not already running. The `ps -aux` command can be used to display the currently running processes. Another way that you can check whether the daemon is running is with the script. The `status` option will display whether the daemon is running, for example:

```
/etc/rc.d/init.d/httpd status
httpd (pid 2947 2946 2945 2944 612) is running...
```

Other options used with the script are `start`, `stop`, and `restart`. The options do exactly what they say—either start, stop, or restart the daemon. The options work regardless of where the script is stored. The correct use is always as follows:

```
/path/script option
```

Whenever a change has been made to the configuration of a daemon, it's a good idea to restart the daemon so that the changes will take effect.

Using the kill command

Daemons can also be stopped and restarted manually using the `kill` command. The syntax for this command is as follows:

```
kill -SIGNAL 'cat /var/run/script.pid'
```

You can also use the following:

```
kill -SIGNAL PID
```



The `script.pid` file is created by the `rc` script when the daemon is started. Not all scripts create this file.

Using `cat` with the `/path/script.pid` outputs the process identification number, so the commands are actually the same. Signals can be specified using name or number. Table 11-1 lists the more common signals that you may use.

Table 11-1
Common Signals Used with kill

<i>Signal</i>	<i>Meaning</i>
SIGHUP or -1	Used to restart the process. Often used after a configuration change is made.
SIGKILL or -9	Used to kill the daemon without trying to stop it nicely.
SIGTERM or -15	Used to stop the daemon.



Be careful when using `kill -9`, as it does not give the process a chance to gracefully shut down. Loss of data may occur.

System Logging

Objective
2.11 Administrative Tasks

- **Configure and use system log files to meet administrative and security needs.** Configure the type and level of information logged, manually scan log files for notable activity, arrange for automatic rotation and archiving of logs, track down problems noted in logs. Involves editing `/etc/syslog.conf`

System logs are a network administrator's best friend. They can be used to verify that the system is running properly and to alert you to potential problems with the system. You can use logs to monitor many different aspects of the system. Specific programs, such as Apache, can use logging to report on a variety of items from usage levels to attempts at intrusion. Logs written by applications that are running are known as *software logs*. The Linux system itself also uses logging to help you track information on the functions of your system, such as disk usage. These logs are generated by kernel and system processes and are known as *system logs*. The usefulness of logging is really up to the administrator who configures and maintains the logs. Logs should be configured to show the important data while leaving out the trivial, or else they can be difficult to read and use. Also, if the logs aren't monitored properly they are useless, since you may miss the warning signs of a potential problem.

The daemon responsible for system logging is `syslogd`. It is normally started by the `rc` scripts as the system boots. Because it is started by `rc` scripts, you can also manually control the daemon using the following syntax:

```
/etc/rc.d/init.d/syslogd option
```

The options are `start`, `stop`, and `restart`.

Signals can also be used to control `syslogd`, which means you can use the `kill` command, as described in the previous section.

Options used by `syslogd` are listed in Table 11-2.

Table 11-2
Options Used with syslogd

<i>Option</i>	<i>Meaning</i>
<code>-f</code>	Specifies another configuration file to use instead of <code>syslog.conf</code> .
<code>-h</code>	Forwards messages received by other hosts.
<code>-l hostnames</code>	Logs messages with simple hostname rather than with fully qualified domain name. Multiple names are separated with a colon.

Continued

Table 11-2 (continued)

<i>Option</i>	<i>Meaning</i>
<code>-m interval</code>	Specifies the time between two marked lines; the default value is 20 minutes.
<code>-r</code>	Used to allow receipt of network messages.



Note that many distributions ship without the `-r` switch on `syslogd`. By default you cannot use them as network-logging servers.

Configuring system logging

System logging is configured using the `/etc/syslog.conf` file. This file specifies where the information is stored. The default `/etc/syslog.conf` file installed contains many comments which are specified by a `#` symbol. These comments are useful for providing instruction on how to change the file should you need to do so, or for indicating when changes were made, or by whom. Entries are made using two columns that are separated using tabs. The first column specifies what to log by using two fields separated by periods. The first field is the facility, which designates the type of program that generates the message. The facility terms are listed in Table 11-3. The second field is the priority, which specifies the logging level necessary for the facility. Only the messages with a priority equal to or greater than the set priority are logged. If you want to log only the specified priority level, you can put an equal (`=`) sign in front of the priority. To log all priority levels but the one specified use an exclamation (!) sign before the priority. Wildcards (`*`) can be used in the `/etc/syslog.conf` file to refer to all facilities or all priorities. If you want no priority levels of the facility to be logged you can use the entry `none`. When specifying multiple facilities to be logged with the same priority level you can separate the facility names with a comma.

Table 11-3
Facilities to Use in the `syslog.conf` File

<i>Facility</i>	<i>Use</i>
<code>auth</code>	Authorization programs
<code>authpriv</code>	Private authorization programs
<code>cron</code>	<code>cron</code> and <code>at</code>
<code>daemon</code>	System daemons
<code>kern</code>	System kernel
<code>lpr</code>	Printer daemon

Facility	Use
mail	Mail system
news	News system
syslog	Syslog daemon
user	General log
uucp	UUCP system
local0-local7	Local software

The second column in the `/etc/syslog.conf` file specifies the location of the log file. Placing a dash (-) directly before the path to the file can increase the speed of logging by not syncing the log file after each write. This could lead to information not being written to the log in the event of a system crash. If you want multiple facilities to log to the same file you can place the facilities on the same line and separate them with a semicolon.

The following is an example of a `syslog.conf` file:

```

auth,authpriv.*           /var/log/auth.log
cron.*                     -/var/log/cron.log
daemon.*                   -/var/log/daemon.log
kern.*                     /var/log/kern.log
mail.*                     -/var/log/mail.log
lpr.*                      -/var/log/lpr.log
user.*                     -/var/log/user.log
news.=notice               -/var/log/news/news.not
mail.warn                  -/var/log/mail/mail.warn

```

As you can see in the first line, all priorities of events for either authorization programs or private authorization programs are logged to `/var/log/auth.log`. All cron, daemon, kern, mail, lpr, and user events are logged to files as well. All of these logs except for `/var/log/auth.log` and `/var/log/kern.log` are written without synchronization after each write. News events of the priority notice are logged without synchronization as well. Finally, mail events with a priority of warn or higher are logged without synchronization to `/var/log/mail/mail.warn`.

The default location for system log files is `/var/log/messages`, but they can be written to any location that you specify in the `/etc/syslog.conf` file. Log messages are written along with the date and source on a single line.

The log files `/var/log/wtmp`, `/var/log/utmp`, and `/var/log/lastlog` contain information on users logged into the system. This information can be useful to examine if you suspect intruders on your system. The log files can be examined manually or by using the `lastlog` command. These files are discussed in Table 11-4.

Table 11-4
Default Log Files

<i>Log Filename</i>	<i>Purpose</i>
<code>/var/log/wtmp</code>	Stores the login times and duration for each user. This information is used by the command <code>last</code> . The <code>last</code> command shows all users listed in the file, and users may be listed more than once.
<code>/var/log/utmp</code>	Stores information on currently logged-in users. The <code>who</code> , <code>w</code> , and <code>finger</code> commands use this information. The <code>who</code> and <code>w</code> commands display who is currently logged into the system. The <code>w</code> command also shows the time that the user logged in. The <code>finger</code> command displays the comment field from the user's entry in <code>/etc/passwd</code> .
<code>/var/log/lastlog</code>	Stores login times for users. The <code>lastlog</code> command uses this information. If a user has never logged in before, that is also displayed. By default, entries are ordered by <code>userid</code> . The <code>-t</code> option can be used to show all login times for the specified number of days. The <code>-u</code> option displays the user's last login.



Be sure to know which command uses which file for the exam.

Rotating system logs

Depending on which data you log, your log files can grow to be quite large seemingly overnight. *Log rotation* can be very helpful in maintaining a manageable size for your log files. You simply need to decide which data you want to save and move it to another location. Moving the data to a new location will free up space in the default location, but you still have to deal with the old log files taking up space on your system. It is useful to have a standard regarding which log files to keep and for how long. Depending on your situation, you may want to track certain data over long periods of time. Whatever your needs may be, Linux includes a utility to help you manage your log files. The program `/usr/sbin/logrotate` is used to rotate the log files. Older logs are copied to a new location and new log files are created. The default log file used by `logrotate` is the `/etc/logrotate.conf` file. This file is well documented with comments and is configured by default to suit many administrative needs. However, other configuration files can be used by using the following command:

```
logrotate [options] config_file
```

The configuration file can contain both global and local options. Global options apply to all log files, and local options apply only to a specific log file. The local options overwrite those set globally. Multiple configuration files can be used, but if a conflict occurs, then the instructions contained in the last file overwrite those in the first. The following is a sample of a `logrotate.conf` file:

```
# see "man logrotate" for details
# rotate log files weekly
weekly

# keep 4 weeks worth of backlogs
rotate 4

# send errors to root
errors root

# 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 lastlog or wtmp -- we'll rotate them here
/var/log/wtmp {
    monthly
    create 0664 root wtmp
    rotate 1
}

/var/log/lastlog {
    monthly
    rotate 1
}

# system-specific logs may be configured here
```

The configuration files use many different commands to provide instructions for log rotation. The commands used are listed and explained in Table 11-5.

Table 11-5
logrotate Commands

Command	Use
<code>compress</code>	Compresses old logs using <code>gzip</code> .
<code>copytruncate</code>	Copies old logs and then truncates them.
<code>create</code>	Creates permissions for files. If none are specified, the permissions of the old files are used.
<code>daily</code>	Rotates log files daily.
<code>delaycompress</code>	Compresses log files during the next rotation.
<code>errors <i>mailto</i></code>	Mails error messages to the specified address.
<code>ifempty</code>	Rotates empty log files.
<code>include <i>filename</i></code>	Used to read the <i>filename</i> into the config file.
<code>mail <i>mailto</i></code>	Specifies address to mail log files to when they are deleted.
<code>monthly</code>	Rotates log files monthly.
<code>nocompress</code>	Does not compress log files.
<code>nocopytruncate</code>	Does not copy and truncate.
<code>nocreate</code>	Specifies not to create a new log file.
<code>nodelaycompress</code>	Compresses log files immediately.
<code>noolddir</code>	Log files are not moved to another directory.
<code>noifempty</code>	Empty log files are not rotated.
<code>olddir <i>directory</i></code>	Old log files are moved to a specified directory.
<code>postrotate</code>	Script is run after log is rotated.
<code>prerotate</code>	Script is run before log is rotated.
<code>rotate <i>n</i></code>	Specifies the number of old log files to keep.
<code>size <i>n</i></code>	Rotates when log files reach the size specified. Can be used with <i>n</i> for bytes, <i>k</i> for kilobytes, and <i>M</i> for megabytes.

Identifying problems using log files

Simply keeping log files won't solve or prevent problems on your systems. For log files to be useful you must regularly examine the data that is logged. The `dmesg` command can be used to display system messages. The command is used with the following syntax:

```
dmesg [options]
```

The `dmesg` command displays system messages from the kernel ring buffer. Using the `-n level` option will display the messages of a specific level. The message levels are defined in the `kernel.h` file. The `-c` option will clear the buffer after displaying the contents. If both are used the first option is ignored.

The following is an example of output received after typing the `dmesg` command:

```
Linux version 2.2.15-4mdk (root@linux.the-nashes.net
30 (prerelease)) #2 Wed Jun 21 22:22:09 EDT 2000
Detected 750033537 Hz processor.
ide_setup: hdd=ide-scsi
Console: colour VGA+ 80x25
Calibrating delay loop... 1494.22 BogoMIPS
Memory: 257488k/262080k available (1208k kernel code
 72k init, 0k bigmem)
Dentry hash table entries: 32768 (order 6, 256k)
Buffer cache hash table entries: 262144 (order 8, 10
Page cache hash table entries: 65536 (order 6, 256k)
VFS: Diskquotas version dquot_6.4.0 initialized
CPU: L1 I Cache: 64K L1 D Cache: 64K
CPU: L2 Cache: 512K
CPU: AMD AMD Athlon(tm) Processor stepping 01
Checking 386/387 coupling... OK, FPU using exception
Checking 'hlt' instruction... OK.
POSIX conformance testing by UNIFIX
mtrr: v1.35a (19990819) Richard Gooch (rgooch@atnf.c
PCI: PCI BIOS revision 2.10 entry at 0xf4d0
PCI: Using configuration type 1
PCI: Probing PCI hardware
Linux NET4.0 for Linux 2.2
Based upon Swansea University Computer Society NET3.
NET4: Unix domain sockets 1.0 for Linux NET4.0.
NET4: Linux TCP/IP 1.0 for NET4.0
IP Protocols: ICMP, UDP, TCP, IGMP
TCP: Hash tables configured (ehash 262144 bhash 6553
Initializing RT netlink socket
Starting kswapd v 1.5
Detected PS/2 Mouse Port.
Serial driver version 4.27 with MANY_PORTS MULTIPORT
ttyS00 at 0x03f8 (irq = 4) is a 16550A
ttyS01 at 0x02f8 (irq = 3) is a 16550A
pty: 256 Unix98 ptys configured
apm: BIOS version 1.2 Flags 0x07 (Driver version 1.1
Real Time Clock Driver v1.09
RAM disk driver initialized: 16 RAM disks of 4096K
Uniform Multi-Platform E-IDE driver Revision: 6.30
ide: Assuming 33MHz system bus speed for PIO modes;
VP_IDE: IDE controller on PCI bus 00 dev 39
VP_IDE: not 100% native mode: will probe irqs later
VT 8371
Chipset Core ATA-66
```

```

Split FIFO Configuration:  8 Primary buffers, thresh
                          8 Second. buffers, thresh
ide0: BM-DMA at 0xd000-0xd007, BIOS settings: hda:DMA, hdb:DMA
ide0: VIA Bus-Master (U)DMA Timing Config Success
ide1: BM-DMA at 0xd008-0xd00f, BIOS settings: hda:DMA, hdb:DMA
ide1: VIA Bus-Master (U)DMA Timing Config Success
hda: WDC AC418000D, ATA DISK drive
hdb: KENWOOD CD-ROM UCR-421 V221G, ATAPI CDROM drive
hdc: WDC AC418000D, ATA DISK drive
hdd: PLEXTOR CD-R PX-W1210A, ATAPI CDROM drive
ide0 at 0x1f0-0x1f7,0x3f6 on irq 14
ide1 at 0x170-0x177,0x376 on irq 15
hda: WDC AC418000D, 17206MB w/1966kB Cache, CHS=2193
hdc: WDC AC418000D, 17206MB w/1966kB Cache, CHS=3496
hdb: ATAPI 68X CD-ROM drive, 2048kB Cache
Uniform CDROM driver Revision: 2.56
Floppy drive(s): fd0 is 1.44M
FDC 0 is a post-1991 82077
md driver 0.90.0 MAX_MD_DEVS=256, MAX_REAL=12
raid5: measuring checksumming speed
raid5: MMX detected, trying high-speed MMX checksum
   pII_mmx   : 1738.122 MB/sec
   p5_mmx   : 2184.273 MB/sec
   8regs    : 1136.142 MB/sec
   32regs   :  872.871 MB/sec
using fastest function: p5_mmx (2184.273 MB/sec)
scsi : 0 hosts.
scsi : detected total.
md.c: sizeof(mdp_super_t) = 4096
Partition check:
  hda: hda1
  hdc: hdc1 hdc2 < hdc5 >
autodetecting RAID arrays
autorun ...
... autorun DONE.
VFS: Mounted root (ext2 filesystem) readonly.
Freeing unused kernel memory: 72k freed
Adding Swap: 212144k swap-space (priority -1)
scsi : 0 hosts.
tulip.c:v0.91g-ppc 7/16/99 becker@cesdis.gsfc.nasa.g
eth0: Lite-On 82c168 PNIC rev 32 at 0xe400, 00:A0:CC
eth0: MII transceiver #1 config 3000 status 7829 ad
mouse0: PS/2 mouse device for input0
mice: PS/2 mouse device common for all mice
nessud uses obsolete (PF_INET,SOCK_PACKET)
end_request: I/O error, dev 02:00 (floppy), sector 0
end_request: I/O error, dev 02:00 (floppy), sector 0

```

As you can see, information is displayed on the Linux version, hardware detection, settings, and errors. This information can be extremely useful when trying to configure hardware that is not working correctly. In this output you can see the checks that occur during the system boot. All hardware is checked during this time, as well as the CPU type and cache. The amount of memory is verified along with the hard disks that are installed on the system. You can find information here concerning the protocols that are loaded, the adapters such as the network card, the mouse, and the file systems. If there are errors or problems with your system, the `dmesg` utility provides thorough data for locating the source of the problem.

**Tip**

The `dmesg` output is very long; therefore you should use the `less` or `more` commands.

Scheduling Jobs



2.11 Administrative Tasks

- Automate system administration tasks by scheduling jobs to run in the future. Use `cron` to run jobs at regular intervals, use `at` to run jobs at a specific time, manage `cron` and `at` jobs, configure user access to `cron` and `at` services.

As a system administrator there are some tasks and programs that you will need to run on a regular basis. Some of these tasks, such as file backups, are best performed at night when there are no users connected to the systems. It is helpful if these jobs can run automatically, allowing you to spend your time on less mundane tasks. Linux has a couple of tools to fill this need. If you need to run a job once, the `at` and `batch` commands are useful. For jobs that need to be scheduled to run on a recurring basis, Linux provides the `cron` utility.

Using the `at` utility

For jobs that need to be scheduled to run once the `at` command allows you to specify a time in the future for this to occur. The `at` command is used with the following syntax:

```
at [options] time
```

The options used with the `at` command are listed in Table 11-6.

Table 11-6
Options Used with at

<i>Option</i>	<i>Meaning</i>
-b	Allows jobs to be run when the system load is low. This is the same as the <code>batch</code> command.
-c <i>job</i>	Outputs the jobs listed to standard output.
-d	Removes an <code>at</code> job.
-f <i>FILE</i>	Used to specify a file containing the jobs that are to run.
-l	Lists all jobs for the user.
-m	Specifies that the user is to be mailed when the job completes.
-v	Shows the time the job is scheduled to run.

The `at` command allows fairly complex time specifications. It accepts times of the form HH:MM to run a job at a specific time of day. If that time is already past, the next day is assumed. You can also specify midnight, noon, or teatime (4 p.m.), and you can have a time of day suffixed with AM or PM for running in the morning or the evening. You can also say what day the job will be run, by giving a date in the form *month day* with an optional year, or giving a date of the form MMDDYY or MM/DD/YY or DD.MM.YY. The specification of a date must follow the specification of the time of day. You can also give times such as `now + count time-units`, where the *time-units* can be minutes, hours, days, or weeks. You can also use `today` to run the job at a certain time today, and `tomorrow` to run the job tomorrow.

After entering the `at` command you will be given an `at` prompt where you enter the commands you wish to run. You can enter multiple commands here, which are run sequentially in the order listed. When you have finished entering commands, press Ctrl-D to save the job. This also saves the current environment including the directory and uses this information when the job is run. When running a job that has been saved to a file, include the correct paths so that the job will run correctly.

If you wish to control who has access to the `at` command on your system, you can use the `/etc/at.allow` and `/etc/at.deny` files. When checking for these files, the system first checks for the `/etc/at.allow` file. If this file does not exist, the `/etc/at.deny` file is checked. Because the `/etc/at.allow` file is checked first, if a user is listed there they will be allowed to run the `at` command regardless of whether they are listed in the `/etc/at.deny` file. It is a good idea to use either one or the other to manage your users' rights to the `at` command. If you wish to allow all of your users to run the `at` command, simply empty the `/etc/at.deny` file. If neither `/etc/at.allow` or `/etc/at.deny` exists, only root is allowed to run the `at` command.

If you decide to allow users to run the `at` command, you will probably want to monitor them from time to time. You can use the options and commands listed in this section to view and delete `at` jobs. Viewing jobs as a user shows you the jobs for that user, viewing them as `root` shows the jobs for all users.

Using the batch utility

Jobs that are run as `batch` jobs are run when system load is low. If no time is specified, the job will run as soon as the system load drops. When a time is specified, the job will run at a time after the time specified when system resources are low. System load is determined by the system using the `/proc/loadavg`, and jobs are run when this is below 1.5. Batch jobs aren't ended when a user logs off of a system, like other `at` jobs. Instead they will run until the system is shut down.

The commands `at` and `batch` provide a good way to run jobs automatically. The `batch` command also provides the functionality of running jobs when system resources are low. The limitation with these commands is that they are single use. The command has to be run for every instance that you wish to run the job. For recurring jobs, the `crond` utility may provide a better solution for you.

Using the crond daemon

Some jobs, such as backups, need to run on a recurring basis. Instead of having to create an `at` job for each occurrence of the job, you can use the `cron` utility. This utility allows you to specify jobs to be run automatically based on the configuration file. `Crontab` files containing the commands and time for execution are used by this utility. Each user can have a `crontab` file, which will execute as the user. In this file all blank lines, spaces, and tabs beginning lines, and lines beginning with the `#` character are ignored. The lines beginning with the `#` character are comment lines; these lines are not allowed on the same lines as commands or variable settings. The environment variable of `SHELL` is automatically set to `/bin/sh` and the `LOGNAME` and `HOME` variables are configured according to the user's settings in the `/etc/passwd` file. The `SHELL` and `HOME` variables can be overwritten within the `crontab` file, but the `LOGNAME` cannot.

Access to the `cron` command is controlled using the `allow` and `deny` files. If the `allow` file exists users must be listed in that file in order to be allowed to use this command. When no `allow` file is found the system searches for a `deny` file. All users except those listed in the `deny` file will be allowed to use the command. If neither of these files exists, either only `root` will be allowed to use the command or all users will be able to access it, this varies according to the system configuration. On Debian systems all users will be allowed access. The options used with the `cron` command are listed in Table 11-7.

Table 11-7
Options Used with cron

<i>Option</i>	<i>Meaning</i>
-u <i>user</i>	Runs the command as the specified user.
-l	Displays the current crontab file.
-r	Removes the current crontab.
-e	Edits the current crontab file using the editor specified by the VISUAL or EDITOR environment variables.

The time and date fields along with their values are illustrated in Table 11-8.

Table 11-8
Time and Date Fields

<i>Field</i>	<i>Allowed Values</i>
minute	0–59
hour	0–23
day of month	1–31
month	1–12 (or names)
day of week	0–7 (0 or 7 is Sunday, or use names)

A field may be an asterisk (*), which always stands for “first-last”.

Ranges of numbers are allowed. Ranges are two numbers separated with a hyphen. The specified range is inclusive. For example, 8-11 for an hour entry specifies execution at hours 8, 9, 10, and 11.

Names can also be used for the month and day of week fields. Use the first three letters of the particular day or month (case doesn’t matter). Ranges or lists of names are not allowed.

An example crontab entry is as follows:

```
MAILTO=betty
30 1 * * sun mv mybackup.tar.gz myoldbackup.tar.gz
0 2 * * sun tar -czf mybackup.tar.gz /home/betty/*
```

This will cause user `betty` to be e-mailed regardless of the user running the job. The file `mybackup.tar.gz` will be renamed to `myoldbackup.tar.gz` at 1:30 a.m. every Sunday. At 2:00 a.m. every Sunday the contents of `/home/betty` will be archived and compressed into the file `mybackup.tar.gz` using the `tar` utility.

`cron` searches `/var/spool/cron` for `crontab` files that are named after accounts in `/etc/passwd`; `crontab` files found are loaded into memory. `cron` also searches for `/etc/crontab` and the files in the `/etc/cron.d/` directory, which are in a different format. These are usually used for system maintenance. `cron` then wakes up every minute, examining all stored `crontab` files, checking each command to see if it should be run in the current minute. When executing commands, any output is mailed to the owner.



Remember that `at` is for jobs that need to be run once, and `cron` is for jobs that are to run repeatedly.

Performing Backups

Objective

2.11 Administrative Tasks

- **Maintain an effective data backup strategy.** Plan a backup strategy, backup filesystems automatically to various media, perform partial and manual backups, verify the integrity of backup files, partially or fully restore backups.

Regardless of the function of your server, it will probably be used to store data. Whether the data is e-mail, Web server configuration, system configuration, or company data, you will want to make sure this data is safeguarded in case of accidents. Many threats to your system data exist. Hardware failures, especially of hard disks, are common. Natural disasters such as hurricanes, floods, tornadoes, and fires are always a threat. On top of this you have to consider malicious attacks from both inside and outside the network. Users may also unintentionally delete data they still require. Any of these can destroy your data, if not your entire system. Therefore, it's important to have a clear strategy of how you can recover from one of these events. Often this is considered after the fact and in most cases that is too late. You can be sure that at least one of these events will happen and the best you can do is to be prepared.

Planning the backups

You must consider several things when planning your backups. You must decide what to back up, how often it should be done, what media should be used, and when to do it. Each of these questions helps to shape your backup strategy and can determine what steps are required for restoring data. Give and take is involved in planning your strategy, and balancing all of the many needs to find a solution that works best in each situation is important.

When deciding what to back up, you should do an inventory of what is on your system. The core system and application files will change only when you upgrade them, so it isn't necessary to back them up nightly. It is important to examine whether it is quicker to recreate the files or restore them. If you decide that it's quicker to restore these files than to reinstall the system and applications, you simply need to back them up whenever you make a change.

Next you want to examine your data files. The frequency of change here might vary greatly across your system. You might consider placing the more frequently changed files, such as users' home directories, on a separate partition since these files need to be backed up often. When deciding on your backup schedule and procedure, be sure that your users are aware of the policies. They must know where to put important information so that it will be backed up. They also need to know how frequently backups are performed. This education is an important part of the backup strategy that is often overlooked. It does little good to create a good backup strategy when the files that need to be saved aren't included in the backup procedure.

As you see, "what" and "how often" depend largely on the data stored on your system. "When" is another important component to consider. Generally, you want to perform backups when system load is low. For many systems this is in the middle of the night. It is generally best not to run a backup job at the same time other jobs such as virus scans are running. Systems that provide services across the country or the world require special consideration. What might seem like a good time locally to back up the system isn't always the best time for the system. This not only limits the time that backups can be started but limits the length of the backup process. If you discover that the system is in the best condition to for the backup to run between 2 a.m. and 5 a.m. you must discover how much data can be backed up during that time frame. This alone can determine a large part of your backup strategy.

Backups are performed to provide recovery of data. The goal is to get the data back in place quickly. Some strategies can aid with this. RAID (Redundant Array of Inexpensive Disks) systems can help with this in the case of a drive failure. Redundant disk systems allow for a drive to be lost with little or no downtime. However, a good RAID system is still no replacement for a careful backup strategy. Drive failures are just one of the many possible causes of data loss.

In some cases you will discover that the amount of time spent backing up data is inversely proportional to the amount of downtime required to restore a system. The more time you spend backing up data, the quicker it can be restored.

Backup methods

The different types of methods are distinguished by what is backed up. They each have different requirements for restoration. The common types of backups are as follows:

- ♦ Copy
- ♦ Full
- ♦ Partial
- ♦ Incremental
- ♦ Differential

Copy

This method is exactly what it sounds like. Data is simply copied to save it. This type of backup can be creating an archival copy of a file on the same disk, as well as copying files to other disks. You often copy key files before they are changed to ensure that you can fall back to a working configuration in the event of problems. Many people are familiar with copying files to floppy disks and more recently CD-ROMs. This is a quick way to save files.

Simply copying entire systems files as a backup method can be cumbersome. Another problem with copying is that you might have issues doing a simple file copy if the file is in use. This can cause key files to be missed.

Full backup

A full backup involves backing up every file on the system to the specified backup location. This method has several advantages and disadvantages that can help you determine whether it is appropriate for use on your systems.

One large disadvantage is the time required to perform the backup. Depending on the size of your system and the backup media you select, it can be very time consuming to back up every file on the system. In fact, it may be impossible to do this in the time window that the system is available for backups. Another way a full backup can be a problem is when the backup is larger than the media that is used to store it. This may require that someone be available to change the media during the backup job so that it can complete.

One advantage to a full backup is ease of restore. Having all files contained on one backup job allows for the quickest restore process. This can reduce the amount of downtime required to repair your system.

Partial backup

Similar to a full backup, a partial backup doesn't include all files on the system. This allows you to select which partitions and directories you want to backup. This allows the backup jobs to be smaller and quicker than a full backup allows. This works well when the files that change frequently are maintained on separate partitions or directories.

When restoring an entire system using a partial backup, you must first restore or reinstall the system and applications and then restore the partial backup information. This can make the restore process a bit longer than with a full backup due to the time saved during the backup process.

Differential backup

This method backs up the files changed since the last full. This allows for a quicker restore process than that used with an incremental backup. To restore, you simply restore the full backup and then the most recent differential backup. As you may have guessed this method usually requires more time to back up files than the incremental backup method.

As with incremental backups, this method is often used in conjunction with full backups. Often a full backup is run either weekly or monthly and differential backups are performed between the full backups. With time the length of the backup process will grow as the number of files that have been changed grows.

It is important not to use both incremental and differential backup methods on the same files. This can make restoration difficult and confusing.

Backup media

No discussion of backups would be complete without discussing the backup media used to store backups. Many choices for backup media are available, from the simple and inexpensive methods such as floppies and CDs to the more expensive devices such as tape autoloaders that allow for greater speed and capacity.

One of the primary considerations to factor into your backup media decision is Linux support for the hardware. There is support for a variety of removable media such as Zip drives, tape drives, and writeable CD-ROM drives. The method you choose varies according to the size, speed, and cost requirements that you have.

Zip drives provide more space than a floppy diskette, up to 250MB, and can be used on IDE and SCSI interfaces, which make them quicker as well. Writeable and rewriteable CD-ROM drives also operate on these interfaces and can store up to 700MB. For systems with large amounts of data, tape drives are a good choice. They can provide space ranging from several hundred megabytes to more than 70GB on a single tape. When combined with an autoloader, they can provide enough storage space for even the largest systems.

Other considerations

Whichever backup method and media you decide on, it's important to follow a few basic precautions. Periodically perform test restores to ensure that in case of an actual failure the backup methods that you are using are reliable. Tapes and other media sometimes go bad, so it is important to verify that everything is working properly.

Another precaution to take is to move some of the media off-site. A natural disaster can destroy your backup media along with the servers. Many services will pick up and store your data at a secure location. This provides another layer of insurance to ensure that the system can be restored.

You may also want to utilize a collection of tapes that are used according to a rotation scheme. If you wish to maintain a collection of the backups that have occurred over the past month, you will need a collection of tapes large enough to support this. It is also key to label the tapes properly, store them neatly in a specific location, and maintain a backup log. This will make it much easier when the time comes to restore files.

Backup commands

There are several commands in Linux that can aid with backups. The most commonly used of these commands are the following:

- ♦ tar
- ♦ cpio
- ♦ compress
- ♦ gzip
- ♦ gunzip



See Chapter 6 for more information on compression tools.

Using the tar utility

The `tar` utility is used to combine many files into one. It saves the directory structure of the files within the archive file. Another benefit with most recent versions of the utility is the ability to compress the new file when it is created. These make it a popular solution for backups.

The `tar` utility is used as in the following example. In this example the contents of the `/home/me` directory are archived and compressed in the `myback.tar.gz` file.

```
tar -cvzf mybackup.tar.gz /home/me
```

The `tar` utility uses both functions and options. At least one function must be used and can be used with many options. The function must come before the options. The functions that can be used are listed in Table 11-9.

Table 11-9
Functions Used with `tar`

<i>Function</i>	<i>Meaning</i>
-A	Adds additional <code>tar</code> files to an archive file.
-c	Creates a new archive file.
-d	Compares the files stored in an archive.
-delete	Deletes one or more files from the archive.
-r	Places new files at the end of the archive.
-t	Lists the files included in the archive.
-u	Appends files that are newer than the version already included in the archive.
-x	Extracts files stored in the archive.

The options used with `tar` are listed in Table 11-10.

Table 11-10
Options Used with `tar`

<i>Option</i>	<i>Meaning</i>
-b	Specifies the block size to be used when creating the archive.
-C	Changes to the specified directory.
-f	Specifies the file or device to utilize.
-h	Specifies to archive the target file on symbolic links.
-i	Ignores the EOF zeros in archive files.
-I	Uses the <code>bzip2</code> compression utility in combination with the <code>tar</code> utility.
-K	Starts working with the archive at the specified file.
-l	Remains in the local file system when archives are created.
-L	Specifies the size of the file to write before switching to a new tape or file.
-m	Ignores the file modification times when extracting files.
-M	Specifies that the archive consists of multiple volumes.
-N	Only files newer than the date specified are stored.

Option	Meaning
-O	Used to extract files to standard output.
-p	Preserves permissions on extracted files.
-P	Uses absolute paths for files contained in the archive so that the leading slashes aren't stripped from the files.
-s	Specifies that the file order is maintained.
-T <i>FILE</i>	Extracts or creates the specified file.
-v	Lists the files verbosely as they are processed.
-V	Used to specify the archive volume name.
-w	Prompts for confirmation when attempting to complete tasks.
-W	Verifies the files contained in the archive.
-X	Excludes the specified files from the archive.
-Z	Utilizes the <code>compress</code> utility when working with the archive files.
-z	Utilizes the <code>gzip</code> utility when working with the archive files.

So if you use the example from the beginning of this section, `tar` will create a file called `mybackup.tar.gz` containing all of the files and directories located in `/home/me`. The new file will be compressed using `gzip` and the files are listed verbosely as they are archived. If you enter this command, you will see a message informing you that the leading `/` is not included in the archive file. This is to prevent the accidental overwriting of files when the archive is extracted. Because the directory structure is contained in the archive file, it is important to pay close attention to how files are created and extracted. Using the path `/home/me` when creating the file will result in the extracted files to be placed in a directory called `/home/me` off your present working directory. So if you extract these files from `/home/me`, they will be placed in `/home/me/home/me`.

When writing to a device, such as a tape device, keep a few things in mind. You will need to specify the blocking factor using the `-b` option. The number of blocks multiplied by 512 bytes specifies the amount of data to be written to the tape at a time. This option is needed only for writing the file, not for reading it.

When creating a `tar` file larger than the backup media, you must specify the size of the media and that the file will be contained on multiple media. You will then be prompted to manually change the media when it is filled.



Not all versions of `tar` support `gzip` or `bzip` compression.

Using the cpio utility

The `cpio` command (copy in and out) is used to copy to and from archive. The archive can be a file, device, or a pipe. It has three modes of operation:

- ♦ **Copy-out**— This mode is used to copy files to an archive. A list of filenames is read, one per line, from the standard input and writes to standard output. This mode is specified with the `-o` option.
- ♦ **Copy-in**— This mode is used to copy files out of an archive or lists the archive contents that are read from standard input. This mode is specified with the `-i` option.
- ♦ **Copy-pass**— This mode is used to copy files from one directory tree to another. This mode is specified with the `-p` option.

Other options used with this command are listed in Table 11-11.

Table 11-11
Options Used with cpio

<i>Option</i>	<i>Meaning</i>
<code>-a</code>	Resets the access time on the files.
<code>-A</code>	Used with copy-out mode to append to the archive file. Must be used with the <code>-o</code> or <code>-F</code> options.
<code>--block-size=BLOCK-SIZE</code>	Specifies the block size to be used when creating files; the specified number is multiplied by 512 bytes.
<code>-d</code>	Specifies that leading directories be created.
<code>-E FILE</code>	Used in copy-in mode to list or extract files matching the pattern specified.
<code>-f</code>	Copies only the files that don't match the specified patterns.
<code>-F</code>	Specifies an archive filename to be used rather than the standard input or output. This is also used when archiving to a tape device located on another computer.
<code>-0</code>	Lists of filenames that are terminated by the null character are read. This option is used in copy-out and copy-pass modes.
<code>-r</code>	Allows files to be renamed interactively.
<code>-u</code>	Replaces all files without prompting.
<code>-v</code>	Verbosely lists the files that are processed.

Using the compress utility

The `compress` utility is one of the first utilities for compressing files. Files compressed with this utility end in `.Z`. The original file is replaced by the compressed version. To compress a file use the following syntax:

```
compress filename
```

To uncompress the file use the following syntax:

```
uncompress filename.Z
```

With the ability of `tar` to compress an archive file when it is created, `compress` is rarely used.

Using the gzip utility

The `gzip` utility is used to compress files. As you saw earlier, `tar` has the ability to do this automatically when creating an archive file. The command can also be run separately to compress any file. When a file is created using `gzip`, the original file is replaced by a new file with the same name and a `.gz` extension. The new file is created with the same permissions of the original file. Parameters for this command are listed in Table 11-12.

Table 11-12
Options Used with gzip

<i>Option</i>	<i>Meaning</i>
-c	Specifies that files be concatenated before compression, which allows for better compression.
-d	Decompresses the file.
-f	Forces compression and decompression of files.
-h	Displays the help information for the <code>gzip</code> utility.
-l	Lists all compression and file information for compressed files.
-r	Recursively searches directories for files to compress or uncompress depending on the options specified.
-t	Tests the files' integrity.
-v	Verbosely lists files as they are compressed or uncompressed.



While most people use `gunzip` to decompress a file, remember for the exam that you can also use `gzip -d`.

Using the gunzip utility

The `gunzip` utility is used to decompress a `gzip` or `compress` file. The command is used in much the same way and when the file is uncompressed, the compressed file is deleted and replaced with the uncompressed version. This can be done automatically with the `tar` utility. When files are uncompressed with the `tar` utility, the original compressed file is left intact. The options for the `gunzip` utility are the same as the options for the `gzip` utility.

Limiting Core Dump Files

One other potential problem to check for involves *core dump files*. These files contain everything in RAM when a problem causes a system core dump, which is a fatal system error. Because everything contained in RAM is included in these files they can become quite large. These can quickly become a major source of wasted disk space.

Although the files may be useful to programmers, they provide no use if you aren't debugging a problem. If you wish to view the contents of these files, you can use the GNU Debugger, also known as `gdb`.

A couple of methods are used to deal with core files. One method used is `cron jobs`. A job can be created either to inform you of the location and size of the `.core` files, these are the system core dump files, on your system or to automatically delete them. The `ulimit -c` command is another way to manage these files. This command allows you to specify the maximum size of `.core` files that can be created. The syntax for using this command is as follows:

```
ulimit -c limit
```

Without the *limit* specification, the current setting is displayed. The *limit* values are in 1024-byte increments.

Using a combination of these methods can help keep these files in check.

Key Point Summary

This chapter covers many of the administration topics that are not only important for the exam but also for the job of system administrator. All of these utilities can be used together to provide the administrator a high level of control over their systems. Although it is important to know these for the exam, the real test of how well you can utilize these will come in real life.

- ♦ Daemons are programs that run without interactivity. They can be started using scripts or the `kill` command.

- ♦ System logging allows you to track vital system information. These logs can alert you to potential problems with the system and with security. It is important to configure them correctly and monitor them carefully.
- ♦ Processes can be scheduled to run automatically using `at`, `batch`, and `cron`. The `at` and `batch` commands allow for a one-time execution while `cron` is used for recurring jobs.
- ♦ Backups are often run as `cron` jobs using a variety of utilities. These utilities include `tar`, `cpio`, `compress`, `gzip`, and `gunzip`.
- ♦ Core dump files are managed using `cron` jobs and the `ulimit -c` command.



STUDY GUIDE

The following questions and exercises will allow you to review the information covered in this chapter. Take your time when completing this section of the chapter, carefully reviewing any questions that you may have. Answering the question correctly is not as important as understanding the answer, so review any material that you might still be unsure of. Being comfortable with the questions and answers presented here will help you be more prepared for the certification exam questions.

Assessment Questions

1. Which of the following will restart the `crond` daemon running as process id 446?
 - A. `kill -9 446`
 - B. `kill -SIGHUP 446`
 - C. `kill -1 446`
 - D. `kill -SIGTERM 446`
 - E. Answers A and D
 - F. Answers B and C
 - G. None of the above
2. Choose all of the correct facility names used in `/etc/syslog.conf`.
 - A. `auth`
 - B. `access`
 - C. `authpriv`
 - D. `kern`
 - E. `panic`
 - F. `notice`
3. Where are system startup scripts stored on a Red Hat–based system?
 - A. `/etc`
 - B. `/etc/rc.d/rc3.d`
 - C. `/etc/init.d`
 - D. `/etc/rc.d/init.d`

4. Which parameter allows `syslog` to receive network messages?
 - A. `-n`
 - B. `-r`
 - C. `-m`
 - D. `-s`

5. Where are user `crontab` files stored?
 - A. `/etc/crontab`
 - B. `/var/spool`
 - C. `/var/spool/cron`
 - D. `~/.cron`

6. Which command(s) will decompress the file named `file.gz`?
 - A. `uncompress file.gz`
 - B. `gunzip file.gz`
 - C. `tar zxvf file.gz`
 - D. `gzip -d file.gz`

7. The proper way to `tar` files without compressing is:
 - A. `tar -cvf backup.tar /home/Jason`
 - B. `tar -cvzf backup.tar /home/Jason`
 - C. `tar -zvf backup.gz /home/Jason`

8. To limit the amount of space core dumps may occupy use:
 - A. `climit`
 - B. `ulimit`
 - C. `quota`
 - D. `~/.ulimit`

9. The location of system logs can be specified in the file `/var/log/syslog.conf` file.
 - A. True
 - B. False

10. _____ is used to automatically run a job when system load is low.

11. Which type of backup is used to back up only the files that have changed since the last full backup?
 - A. Full
 - B. Partial
 - C. Incremental
 - D. Differential

12. Which mode of `cpio` is used to copy files from an archive file?
 - A. Copy-out
 - B. Copy-in
 - C. Copy-pass

13. The `tar` utility is capable of using the `gzip` utility to compress files during archival.
 - A. True
 - B. False

14. The default location of the system log containing login times for users is:
 - A. `/var/log/utmp`
 - B. `/var/log/users`
 - C. `/var/log/lastlog`
 - D. `var/log/login`

15. The _____ command is used to rotate system logs.

16. The `gunzip` utility is capable of uncompressing files that were compressed with both the `gzip` and `compress` utilities.
 - A. True
 - B. False

17. Which option on the `cron` command is used to display entries in the current crontab?
 - A. `-e`
 - B. `-r`
 - C. `-l`
 - D. `-s`

18. In the `/etc/syslog.conf` file, which character is used to increase the speed of system logging?
- A. *
 - B. =
 - C. !
 - D. -
19. Which file is used to specify message levels that are displayed using the `dmesg` command?
- A. `dmesg.conf`
 - B. `kernel.h`
 - C. `kernel.conf`
 - D. `dmesg.h`
20. Core files are viewed using which of the following?
- A. `coreview`
 - B. `debug`
 - C. `gdb`
 - D. `dumpview`

Scenarios

1. You have installed a new CD-ROM drive in to your workstation, but it does not appear to function when you boot. Where might you look for more information?
2. You need to back up the user directory on your server every night automatically. What are efficient ways to accomplish this?
3. The last network administrator at your company did not do a good job of removing accounts that were no longer needed. It is now your responsibility to clear out the user database. What could you use to make this job easier?

Lab Exercises

Lab 12-1 Set up a cron job

This lab sets up a cron job to update the `locate` database at 3 a.m.

1. Log in to the system as root.
2. Run the command `crontab -e`.
3. Hit `a` to add the following line to create the cron entry: `00 3 * * * updatedb`
4. Hit `Esc`, and then `:wq` to exit.

Lab 12-2 Create a backup

In this lab, you will create a backup of your home directory.

1. Log into the system as the user whose home directory you want to backup.
2. Use the following command to create the backup: `tar cvf filename.tar /home/username`
3. Make changes to one or more files in the home directory.
4. Use the following command to show which files were changed: `tar dvf filename.tar`
5. Update the changed files by using the following: `tar uvf filename.tar /home/username`
6. To make the backup smaller use the following: `gzip filename.tar`



The `tar` command offers many parameters to help with backups. Using them with some scripting and redirection you can create some powerful backup scripts to handle things like differential backups.

Answers to Chapter Questions

Chapter Pre-Test

1. The `cron` utility allows you to automatically run a recurring job.
2. The `sigkill` and `-9` signals are used to kill a process.
3. The `batch` command allows for jobs to automatically run when system load is low.

4. The second field in the `/etc/syslog.conf` file is used to specify the priority of the messages logged.
5. The `dmesg` command is used to display system messages.
6. A partial backup can be used to back up all files located in `/home`.
7. The `ulimit -c` command is used to limit the size of `.core` files.
8. The person who creates the `cron` job is mailed the results.
9. Archived files can be created and automatically compressed using the command `tar czf backup.tar.gz /home`.
10. The `/etc/rc.d/init.d/crond restart` command will restart the `cron` daemon.

Assessment Questions

1. **F.** The `kill` command can use signal names and numbers to stop, start, and restart processes. See the “Using the `kill` command” section for more information.
2. **A, C, and D.** `auth`, `authpriv`, and `kern` are all facility names used in `/etc/syslog.conf`. See the “Configuring system logging” section for more information.
3. **D.** On Red Hat–based systems, system scripts are stored in `/etc/rc.d/init.d`. See the “Using the `/etc/rc.d` scripts” section for more information.
4. **B.** The `syslog -r` command is used to allow `syslog` to receive network messages. See the “System Logging” section for more information.
5. **C.** User `crontab` files are stored in `/var/spool/cron`. See the “Using the `crond` daemon” section for more information.
6. **B and D.** Files can be decompressed using the commands `gunzip` and `gzip -d`. See the “Using the `gzip` utility” and “Using the `gunzip` utility” sections for more information.
7. **A.** The command `tar -cvf backup.tar /home/Jason` will create a new file named `backup.tar` and list the files during creation. The `-z` option compresses the files. See the “Using the `tar` utility” section for more information.
8. **B.** The size of `.core` files can be limited using `ulimit`. See the “Limiting Core Dump Files” section for more information.
9. **B.** The location of system log files can be specified in the `/etc/syslog.conf` file. See the “Configuring system logging” section for more information.
10. **batch.** The `batch` utility allows jobs to automatically be run when system load is low. See the “Using the `batch` utility” section for more information.

11. **D.** Differential backups are used to back up files that have changed since the last full backup. See the “Differential backup” section for more information.
12. **B.** The `cpio` utility will copy files from an archive when used in the copy-out mode. See the “Using the `cpio` utility” section for more information.
13. **A.** The `tar` utility can use the `gzip` utility to automatically compress and uncompress files during file creation. See the “Using the `tar` utility” section for more information.
14. **C.** The log file containing login times for users is stored in `/var/log/lastlog` by default. See the “Configuring system logging” section for more information.
15. `logrotate`. The `logrotate` command is used to rotate system logs. See the “Rotating system logs” section for more information.
16. **A.** The `gunzip` utility can compress files created with `gzip` and `compress`. See the “Using the `gunzip` utility” section for more information.
17. **C.** The `cron -l` command is used to display the current contents of `crontab`. See the “Using the `crond` daemon” section for more information.
18. **D.** The `-` character increases the speed of system logging by not synching the log after each write. See the “Configuring system logging” section for more information.
19. **B.** The `kernel.h` file is used to specify the message levels that are displayed using `dmesg`. See the “Identifying problems using log files” section for more information.
20. **C.** Core dump files can be viewed using the GNU Debugger or `gdb`. See the “Limiting Core Dump Files” section for more information.

Scenarios

1. Using the `boot log` you can see which hardware was detected and which device files they were associated with. Most likely you would use `dmesg | more` or `dmesg | grep`.
2. The easiest solution is to use a combination of `cron` and `tar`. Use `cron` to execute the job every night and `tar` to do the backup. By using the `-u` option with `tar` you can do a differential backup every night and a full backup on the weekends. With some minor scripting you can also use `gzip` to compress the archive to make it smaller.
3. The first step is to note any accounts that you do not want to delete, even though they may not have been logged in recently. Some daemons use special accounts that are not shown as logged in. After that, you can use the `lastlog` command to see when the last login was for an account.

Printing

12

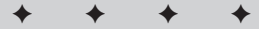
CHAPTER

EXAM OBJECTIVES

Exam 102 ♦ General Linux, Part 2

1.7 Text editing, Processing, Printing

- **Manage printers and print queues.** Monitor and manage print queues and user print jobs, troubleshoot general printing problems. Includes the commands: `lpc`, `lpq`, `lprm` and `lpr`. Includes reviewing the file: `/etc/printcap`.
- **Print files.** Submit jobs to print queues, convert text files to postscript for printing. Includes `lpr` command.
- **Install and configure local and remote printers.** Install a printer daemon, install and configure a print filter (e.g.: `apsfilter`, `magicfilter`). Make local and remote printers accessible for a Linux system, including postscript, non-postscript, and Samba printers. Involves the daemon: `lpd`. Involves editing or reviewing the files and directories: `/etc/printcap`, `/etc/apsfilterrc`, `/usr/lib/apsfilter/filter/*/`, `/etc/magicfilter/*/`, `/var/spool/lpd/*/`



CHAPTER PRE-TEST

1. Which file configures the printing system?
2. Which tool can be used to manually print a file?
3. What is used to convert different file formats to a language the printer understands?
4. Which command is used to remove a print job from the queue?
5. Which daemon is responsible for spooling print jobs to the print device?
6. Which daemon receives print jobs from clients?
7. Which `lpd` command will stop new print jobs from being spooled?
8. Which definition in the `printcap` file tells `lpd` which filter to use?
9. When using a remote network printer, where should the print filter be installed?
10. Which tool can be used to convert text files to PostScript format?

This chapter covers the installation and configuration of the printing system in Linux. One of the biggest responsibilities a network administrator handles is that of maintaining the printing environment. It does not take long for user complaints to pile up when there are problems with printing.

Installing Printers

Objective

1.7 Text editing, Processing, Printing

- **Manage printers and print queues.** Monitor and manage print queues and user print jobs, troubleshoot general printing problems. Includes the commands: `lpc`, `lpq`, `lprm` and `lpr`. Includes reviewing the file: `/etc/printcap`.
- **Install and configure local and remote printers.** Install a printer daemon, install and configure a print filter (e.g.: `apsfilter`, `magicfilter`). Make local and remote printers accessible for a Linux system, including `postscript`, `non-postscript`, and `Samba` printers. Involves the daemon: `lpd`. Involves editing or reviewing the files and directories: `/etc/printcap`, `/etc/apsfilterrc`, `/usr/lib/apsfilter/filter/*`, `/etc/magicfilter/*`, `/var/spool/lpd/*`

Printing services in Linux are among the more difficult things to configure. Several GUI utilities can ease you through the process; however, these are not covered on the LPI exam and they do not teach you the actual printing process. The GUI tools also do not help when a problem occurs and you need to troubleshoot, or do something the GUI creators did not prepare for.

Most Linux distributions ship the LPR, line printer, printing system. LPR has been around for many years and is having trouble growing and supporting the latest consumer printing technology. Several projects are aimed to enhance LPR or create new printing systems, but LPR is still very widely used. Every Linux administrator should be very familiar with its functionality and configuration.



Even though both LPR and LPRng will be covered in this chapter, only LPR will be covered on the exam.

Most distributions have just switched to the new LPRng package, LPR next generation. It is command- and file-compatible with the original LPR derived from BSD UNIX, but has some newer features. Check with your distribution to see which version you are running.



The site <http://www.linuxprinting.org> provides a lot of documentation and tools for help on configuring printing in the Linux environment.

Configuring the /etc/printcap file

The `/etc/printcap` file is used to define the printers available to the system, whether they are local or remote on the network. The format for this file is as follows:

```
printer_name|alias_name:\
:definitions:\
:last_definition;
```

Any line that does not start with a colon or a pipe designates the beginning of a printer definition. Each line of a printer definition ends in a `\`, except for the last one, which ends in a semicolon. Multiple printers can be configured in the file, each with their own definition block. A `#` designates a comment. The following is an example `printcap` file.

```
#REMOTE POSTSCRIPT 300x300 letter {} PostScript Default 1
lp|hplj:\
:sd=/var/spool/lpd/lp:\
:mx#0:\
:sh:\
:rm=192.168.1.12:\
:rp=hplj:\
:if=/usr/share/printtool/master-filter:
```

After the printer name and optional aliases, there are several definitions that configure the printer. Table 12-1 lists the most common definitions. These are probably the only ones you will ever see, but the `printcap` man page lists them all.



You should restart the `lpd` daemon any time you make a change to the `/etc/printcap` file.

Table 12-1
/etc/printcap Definitions

Definition	Function	Arguments	Example
<code>if</code>	Defines the input filter that formats the data before it is sent to the printer.	The path and filename of the filter	<code>:if=/usr/share/printtool/master-filter:</code>
<code>lf</code>	Defines the printer error log file.	The path to the log file.	<code>:lf=/var/spool/lpd/hplj/errorlog:\</code>
<code>lo</code>	Defines the lock file that is created when the printer is in use.	Path to the lock file.	<code>:lo=/var/spool/lpd/hplj/hplj.lock:\</code>

Definition	Function	Arguments	Example
mx	Defines the maximum size of the print job.	The size in blocks, :mx:#0:\ with 0 meaning unlimited.	
rm	Specifies that this printer is actually on a remote machine.	The IP address or hostname of the remote system.	:rm=192.168.1.12:\
rp	Defines the remote printer name.	The name of the remote printer.	rp=hp1j:\
sh	Does not print banner pages.	None.	:sh:\
sd	Specifies the spool directory.	The path to the spool directory for this printer.	:sd=/var/spool/lpd/lp:\

Consider the previous example file.

```
lp|hp1j:\
```

Since this line does not begin with a colon or a pipe, it is the beginning of another printer definition. The printer's name is `lp`, which is short for line printer and has an alias of `hp1j`. It is important to have a printer named `lp`, and it should be the one you use most often since this is the default printer that most applications use, unless told otherwise.

```
:sd=/var/spool/lpd/lp:\
```

This line configures the spool directory, which is where documents are stored before they are sent to the printer hardware.

```
:mx:#0:\
```

This line removes any print job size limit. Users are free to submit jobs of any length.

```
:sh:\
```

Some people like to use banner pages to separate print jobs at a busy printer. In this case banner pages have been disabled to save paper.

```
:rm=192.168.1.12:\
```


This printer is shared out from a print server on the network. This definition configures the remote address of that print server. Hostnames or IP addresses can be used, but if you use hostnames, remember that a DNS failure can stop users from printing. The advantage is that if the print server should ever move, only the DNS entry needs to be changed, and not every workstation.

```
:rp=hp1j:\
```

This line specifies the name of the printer on the remote print server.

```
:if=/usr/share/printtool/master-filter:
```

This definition specifies the print filter to use for this printer. A print filter is similar to a printer driver in other operating systems, such as Windows. It formats the data to accommodate the language that the printer can use. When most people think of printing on a UNIX system, they think of a PostScript printer. But as Linux has moved into the consumer market where most people do not have expensive PostScript printers, print filters have become very important. Several different print filters are available, and the most common are covered later in this chapter.



All lines in the `/etc/printcap` file end in `:\`, except for the last line in a printer block.

Creating the spool directory and log file

Once you edit the `/etc/printcap` file and add the necessary printer definitions, you then need to create the spool directory and log file.

Print spool directory

The print spool directory is where print jobs are sent before they are streamed to the printing device. When you manually create a printer configuration you must make this directory by hand. You should make sure to give this directory the exact path and name you defined in `/etc/printcap`. For example:

```
debian:~# mkdir /var/spool/lpd/lp
```

Next, you need to set the proper ownership and permissions on the newly created directory. Most distributions have a special user or group that owns the LPR directories and files. Usually these are called `lp`. For example:

```
debian:/var/spool/lpd# ls -l
total 5
drwx----- 2 daemon lp          232 Jan 12 12:40 lp
-rw-r--r--  1 root  lp          6 Dec 18 10:55 lp.lock
```

Check your distribution to make sure your new directory has the correct ownership and permissions. In this example you can see that Debian has user ownership of the daemon because its LPR daemon (`lpd`) runs as the user `daemon`. If your distribution has `lpd` running as `root`, give `root` ownership. Directory permissions should be set to `700` or `770`.



The spool directory must be created manually.

Log file

The optional log file defined with the `lf` option in the `/etc/printcap` must also be created manually. This can be done with the `touch` command to create an empty file. For example:

```
debian:~# touch /var/spool/lpd/hplj/errorlog
```

Set the ownership the same as the spool directory, and change the file permissions to `666`, or `rw-rw-rw`.



If you use the optional log file, you must create the file manually.

Controlling printer access

Access to printers can be restricted to certain hosts and users if needed by using the `/etc/hosts.lpd` file. To limit access to the local printer to only the three defined workstations, you would use the following:

```
dekoenigsberg.bowling.com  
mcmillan.bowling.com  
golaski.bowling.com
```

The `/etc/hosts.lpd` file can also be used to restrict access to only certain users by putting the user name after the host. For example:

```
dekoenigsberg.bowling.com greg  
mcmillan.bowling.com bret  
golaski.bowling.com lorien
```

Keep in mind that if even one host or user is listed in the `hosts.lpd` file, then only the listed host(s) can access the printer. You must add all hosts or users to the file. If no hosts or users are listed then everyone is given access.



If you put any entries in the `hosts.lpd` file, only those entries will be given access and all others will be denied.

Using Print Filters

Objective**1.7 Text editing, Processing, Printing**

- **Install and configure local and remote printers.** Install a printer daemon, install and configure a print filter (e.g.: `apsfilter`, `magicfilter`). Make local and remote printers accessible for a Linux system, including postscript, non-postscript, and Samba printers. Involves the daemon: `lpd`. Involves editing or reviewing the files and directories: `/etc/printcap`, `/etc/apsfilterrc`, `/usr/lib/apsfilter/filter/*`, `/etc/magicfilter/*`, `/var/spool/lpd/*`

Print filters convert the data to be printed to the format that the print device can understand. If you only print text documents, you do not need a print filter, but if you print graphics, charts, or specially formatted text from most word processors, you do.

As an administrator, you have the option of either creating your own print filter or downloading one from the Internet that has already been created. Unless you are very familiar with writing shell scripts and working with all different types of files, it is best to just use a filtering package from someone else. The three most popular print filter packages are as follows:

- ♦ `Apsfilter`
- ♦ `Magicfilter`
- ♦ Red Hat's `PrintTool`

The choice of which to use depends on your preference and the printers that each package supports. The print filter should be installed on the print server. Network attached printers do not run print filters, so the job should be run through the local filter before being sent to the print device.



Some versions of LPR, especially the one that ships with Debian, do not send the print job through a filter before sending it to a remote print host. To fix this problem upgrade to LPRng.

Installing Apsfilter

The `Apsfilter` package is available from <http://www.apsfilter.org>. It provides a very easy to use menu-based interface to set up and configure many different printers. The `Apsfilter` package is not used to manage printers, so it is not used to change or remove already installed printers. To install and configure the filter follow these steps.

1. Download the Apsfilter package from the <http://www.apsfilter.org> site. Usually getting the latest version is the best idea since it should have fewer bugs and supports more printers.
2. Uncompress and untar the package. For example:

```
[root@redhat /root]# tar zxvf apsfilter-6.0.0.tar.gz
```
3. Execute the `./SETUP` install script. Pay close attention to the first several screens as they provide information on exactly what will happen.
4. The install script will check several things on the system and will prompt you for confirmation. Make sure all of the detected settings are correct.
5. Choose your printer driver. This is done by selecting 1 from the menu. The next menu asks you where the driver would be located. If you do not know, just guess and look through the different options. Once you have found your printer, or one compatible with it, enter the number given.
6. The next option is to configure the printer interface. This is done by selecting 2 from the main menu. As you can see, Apsfilter supports many different connections including parallel, serial, USB, `lpd`, AppleTalk, and Samba (Windows).

The next questions asked depend on the interface you have selected. If you choose `lpd`, you will enter the remote system's address and printer name. A serial or parallel connection requires the local device name, and a Samba share requires Windows information. All of the options are well documented with examples.
7. The next step is to test the printer by printing a test page. This is done by selecting T from the menu. Before doing this you may need to set the paper type and whether you want color or black and white. Make sure the test page prints correctly before continuing. Change the driver if needed.
8. Next, you write the `/etc/printcap` file with the information you have configured. This is done by selecting I from the menu. You will be prompted for some information during this step including the printer and print queue names.
9. The next step is to exit Apsfilter by pressing Q. You will be prompted about where to install the man pages and how to compress them. `/usr/local/man` and `gzip` are good answers.
10. The final step includes reading the informational screens telling you about the setup and sending the author the requested postcard if you like the software.

Installing Magicfilter

Magicfilter is available at <ftp://metalab.unc.edu/pub/Linux/system/printing/>. It provides a set of printer filters that you compile and install on your system. It does not provide the configuration support that Apsfilter does. You must manually configure the `/etc/printcap` file and then just insert a Magicfilter filter.

1. Obtain the Magicfilter package from `ftp://metalab.unc.edu/pub/Linux/system/printing/`.
2. Uncompress and untar the package file, for example:


```
[root@aurora /root]# tar zxvf magicfilter-1.2.tar.gz
```
3. Execute the `./configure` script. You will see it perform many tests and report the results.
4. The next step is to compile the Magicfilter software by typing `make`. A lot of output will be displayed while the software is compiled.
5. Install the Magicfilter software by typing `make install`. It will display where the files have been placed.
6. You have the option of either installing all print filters with `make install_filters` or copying the specific filter you need from the `filters` directory to `/usr/local/bin`. If in doubt, install them all.
7. The final step is to add a print filter line to your existing `/etc/printcap` file. For example:


```
:if=/usr/local/bin/ljet4-filter:
```

As you can see, Magicfilter does not provide the ease of use or testing capability of Apsfilter.

Using PrintTool

Red Hat has created an easy to use printer management tool and filter package. It is installed by default with Red Hat and most variants, and packages are available for other distributions including Debian. The PrintTool interface is a GUI that uses the X Window system. It lets you add, remove, and manage printers. Figure 12-1 shows the main menu.

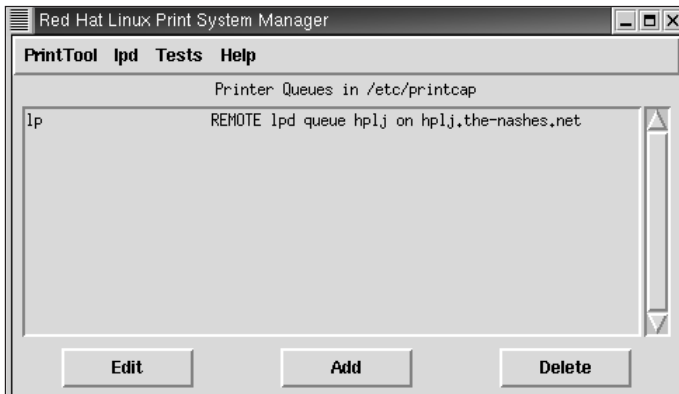


Figure 12-1: PrintTool

Managing the Printer Services

Objective

1.7 Text editing, Processing, Printing

- **Manage printers and print queues.** Monitor and manage print queues and user print jobs, troubleshoot general printing problems. Includes the commands: `lpc`, `lpq`, `lprm` and `lpr`. Includes reviewing the file: `/etc/printcap`.

Several tools are provided in the LPR package to manage the printer services. They can be used to view and manage the print queues, jobs, and devices.

Managing the printer daemon

The daemon that handles print spooling in Linux is `lpd`. The `lpd` daemon receives print jobs from a user, it then takes them and runs them through a filter, if configured. It then takes the job and puts it in the printer's spool directory and adds it to the queue to be printed.

The `lpd` daemon is usually started, stopped, and managed by the runlevel script that runs at boot. The script is located in `/etc/rc.d/init.d/lpd` on Red Hat and variants, and `/etc/init.d/lpd` on Debian, unless you are running LPRng, which uses the `/etc/init.d/lprng` script. This script takes the usual commands, such as `start`, `stop`, `status`, and `restart`.

While `lpd` is running you may notice multiple instances of `lpd`. The parent `lpd` process may spawn child processes to handle requests, and in this way, it can keep listening for new print jobs.

Managing printers

Managing the print devices is an important administrative task. Printers occasionally must be taken off line for upgrades or to repair jams. The management of printers is handled with the `lpc` command. A complete list of commands for `lpc` is shown in Table 12-2. Commands can either be given at the command line or from the interactive shell presented by typing `lpc` with no commands. The syntax for `lpc` is as follows:

```
lpc [command [argument ...]]
```

The syntax for `lpc` in LPRng is as follows:

```
lpc [ -A ] [ -a ] [ -Ddebugopts ] [ -Pprinter ] [ -Sserver ]
[ -Uusername ] [ -V ] [ command [ argument ... ] ]
```

Table 12-2
lpc Commands

Command	Function
<code>help or ? [command]</code>	Shows help about the specified command. If no argument is given, shows a list of recognized commands.
<code>active [pr@[host]]</code>	Makes a connection to another LPD server to see if it is alive. Works only with LPRng.
<code>abort {all printer name}</code>	Immediately terminates the active spool daemon for the specified printer and then disables printing for that printer.
<code>class {all printer name}</code> <code>(off class list glob match)</code>	Controls the class of the jobs being printed. Classes are given in a range between A (lowest priority) to Z (highest priority). Works only with LPRng.
<code>defaultq</code>	Lists the default queue for the <code>lpc</code> command.
<code>defaults</code>	Lists the default settings. Works only with LPRng.
<code>debug j all printer }</code> <code>[string off]</code>	Specifies the debug string for the specified printer. Works only with LPRng.
<code>disable {all printer name}</code>	Stops spooling for this printer.
<code>down</code> <code>{all printer name} <message></code>	Disables the specified queue and places the message in the status file that will be reported by <code>lpq</code> .
<code>enable {all printer name}</code>	Enables the spool for the specified printers.
<code>exit or quit</code>	Exits out of <code>lpc</code> .
<code>hold printer { job id }</code>	Stops the job specified from being printed, until it has been released. Works only with LPRng.
<code>holdall { all printer* }</code>	Holds all new jobs so they are not printed until released. Works only with LPRng.
<code>kill { all printer ... }</code>	Performs an <code>abort</code> command followed by a <code>start</code> command. This is useful to reset a server that is causing problems. Works only with LPRng.
<code>client { all printer ... }</code>	Shows the client configuration and printcap information on the local host. Works only with LPRng.
<code>lpd [printer@[host]]</code>	Checks to see if the specified LPD daemon is running and retrieves its process ID. Works only with LPRng.

Command	Function
<code>lpq printer [options]</code>	Executes the <code>lpq</code> command from inside of <code>lpc</code> . Works only with LPRng.
<code>lprm printer jobid [jobid]*</code>	Executes <code>lprm</code> from inside of <code>lpc</code> . Works only with LPRng.
<code>move printer { jobid } destinationPrinter</code>	Moves the specified jobs to the destination printer, and removes them from the existing queue. Works only with LPRng.
<code>msg printer message text</code>	Updates the status message for the printer. Works only with LPRng.
<code>noholdall { all printer* }</code>	Disables automatic job holding started by <code>holdall</code> . Works only with LPRng.
<code>redirect [printer [destinationPrinter off]]</code>	Redirects jobs from the specified printer to the new destination printer. Use the <code>off</code> parameter to turn off redirection. Works only with LPRng.
<code>redo [printer [jobid]]</code>	Reprints the specified job. Works only with LPRng.
<code>release [printer [jobid]]</code>	Releases the selected held job for printing. Works only with LPRng.
<code>reread [printer [@host]]</code>	Instructs <code>lpd</code> to reread the <code>printcap</code> file. Works only with LPRng.
<code>restart { all printer }</code>	Starts a new printer daemon.
<code>server { all printer }</code>	Shows the <code>printcap</code> entries for the specified printer(s). Works only with LPRng.
<code>start {all printer name}</code>	Enables the spool and starts the print daemon for the specified printer(s).
<code>status {printer name}</code>	Displays the status of the printer and queue.
<code>stop {all printer name}</code>	Stops the spooling daemon after the current job is completed and then disables printing for this printer.
<code>topq <printer> [job number...] [user...]</code>	Places the jobs in the order you list at the top of the printer queue.
<code>up {all printer name}</code>	Enables printing and spooling and starts a printing daemon.

Command-line options for the LPRng `lpc` are shown in Table 12-3.

Table 12-3
LPRng lpc Command-Line Options

Option	Function
-A	Uses the authentication method specified in the AUTH environment variable.
-a	Means the same thing as -Pa11.
-P<printer name>	Specifies the printer spool to work with.
-S<server>	Specifies the server to send commands to.
-V	Displays version information.
-U<username>	Allows root to specify the user name to use for the request.
-D<debug options>	Specifies options to use for debugging. See the man page for more information.

Listing all printers

To show information about all of the printers on the system you can use the command `lpc status`. For example:

```
debian:~# lpc status
lp:
    queuing is enabled
    printing is enabled
    no entries
    no daemon present
```

An example of this command with LPRng is as follows:

```
debian:~# lpc status all
Printer      Printing Spooling Jobs  Server Subserver
Redirect Status/(Debug)
lp@debian    enabled  enabled    0    none    none
```

Stopping a printer

To stop the use of a printer you can use the `lpc stop <printer name>` command. The printer name is that given in the `/etc/printcap` file. For example:

```
[root@redhat /root]# lpc stop lp
Printer: lp@redhat
lp@redhat.the-nashes.net: stopped
```

Starting a printer

To enable a printer for use by users you can use the `lpc start <printer name>` command. For example:

```
[root@ajax /root]# lpc start lp
Printer: lp@ajax
lp@ajax.planetportal.com: started
```

Managing print queues

Once a print job is ready to print, it is placed in the printer's queue. The `lpq` command is used to check the status of the print queues. To control the print queue and move jobs, you use the `lpc` command. The available command-line arguments for `lpq` are shown in Table 12-4. The syntax for `lpq` is as follows:

```
lpq [-l] [-Pprinter] [job # ...] [user ...]
```

The syntax for `lpq` in LPRng is as follows:

```
lpq [ -a ] [ -A ] [ -l ] [ -L ] [ -P printer ] [ -s ]
[ -t sleeptime ] [ -V ] [ -D debugopt ] [ jobid... ]
```



The `lpq` command is used to view the print queue while `lpc` moves jobs to the top of the queue.

Table 12-4
lpq Command-Line Options

Option	Function
-A	Uses authentication that is specified by the value of the AUTH environment variable. Works only with LPRng.
-D<debug options>	Passes debugging information to <code>lpq</code> . See the man page for more information. Works only with LPRng.
-P <printer>	Specifies the destination printer name.
-V	Displays version information. Works only with LPRng.
-a	Lists information about all of the printers. Works only with LPRng.
-l	Enables verbose mode. Multiple <code>-l</code> options may be used to increase the amount of information generated.
-L	Enables maximum verbosity. Works only with LPRng.

Continued

Table 12-4 (continued)

Option	Function
-s	Displays a single line summary for each queue and subqueue. Works only with LPRng.
-t<sleep time>	Keeps displaying the spool queues but waits the specified seconds between updates. Works only with LPRng.
<job id> ... all	Displays information about specified job IDs or all. A job ID can be a user name, job identifier, job number, or a pattern. Works only with LPRng.

Listing the contents of print queues

To see the jobs in one or more print queues you can use the `lpq` command. To display the queue of the default printer, you would use no arguments. For example:

```
Printer: lp@redhat (dest lp@4.18.168.26)
Queue: no printable jobs in queue
Status: job 'jason@redhat+967' removed at 11:59:50.603
```

To view a nondefault printer named `hpj1`, you would use the command `lpq -Phpj1`. The printer name is the name defined in the `/etc/printcap` file.

Moving jobs in the print queue

After using `lpq` to see the jobs in the queue, you can use `lpc` to move print jobs around in the queue. For example, to move job numbers 30, 35, and 40 to the top of the queue for the printer named `lp` you would enter:

```
[root@redhat /root]# lpc topq lp 30 35 40
```

Checking the status of a print queue

To check the status of a print queue you use the `lpc` command with the `status` option. For example:

```
debian:~# lpc status lp
lp:
    queuing is enabled
    printing is enabled
    no entries
    no daemon present
```

An example of the command using LPRng is as follows:

```
[root@redhat /root]# lpc status lp
Printer          Printing Spooling Jobs  Server Subserver Redirect
Status/(Debug)
lp@redhat        enabled enabled      0    none    none
```

This is the same command used to check the printer.

Disabling a print queue

To disable a print queue you use the `lpc` tool with the `disable` command. For example:

```
debian:~/# lpc
lpc> disable lp
lp:
      queuing disabled
lpc>
```

An example of the command using LPRng is as follows:

```
[root@redhat /root]# lpc
lpc>disable lp
Printer: lp@redhat
lp@redhat.the-nashes.net: disabled
lpc>
```

Enabling a print queue

To enable a print queue you use the `lpc` tool with the `enable` command. For example:

```
[root@redhat /root]# lpc
lpc>enable lp
Printer: lp@redhat
lp@redhat.the-nashes.net: enabled
lpc>
```

Managing print jobs

Managing the print queue involves making sure queues go to certain printers and moving jobs up to the top of the list. Managing print jobs deals with removing individual jobs from the queue. This is done with the `lprm` command. The syntax for `lprm` is as follows:

```
lprm [-Pprinter] [-] [job # ...] [user ...]
```

The syntax for `lprm` from LPRng is as follows:

```
lprm [ -a ] [ -A ] [ -Ddebugopt ] [ -Pprinter ] [ -V ]
[-Uuser ] [ jobid... ] [ all]
```

An example of its use is as follows:

```
debian:/etc# lpq
Warning: no daemon present
Rank  Owner      Job  Files                Total Size
1st   root        10   /etc/hosts           27 bytes
debian:/etc# lprm 10
dfA010debian dequeued
cfA010debian dequeued
debian:/etc#
```

An example of the command using LPRng is as follows:

```
[root@redhat /root]# lpq
Printer: lp@redhat (dest lp@4.18.168.26) (holdall)
Queue: no printable jobs in queue
Holding: 1 held jobs in queue
Server: no server active
Status: job 'jason@redhat+967' removed at 11:59:50.603
Rank  Owner/ID      Class Job Files                Size Time
hold  root@redhat+122  A    122 lp,/etc/hosts           78 23:55:21
[root@redhat /root]# lprm 122
Printer lp@redhat:
  checking perms 'root@redhat+122'
  dequeued 'root@redhat+122'
```

First you need to find the job ID, which can be displayed with `lpq`. Next you use the job ID to remove the job from the queue. If only a user name is given, all jobs for that user will be deleted. For example:

```
debian:/etc# lprm root
dfA011debian dequeued
cfA011debian dequeued
```



The `lprm` command is used to remove print jobs from the queue.

Table 12-5 lists the command-line options for `lprm`.

Table 12-5
lprm Command-Line Options

<i>Option</i>	<i>Function</i>
-	Removes all jobs the user owns.
-A	Uses the authentication specified in the <code>AUTH</code> environment variable. Works only with LPRng.
-a	Removes all print jobs in all queues available to the user. Works only with LPRng.
-P<printer>	Specifies the printer queue to control.
-U<user>	Allows root to change the user that requests originate from. Works only with LPRng.
-D<debug options>	Specifies options to enable debugging mode. See the <code>lprm</code> man page for more information. Works only with LPRng.
-V	Displays version information and enable verbose mode. Works only with LPRng.

Printing Files

Objective

1.7 Text editing, Processing, Printing

- **Manage printers and print queues.** Monitor and manage print queues and user print jobs, troubleshoot general printing problems. Includes the commands: `lpc`, `lpq`, `lprm` and `lpr`. Includes reviewing the file: `/etc/printcap`.
- **Print files.** Submit jobs to print queues, convert text files to postscript for printing. Includes `lpr` command.

Most applications provide printing support and may only require the destination printer name. But what if you want to just quickly print a text file or other documentation? The `lpr` command can be used to manually print files, and the `a2ps` tool can be used to convert files to PostScript.

Using lpr

The LPR package provides the `lpr` command to manually print text files from scripts or the command-line. Table 12-6 lists the possible command-line arguments for `lpr`. The syntax for `lpr` is as follows:

```
lpr [-Pprinter] [-#num] [-C class] [-J job] [-T title] [-U user] [-i
    [numcols]] [-1234 font] [-wnum] [-cdfghlnmprstv] [printer name ...]
```

Table 12-6
lpr Command-Line Options

Option	Function
-c	The files are assumed to come from the <code>cifplot</code> application.
-d	The files are assumed to come from the <code>tex</code> application.
-f	Uses a filter that interprets the first character of each line as a standard FORTRAN carriage control character.
-g	The files are assumed to contain standard plot data as produced by the plot routines.
-l	Uses a filter that allows control characters to be printed and suppresses page breaks.
-n	The files are assumed to contain data from <code>ditroff</code> .
-p	Uses <code>pr</code> to format the files.
-t	The files are assumed to contain data from <code>troff</code> .
-v	The files are assumed to contain a raster image.
-P<printer>	Forces output to a specific printer instead of the default.
-h	Does not print the burst page.
-m	Sends mail when the job is completed.
-r	Deletes the file when the print job completes.
-s	Instead of copying the file to be printed to the spool directory, uses symbolic links.
-#<num>	Specifies the number of copies to print.
-[1234]	Specifies that the font name be mounted to the number position.
-C <class>	Specifies the job classification to use on the burst page.
-J <job>	Specifies the job name to print on the burst page.
-T <title>	Specifies the title name to use instead of the filename.
-U <user>	Specifies the user name to put on the burst page.
-i [numcols]	Specifies the number of blanks to indent the printing.
-w<num>	Uses the specified number for the page width.

The syntax for `lpr` from LPRng is as follows, with the options explained in Table 12-7:

```
lpr [ -A ] [ -b,l ] [ -C class ] [ -D debugopt ]
    [ -F filterformat ] [ -h ] [ -i indentcols ]
    [ -k ] [ -J job ] [ -K,# copies ]
    [ -m mailto ] [ -P printer ] [ -Q ] [ -r ]
    [ -R remoteAccount ] [ -s ] [ -T title ] [ -U user ]
    [ -V ] [ -w width ] [ -Z zoptions ] [ -1,2,3,4 font ]
    [ filename... ]
```

Table 12-7
LPRng lpr Command-Line Options

Option	Function
-A	Uses the authentication type specified in the AUTH environment variable.
-C <class>	Specifies the job priority, which ranges from A (slowest) to Z (fastest). The default is A.
-D <debug options>	Specifies options to enable debugging mode. See the <code>lpr</code> man page for more information.
-F <filter format>	Specifies a filter or format specification instead of the default filter.
-h	Indicates no banner page for this job.
-i <indent columns>	Indents the job the specified number of columns.
-J <job name>	Specifies the job name.
-k	Sends the print job directly to the server, instead of waiting for the full job to go through the filter.
-K<copies>, -#<copies>	Prints the specified number of copies.
-m <mail to>	Sends mail to the specified address if the job does not complete successfully.
-P <printer>	Specifies the destination printer instead of using the default.
-Q	Puts the name of the spool queue in to the job file. This is sometimes used with other spooling software.
-R <remote account>	Specifies accounting information for the remote system.
-s	Creates a symbolic link to the file to be printed instead of sending the file directly. This is supported for backward compatibility.
-T <title>	Specifies the title name.
-U <user name>	Allows root to change the username associated with a job.

Continued

Table 12-7 (continued)

Option	Function
-V	Enables verbose mode. Multiple -V parameters can be added to increase output.
-w <width>	Specifies the page width for the job.
-Z <options>	Allows you to pass special options directly to the print spooler.

For example, to print the `/etc/hosts` file to the printer named `hplj` with `lpr` you would type the following:

```
[root@redhat /root]# lpr -Phplj /etc/hosts
```

Using a2ps

Many sites that use UNIX systems have PostScript printers, since this is considered to be the standard printing language to use. But, if you send a text file directly to a PostScript printer, it will not print correctly. To change the format you can use the `a2ps` tool, which ships with many distributions. To convert the file `myfile.txt` to PostScript, you would type the following:

```
a2ps -o myfile.ps myfile.txt
```

The `-o` option specifies the output file name. To print a text file to the default printer as PostScript you would type the following:

```
a2ps myfile.txt
```

Troubleshooting Printing Problems

Troubleshooting printing problems can be confusing because of the different pieces involved. A messed-up queue, crashed `lpd`, or jammed printer all result in the same problem, no printing. The tools are available to help narrow down and find the problem.

lpd problems

If no printers are working, the first thing to check is the `lpd` daemon. A quick process list check will show if it is still loaded. For example:

```
[root@redhat /root]# ps auxw | grep lpd
lp          14743  0.0  0.0  2432  192 ?        S    Jan03   0:00
lpd Waiting
```

Another option is to check the startup script. For example, if you checked the script and the `lpd` daemon had crashed you would see this:

```
[root@redhat /root]# /etc/rc.d/init.d/lpd status
lpd dead but subsys locked
```

A simple restart command should fix the problem. For example:

```
[root@redhat /root]# /etc/rc.d/init.d/lpd restart
Stopping lpd:                [FAILED]
Starting lpd:                [ OK ]
```

If the `lpd` daemon has crashed, be sure to check the `syslog` for possible information.

Queue problems

If print jobs seem to be stalled in the print queue, make sure the queue has not been disabled. This can be checked with the `lpc` command. For example:

```
debian:~# lpc status
lp:
    queuing is enabled
    printing is enabled
    no entries
    no daemon present
```

An example of the command using `LPRng` is as follows:

```
[root@redhat /root]# lpc status all
Printer          Printing Spooling Jobs  Server Subserver Redirect
Status/(Debug)
lp@redhat        enabled disabled      0   none   none      (holdal
```

Notice that the spooling service has been disabled. To enable it use the following command:

```
debian:~# lpc enable all
lp:
    queuing enabled
```

An example of the command using `LPRng` is as follows:

```
[root@redhat /root]# lpc enable all
Printer: lp@redhat
lp@redhat.the-nashes.net: enabled
```

```
[root@redhat /root]# lpc status
Printer          Printing Spooling Jobs  Server Subserver Redirect
Status/(Debug)
lp@redhat       enabled enabled      0   none   none       (holdal
```

Printer problems

The printer itself may also be a problem. First, make sure printing has not been disabled on this printer. This can be checked with `lpc`. For example:

```
debian:~# lpc status
lp:
    queuing is enabled
    printing is enabled
    no entries
    no daemon present
```

An example of this command with LPRng would be as follows:

```
[root@redhat /root]# lpc status
Printer          Printing Spooling Jobs  Server Subserver Redirect
Status/(Debug)
lp@redhat       disabled disabled    0   none   none       (holdal
```

Notice that printing has been disabled on this device. Printing can be turned back on using the `up` command, for example:

```
debian:~# lpc up all
lp:
    printing enabled
    daemon started
```

An example of the command using LPRng is as follows:

```
[root@redhat /root]# lpc up all
Printer: lp@redhat
lp@redhat.the-nashes.net: enabled and started
```

If printing is enabled and still no jobs are getting processed, you may need to check the actual printer. Make sure it is turned on and has paper in it. A quick way to check the printing hardware is to `cat` a file to the printer interface. For example, to check a printer attached to the system's parallel port you would use the following:

```
[root@redhat /root]# cat /etc/host > /dev/lpt0
```

This dumps the text file directly to the hardware interface, bypassing the LPD system.

File and directory problems

Usually the `printcap` file is not the problem, unless this is the first time the printer has been used. Make sure that the first line in a printer block does not start with a colon or pipe. Make sure all lines except the last one in the block end in `:\`. If multiple printers are configured, make sure they have separate spool directories.

Check the printer definitions so the interface device or remote system is correct, and a very small maximum print job size is not set. If you configured the file manually, you may try one of the configuration tools such as `Apsfilter` or `PrintTool`.

The next step is to look at the spool directory, ownership, and permissions. If a configuration tool made these for you, they are most likely correct, but should be checked anyway. Make sure the proper ownership was granted and the permissions are correct, or the `lpd` daemon will not be able to create files there.

Space problems

If your system runs out of disk space new spool files cannot be created. Check the disk space on the volume containing `/var` and make sure adequate space is available.

Key Point Summary

The Linux printing system can be complex to the uninitiated, but the key lies in knowing where to go for information. The different tools sometimes seem to overlap, but provide the ability to troubleshoot and debug almost any problem.

- ♦ Most Linux distributions use the LPR printing system.
- ♦ Local and remote printer configuration is stored in the `/etc/printcap` file.
- ♦ Printer blocks in the `/etc/printcap` begin with a line specifying the printer name and optional aliases.
- ♦ Printer configuration definitions begin with a colon.
- ♦ All lines in the `/etc/printcap` file end in `:\`, except for the last one in a printer block.
- ♦ The print spool directory and log file must be created manually.
- ♦ Print access is controlled with the `/etc/hosts.lpd` file.
- ♦ Print filters format different data types to a language understood by the printer.

- ♦ The `lpd` daemon is managed either by a startup script or manually by calling the daemon.
- ♦ The `lpd` daemon is responsible for getting print jobs from the user, putting them through the print filter, and delivering them to the spool directory.
- ♦ The `lpc` command is used to control the printer and many of the spool processes.
- ♦ The `lpq` command is used to view print queue information and move jobs around the queue.
- ♦ The `lprm` command is used to remove print jobs from the queue.
- ♦ The print client tool that is used to manually print files is `lpr`.



STUDY GUIDE

The following questions and exercises will allow you to review the information covered in this chapter. Take your time when completing this section of the chapter, carefully reviewing any questions that you may have. Simply answering the question correctly is not as important as understanding the answer, so review any material that you might still be unsure of. Being comfortable with the questions and answers presented here will help you be more prepared for the certification exam questions.

Assessment Questions

1. Which command would be used to view the queue for all printers?
 - A. `lpd status all`
 - B. `lpq status all`
 - C. `lpc status all`
 - D. `lprm status all`
2. Which of the following spools print jobs to the print device?
 - A. `lpq`
 - B. `lpc`
 - C. `lpd`
 - D. `lpr`
3. Which command would be used to print the file `mydoc.txt` to the printer named `hplj`?
 - A. `lpr mydoc.txt`
 - B. `print -Phplj mydoc.txt`
 - C. `cat mydoc.txt | lpr hplj`
 - D. `lpr -Phplj mydoc.txt`
4. To place all new print jobs on hold you would use the _____ command.

5. Which command should be used to restart the lpd daemon?
 - A. /etc/rc.d/init.d/lpd restart
 - B. /usr/sbin/lpd restart
 - C. /etc/init.d/lpd stop | start
 - D. kill -1 lpd

6. Which of the following is used to define a filter in the /etc/printcap file?
 - A. of
 - B. filter
 - C. fi
 - D. if

7. Which sequence should be put at the end of a line in the /etc/printcap file, assuming this is not the last line in the printer block?
 - A. \:
 - B. :\
C. ;
D. ;\

8. Which command is used to move a print job to the top of the queue?
 - A. lpc
 - B. lprm
 - C. lpq
 - D. lpr

9. Which definition in /etc/printcap disables print banners?
 - A. :sh:
 - B. :nb:
 - C. :nh:
 - D. :hs:

10. Which definition in /etc/printcap sets the maximum print job size?
 - A. :ms:
 - B. :js:
 - C. :xm:
 - D. :mx#:

11. To remove job 243 from the print queue, you would type _____.
12. What needs to be done after editing the `/etc/printcap` file?
- A. `kill -1 <lpd PID>`
 - B. `/etc/rc.d/init.d/lpd restart`
 - C. `/etc/init.d/lpd reload`
 - D. nothing
13. Which of the following is not a valid `lpc` command?
- A. `refresh`
 - B. `up`
 - C. `down`
 - D. `enable`
14. Which `lpc` command is used to print a job that is currently on hold?
- A. `release`
 - B. `enable`
 - C. `restart`
 - D. `nohold`
15. Which must be created when manually adding a printer?
- A. lock file
 - B. spool directory
 - C. queue file
 - D. job file
16. Which command will stop print jobs in the queue from printing?
- A. `lpq stop all`
 - B. `lpq hold all`
 - C. `lpc stop all`
 - D. `lpc hold all`

17. When doing a `lpc status all`, you see the following output. What do you enter to restart printing?

```
Printer          Printing Spooling Jobs  Server Subserver
Redirect Status/(Debug)
lp@debian        disabled  enabled    0    none    none
```

- A. `lpq start all`
 - B. `lpc start all`
 - C. `lpd reload`
 - D. `lpc enable all`
18. Which command would be used to check the printer connection, without going through the printing system?
- A. `lpr textfile.txt`
 - B. `echo textgfile.txt > /dev/lpt0`
 - C. `print textfile.txt`
 - D. `cat textfile.txt > /dev/lpt0`
19. Which daemon spools print jobs to remote print servers?
- A. `lpq`
 - B. `lpd`
 - C. `lpr`
 - D. `lps`
20. Which print filter provides a text-mode menu interface?
- A. Magicfilter
 - B. PrintTool
 - C. Apsfilter
 - D. LPR Filter

Scenarios

1. Why would you consider writing your own print filter instead of using Apsfilter or Magicfilter?
2. Which steps would you work through to find a problem if your user's print jobs stopped being printed?

Lab Exercises

Lab 12-1 Setting up a printer

Even if you do not have a printer connected to your Linux system, you can configure a printer and make sure the jobs are queued up correctly. Follow these steps to set up a fake printer.

1. Log into the system as root.
2. Open the `/etc/printcap` file in an editor, such as `vi`.
3. Move to the bottom of the file and enter insert mode in the editor.
4. Begin the printer block by giving the printer a name and an alias, for example:

```
lp2|testprinter:\
```

5. Since this is not an actual printer, you will not add a line for the hardware interface. This will cause the print jobs to just sit in the print queue.

6. Add an entry for the print spool directory, for example:

```
:sd=/var/spool/lpd/lp2:\
```

7. Disable the use of print banners by adding the following line:

```
:sh:\
```

8. The last line will set the maximum print job size. This is designated in blocks, which are 1KB. Remember that the last line does not end in `\`. To limit the print job size to 2MB you would use the following:

```
:mx#2048:
```

9. Manually create the spool directory, and give it the correct permissions.

```
drwx----- 2 daemon lp 35 Jan 21 17:19 lp2
```

10. Restart the `lpd` service by using the startup script.

11. Check the status of the new printer, for example:

```
debian:~# lpc status lp2
Printer          Printing Spooling Jobs  Server Subserver
Redirect Status/(Debug)
lp2@debian      enabled  enabled    0    none    none
```

12. Submit a job to the print queue.

```
debian:~# lpr -P lp2 /etc/hosts
```

13. Check to make sure the job was put into the queue as expected. This shows the job was removed from the queue, since there was no device to spool it to.

```
europa:~# lpq -Plp2
Printer: lp2@debian 'testprinter'
Queue: no printable jobs in queue
Status: job 'cfA645debian.the-nashes.net' removed at
17:28:40.505
```

Answers to Chapter Questions

Chapter Pre-Test

1. The printing system is controlled with the `/etc/printcap` file.
2. The `lpr` command is used to manually print files.
3. Print filters convert data to a format understood by the printer.
4. The `lprm` command removes print jobs from the queue.
5. The `lpd` daemon spools print jobs to the print device, or to a remote print server.
6. The `lpd` daemon receives print jobs from users.
7. The `lpc hold all` command prevents new jobs from being spooled.
8. The `:if` line specifies the print filter to use.
9. The print filter should be installed on the print server.
10. The `a2ps` tool converts text files to PostScript.

Assessment Questions

1. **B.** The `lpq` command is used to view the contents of the print queue. For more information, see the “Managing print queues” section.
2. **C.** The `lpd` daemon receives jobs from the user and spools them to the print device. For more information, see the “Managing the printer daemon” section.
3. **D.** The `lpr` command is used to manually print files, and the `-P` option specifies the destination printer. For more information, see the “Printing Files” section.
4. `lpc hold all`. The `hold all` option for `lpc` prevents new jobs from being spooled. For more information, see the “Managing printers” section.
5. **A.** You should use the startup script, if available. For more information, see the “Managing the printer daemon” section.
6. **D.** Remember that `if` means input filter. For more information, see the “Installing Printers” section.
7. **B.** The last line in a printer block ends in a semicolon. For more information, see the “Installing Printers” section.
8. **C.** Use `lpq` to move a print job to the top of the queue. For more information, see the “Managing print queues” section.
9. **A.** The `:sh:` definition disables banners. For more information, see the “Installing Printers” section.

10. **D.** This sets the maximum job size in 1KB blocks. For more information, see the “Installing Printers” section.
11. **lprm 243.** The `lprm` command removes print jobs. For more information, see the “Managing print jobs” section.
12. **B.** The `lpd` daemon should be restarted after any configuration changes. For more information, see the “Installing Printers” section.
13. **A.** `up`, `down`, and `disable` are valid commands, but `refresh` is not. For more information, see the “Managing printers” section.
14. **A.** The `release` command releases a job that’s on hold. For more information, see the “Managing printers” section.
15. **B.** The spool directory must be manually created, and the permissions and ownership must be set. For more information, see the “Creating the spool directory and log file” section.
16. **C.** The `stop all` directive must be used with `lpc`. For more information, see the “Managing printers” section.
17. **B.** This is the opposite of doing `lpc stop` and will enable a stopped printing device. For more information, see the “Managing printers” section.
18. **D.** This sends a file directly to the interface port. For more information, see the “Printer problems” section.
19. **B.** The `lpd` daemon is used to send remote print jobs. For more information, see the “Managing the printer daemon” section.
20. **A.** PrintTool provides an X GUI interface, and Magicfilter has no interface. For more information, see the “Using Print Filters” section.

Scenarios

1. You should only consider writing your own print filters if you are very familiar with different file formats and the language your printer supports. You only need to write a new filter if the existing filters do not support your printer, or if you need to customize the way data is handled.
2. First, check to make sure the `lpd` daemon is still running by performing a `ps` check. Next, check to make sure spooling and printing is enabled with `lpc status all`. Finally, check the queue status to make sure the jobs are not on hold. Finally, restart the `lpd` daemon for a fresh start. If printing still does not start, check the physical interface manually by sending a file directly to the device.

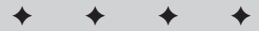
Working with the Kernel

EXAM OBJECTIVES

Exam 102 ♦ General Linux, Part 2

1.5 Kernel

- **Manage kernel modules at runtime.** Learn which functionality is available through loadable kernel modules, and manually load and unload the modules as appropriate. Involves using the commands: `lsmod`, `insmod`, `rmmmod`, `modinfo`, `modprobe`. Involves reviewing the files: `/etc/modules.conf` | `/etc/conf.modules` (* depends on distribution *), `/lib/modules/{kernel-version}/modules.dep`.
- **Reconfigure, build and install a custom kernel and modules.** Obtain and install approved kernel sources and headers (from a repository at your site, CD, kernel.org, or your vendor); Customize the kernel configuration (i.e., reconfigure the kernel from the existing `.config` file when needed, using `oldconfig`, `menuconfig` or `xconfig`); Make a new Linux kernel and modules; Install the new kernel and modules at the proper place; Reconfigure and run `lilo`. N.B.: This does not require to upgrade the kernel to a new version (full source nor patch). Requires the commands: `make` (`dep`, `clean`, `menuconfig`, `bzImage`, `modules`, `modules_install`), `depmod`, `lilo`. Requires reviewing or editing the files: `/usr/src/linux/.config`, `/usr/src/linux/Makefile`, `/lib/modules/{kernelversion}/modules.dep`, `/etc/conf.modules` | `/etc/modules.conf`, `/etc/lilo.conf`.



CHAPTER PRE-TEST

1. What is a modular kernel?
2. When should you compile a new kernel?
3. How many patches are required to update from kernel 2.2.13 to 2.2.18?
4. Which command is used to install compiled modules?
5. Which file should be edited after a new kernel is installed?
6. Which package is required for the graphical kernel configuration?
7. Which tool is used to remove a running module?
8. How can you find the author of a module?
9. What is the name of the file where the kernel configuration is stored?
10. Which command is used to build a bzip'd kernel image?

The kernel is the core of the Linux system. Most Linux distributions ship with a kernel that supports as much hardware as possible and that comes with as many features as possible. The problem with this is you sometimes get a kernel that is not as lean and mean as possible. Some features may cause problems or slow the kernel down. Even if you never upgrade a system's hardware, you will most likely have to upgrade the Linux kernel eventually. New kernels provide performance increases, security fixes, and new hardware support.

This chapter goes through obtaining, configuring, compiling, and installing a new kernel, as well as configuring drivers and modules to support new hardware.

Kernel Overview

The Linux kernel can seem daunting to new users, but does not need to be. The Linux kernel is responsible for mediating between the system's hardware and Linux programs. It handles the memory management for the operating system to make sure each process gets its own memory range. The kernel also divides up the processor's time to make sure each application gets a fair share of the processor's use. Hardware drivers are also part of the kernel and provide an interface for the operating system and its applications to talk to hardware.

Kernel development

When it is time to update the kernel on a system, you may be surprised at the number of choices you have. The Linux kernel development system can sometimes be confusing, but understanding it is important so that you know which version to use on a system.

Linus Torvalds, the original Linux kernel author, is still head of the kernel development effort. Many other people around the world also work on the kernel and submit new patches and features. Who does the kernel development is not as important as how they release their work.

At any given time, two current kernel versions will usually be available. Why two? Because a stable tree and a development tree are being worked on simultaneously.

The kernel version numbers are pretty simple to understand. They use the following syntax:

major.minor.patch

At the time of this writing, the current stable kernel is 2.2.18. That is major version 2, minor version 2, and patch level 18. The minor version number tells you if the kernel is a stable release or a development release. Even minor numbers mean it is a stable kernel, while odd minor numbers mean it is a development kernel.

**Tip**

The minor version number specifies if the kernel is a stable release or a development release.

Occasionally you will see a kernel version given as 2.2.18-4. Sometimes distribution makers will add another patch level to show which internal version it is. They sometimes make small enhancements or feature additions to a kernel.

**Tip**

To see which kernel version your system is running, you can type `uname -r` or `uname -a`.

Development branch

The *development branch* of kernels is used to test new features and drivers. This development branch will eventually become the stable branch with a new minor version number. After that, a new development branch will be started. For example, the 2.3.x development branch will eventually become the 2.4.x stable branch. When 2.4.0 is released, a new 2.5.x development branch will be started.

Users commonly run development kernels on their systems. This is usually because the user requires a new feature or hardware driver. It is not recommended to run a development kernel on a production system unless that development kernel has been extensively tested. It is also recommended to keep an older stable kernel installed in case of problems.

Stable branch

The kernels released in the stable branch have undergone the most testing and debugging. Normally, only minor features are added between versions to reduce the risk of introducing bugs. Unless you need a new feature in a new stable release, it is best to stay on the current version shipped with your distribution.

Kernel types

Kernels can be configured one of two ways depending on how drives are configured. These ways are known as follows:

- ♦ Monolithic
- ♦ Modular

**Exam Tip**

Monolithic kernels have all drivers compiled into the kernel file, while modular kernels have drivers compiled as modules.

Monolithic kernels

A monolithic kernel has all driver support needed compiled into the main kernel itself. The benefit of this is that no modules need to be loaded by the user. A *module* is a software driver that can be loaded dynamically. As soon as the kernel boots, all hardware support is available.

The disadvantage to a monolithic kernel is that it is usually much larger than a modular kernel. It is common to compile into the kernel support for hardware that you do not currently use, but may need in the future, or in case of a failure. By compiling in these extra drivers, the kernel can become very large. A larger kernel is a slower kernel and requires more memory because all of the code is resident at all times.

Modular kernels

A modular kernel is set up so that most, if not all, of the needed hardware drivers are compiled as modules. The core kernel is much smaller than it is with a monolithic kernel, but the modules must be configured and loaded separately. This can make booting a system with a nonstandard disk controller more difficult. Therefore, it is recommended to have a more hybrid kernel. Consider compiling the core drivers needed by the system into the kernel itself and compiling other extra drivers as modules.



Most kernels are a combination of modular and monolithic. Usually some drivers are compiled into the kernel while others are compiled as modules.

Starting with version 2.0, the Linux kernel introduced a modular structure. Before this, all hardware drivers and special features had to be compiled into the kernel. When you installed a new piece of hardware, you were required to recompile the kernel, or to compile in hardware support for things you did not have currently, but may have eventually.

Linux now supports the use of module, which can be loaded dynamically as needed. When you install a new hardware device in your system, you do not have to recompile the kernel. You need only to load the appropriate module, which can be done without restarting the system.

Most modules are loaded at boot. But, if needed, a module can be loaded at any time. Most kernels are configured to load modules automatically when they are needed. This was handled by the `kerneld` process before the 2.2 kernel, but is now handled by the `kmod` process.

Managing modules

Objective**1.5 Kernel**

- **Manage kernel modules at runtime.** Learn which functionality is available through loadable kernel modules, and manually load and unload the modules as appropriate. Involves using the commands: `lsmod`, `insmod`, `rmmod`, `modinfo`, `modprobe`. Involves reviewing the files: `/etc/modules.conf` | `/etc/conf.modules` (* depends on distribution *), `/lib/modules/{kernel-version}/modules.dep`.

There are several tools to help load, unload, and display information about the modules on the system. These include the following:

- ♦ `insmod`
- ♦ `rmmod`
- ♦ `lsmod`
- ♦ `modinfo`
- ♦ `modprobe`
- ♦ `depmod`

Locating module files

Module files are stored in a directory tree off the `/lib/modules/kernel-version` directory.

- ♦ The `block` directory contains all of the modules for block devices. These include such things as tape drives, floppy drives, and RAID devices.
- ♦ The `cdrom` directory holds all modules for CD-ROMs. These are proprietary CD-ROM controllers, such as the old Creative Labs and Mitsumi interfaces. These are not for current CD-ROMs, which are EIDE/ATAPI or SCSI.
- ♦ The `fs` directory holds the file system modules. Almost all of the Linux supported file systems can be supported via modules. You should always compile in the file system that the root volume needs.
- ♦ The `ipv4` directory holds the IP networking modules. It also includes IP Masq modules to be used when performing network address translation. The `ipv6` directory holds the modules needed to support the new IP v6 specification.
- ♦ The `net` directory holds all network interface driver modules. It also contains drivers for network protocols, such as PPP or SLIP. If you install a new network card and need to find the module to load, you should check this directory.
- ♦ The `scsi` directory holds all SCSI device drivers. Modules to enable SCSI emulation with IDE devices, used with CD-R/W drives, are also stored here.

- ♦ The `video` directory holds the drivers for video capture devices. This is still in early development, so there are not many drivers yet.
- ♦ The `misc` directory holds everything else. All sound card drivers are kept here, as are game devices such as joysticks. Peripheral interface drivers for things such as serial ports, parallel ports, and USB are stored here.

Loading modules

Modules are normally loaded with the `insmod` command. The syntax for the command is as follows:

```
insmod module-name.o
```

The `.o` extension can be left off. If modules are stored in a nonstandard location you need to provide the full path to the `module.o` file. Table 13-1 shows all of the available options for `insmod`.

Table 13-1
insmod Options

Option	Function
<code>-f, --force</code>	Force the module to load, even if it was compiled for another kernel version.
<code>-h, --help</code>	Display help information on the available options.
<code>-k, --autoclean</code>	Make the module autoclean-able, so that it can be removed automatically when no longer needed.
<code>-L, --lock</code>	Prevent simultaneous loads of the same module.
<code>-m, --map</code>	Generate a load map for debugging information.
<code>-n, --noload</code>	Do not actually load the module. Only show simulated output.
<code>-p, --probe</code>	Enable probe mode, which checks to see if the module matches the currently loaded kernel.
<code>-q, --quiet</code>	Do not print unresolved symbol errors.
<code>-r, --root</code>	Allow root to load modules that are not owned by root.
<code>-s, --syslog</code>	Report errors via <code>syslog</code> .
<code>-S, --kallsyms</code>	Force <code>kallsyms</code> on modules (used in kernel debugging, not available on all versions of <code>insmod</code>).
<code>-v, --verbose</code>	Enable verbose output.

Continued

Table 13-1 (continued)

Option	Function
-x, --noexport	Do not export externs.
-y, --noksymoops	Do not add ksymoops symbols.
-Y, --ksymoops	Do add ksymoops symbols. (default)
-o <i>NAME</i> , --name= <i>NAME</i>	Change the internal module name to <i>NAME</i> .
-O <i>NAME</i> , --blob= <i>NAME</i>	Save the object as a binary blob in <i>NAME</i> (not available in all versions of <code>insmod</code>).
-P <i>PREFIX</i> , --prefix= <i>PREFIX</i>	Set the prefix for kernel or module symbols.



Modules are loaded using the `insmod` command.

Removing modules

Modules are removed with the `rmmmod` command. The syntax for this command is as follows:

```
rmmmod module-name
```

Table 13-2 shows the available options for `rmmmod`. For a module to be removed, it must not be in use, and it should be set as autoclean. The `-r` option can be used to remove modules recursively, by removing all modules referenced by the named module.

Table 13-2
rmmmod Options

Option	Function
-a, --all	Remove all unused autoclean-able modules.
-r, --stacks	Remove a module stack.
-s, --syslog	Output all messages to <code>syslog</code> instead of to the console.



Modules are removed using the `rmmmod` command.

Listing installed modules

Modules that are currently loaded can be listed with the `lsmod` command. The following is an example.

```
debian:~# lsmod
Module                Size  Used by
vfat                  10736  0 (autoclean)
fat                   31264  0 (autoclean) [vfat]
mousedev              3968   1
hid                   11776  0 (unused)
input                 3424   0 [mousedev hid]
usb-ohci              16464  0 (unused)
emu10k1               43344  1
```

The rightmost columns in the output show information about the module. If the module can be autocleaned that column will show `(autoclean)`, and if the module is currently not being used it will display `(unused)`. Module names in brackets show modules that depend on that module. Modules can also be listed by viewing the `/proc/modules` file.



To display all loaded modules, use the `lsmod` command.

Displaying module information

Modules can be queried for information by using the `modinfo` command. The syntax for this command is as follows:

```
modinfo module-file
```

The available options for `modinfo` are shown in Table 13-3. The following is an example of its use.

```
debian:~# modinfo -a emu10k1
Bertrand Lee, Cai Ying. (Email to: emu10k1-
devel@opensource.creative.com)
debian:~# modinfo -d emu10k1
Creative EMU10K1 PCI Audio Driver v0.7
Copyright (C) 1999 Creative Technology Ltd.
```

Table 13-3
modinfo Options

<i>Option</i>	<i>Function</i>
-a, --author	Display author information.
-d, --description	Display description information.
-n, --filename	Display the module filename.
-f <string>, --format <string>	Print the query string.
-p, --parameters	Display the module parameters.
-V, --version	Show the version of the modinfo.
-h, --help	Display option help information.



`modinfo` can display description and author information about a module.

Using modprobe

Adding and removing modules can be handled by another tool called `modprobe`. It provides a little more flexibility than the other tools give and can take some configuration information. When the Linux kernel loads or unloads a module automatically, it actually runs `modprobe` with the appropriate options. The available options are shown in Table 13-4.

Table 13-4
modprobe Options

<i>Option</i>	<i>Function</i>
-a, --all	Load all matching modules, instead of just the first one successfully loaded.
-c, --showconfig	Show the currently used configuration.
-d, --debug	Show internal information about the module stack.
-k, --autoclean	Set all loaded modules as autoclean.
-l, --list <i>PATTERN</i>	List all matching modules.
-n, --show	Do not actually load any modules. Show the output only if they had been loaded.
-q, --quiet	Do not output an error when a module fails to load.
-r, --remove	Remove modules or do an autoclean.
-s, --syslog	Send output to <code>syslog</code> instead of the console.

Option	Function
<code>-t, --type <i>moduletype</i></code>	Consider only modules of this type.
<code>-v, --verbose</code>	Print all commands as they are executed.
<code>-V, --version</code>	Show the version of modprobe.
<code>-C, --config <i>configfile</i></code>	Use the specified module config file instead of <code>/etc/modules.conf</code> .

The `modprobe` tool can be used to load and unload groups of modules. The `-a` option by itself will cause `modprobe` to load all modules defined in its configuration file. The `-r` option will have it unload all autoclean modules that are not currently being used.

The default `modprobe` configuration file is either `/etc/modules.conf` or `/etc/conf.modules`, depending on the distribution. The possible syntax for this file is as follows:

```
[add] above module module_list
    alias alias_name result
[add] below module module_list
define VARIABLE WORD
depfile=A_PATH
else
elseif EXPRESSION
endif
if EXPRESSION
include PATH_TO_CONFIG_FILE
insmod_opt=GENERIC_OPTIONS_TO_INSMOD
install module command ...
keep
[add] options module MODULE_SPECIFIC_OPTIONS
path=A_PATH
path[TAG]=A_PATH
pcimapfile=A_PATH
isapnmapfile=A_PATH
usbmapfile=A_PATH
[add] probe name module_list
[add] probeall name module_list
prune filename
post-install module command ...
post-remove module command ...
pre-install module command ...
pre-remove module command ...
remove module command ...
```


Don't be frightened by the look of this complex syntax. You will rarely use any more than the most basic lines. The following is an example of a `conf.modules` file on a workstation.

```
alias eth0 tulip
alias parport_lowlevel parport_pc
sb
options sb irq=5 dma=1
```

Usually the only options you will use are `alias` and `options`. Aliases allow you to give a module a different name. In this example, the `tulip` network driver has been aliased to the device name of `eth0`. If the `result` field of the alias is set to `off`, the module cannot be loaded. If the `result` field is set to `null`, the module load will always succeed but no module will actually be installed.

In the previous example, the module named `tulip`, a driver for a network card using the Tulip chipset, has been aliased to the name `eth0`. This enables the system to access the device at the new name.

A module name or alias can be configured via options. These options vary by module and are normally used to tell the driver which hardware information to use. If the `-k` option is used before the module name, the autoclean setting on the module will be removed.

In the previous example the `options` line configures the `sb` module, which is a driver for a Sound Blaster sound card. The `options` line specifies that the card has been configured for IRQ 5 and DMA channel 1. Not all drivers require you to tell them which hardware addresses the card uses.



Some distributions still use the filename of `conf.modules`, but the current name is `modules.conf`. Check your distribution to see which name it uses.

Using the following commands:

```
post-install module command ...
post-remove module command ...
pre-install module command ...
pre-remove module command ...
remove module command ...
```

You can have any command execute before or after a module loads or unloads. This capability can be useful for preparation or cleanup processes.



The `modprobe` tool can be used to perform many module management tasks.

Using depmod

Modules sometimes rely on other modules when loaded. If a module is loaded automatically, the system has to know which other modules to load. The `depmod` tool handles the creation of this dependency list, which is stored in `/lib/modules/kernel-version/modules.dep`. The following is a partial example:

```
/lib/modules/2.2.17/fs/fat.o:
/lib/modules/2.2.17/fs/msdos.o: /lib/modules/2.2.17/fs/fat.o
/lib/modules/2.2.17/fs/ntfs.o:
/lib/modules/2.2.17/fs/vfat.o: /lib/modules/2.2.17/fs/fat.o
/lib/modules/2.2.17/misc/emu10k1.o:
/lib/modules/2.2.17/misc/soundcore.o
/lib/modules/2.2.17/misc/parport.o:
/lib/modules/2.2.17/misc/parport_pc.o:
/lib/modules/2.2.17/misc/parport.o
/lib/modules/2.2.17/misc/soundcore.o:
/lib/modules/2.2.17/net/3c501.o:
/lib/modules/2.2.17/net/3c503.o:
/lib/modules/2.2.17/net/8390.o
/lib/modules/2.2.17/net/3c505.o:
```

Usually `depmod` is run at boot by an `rc` file, but if you manually edit the `modules.conf` file, you should run it again. Table 13-5 shows the available options for `depmod`.

Table 13-5
depmod Options

<i>Option</i>	<i>Function</i>
<code>-a, --all</code>	Search for all modules in all directories specified by the <code>modules.conf</code> file.
<code>-A</code>	Compare file time stamps and run a <code>depmod -a</code> if anything has changed.
<code>-e, --errsyms</code>	Show all of the unresolved symbols for each module.
<code>-n, --show</code>	Output the module dependency information to the console instead of the <code>modules.dep</code> file.
<code>-s, --syslog</code>	Output all error information to <code>syslog</code> instead of the console.
<code>-v, --verbose</code>	Enable verbose mode, which lists the name of each module as it is loaded.
<code>-q, --quiet</code>	Do not display missing symbol errors.
<code>-V, --version</code>	Show the <code>depmod</code> version information.
<code>-r, --root</code>	Override the default security precaution that keeps module utilities from loading modules owned by nonroot users.



You should run `depmod` whenever modules change.

Reconfiguring and Installing a New Kernel

Objective

1.5 Kernel

- **Reconfigure, build and install a custom kernel and modules.** Obtain and install approved kernel sources and headers (from a repository at your site, CD, kernel.org, or your vendor); Customize the kernel configuration (i.e., reconfigure the kernel from the existing `.config` file when needed, using `oldconfig`, `menuconfig` or `xconfig`); Make a new Linux kernel and modules; Install the new kernel and modules at the proper place; Reconfigure and run `lilo`. N.B.: This does not require to upgrade the kernel to a new version (full source nor patch). Requires the commands: `make` (`dep`, `clean`, `menuconfig`, `bzimage`, `modules`, `modules_install`), `depmod`, `lilo`. Requires reviewing or editing the files: `/usr/src/linux/.config`, `/usr/src/linux/Makefile`, `/lib/modules/{kernelversion}/modules.dep`, `/etc/conf.modules` | `/etc/modules.conf`, `/etc/lilo.conf`.

Compiling a new kernel is a very straightforward process. First, you have to get the new kernel source or patch your current source up to the version you want to compile. You then have to configure the kernel and enable the options, features, and drivers that you need. Finally, you compile the new kernel and install it so that it can be booted.

Obtaining the kernel source

The kernel source code is available in several locations. Where you get your source code depends on what you intend to do. The kernel source should be expanded to the `/usr/src/linux` directory. If you need to keep more than one version around, it is a good idea to have each one in its own directory under `/usr/src` and to create a link to `/usr/src/linux` for the one you need to compile and use.

Using the distribution source

Distributions include the source code for their shipping kernel on the CD-ROM. If you need to change only the existing kernel, it is best to use this source, since it will include any tweaks the distribution maker provides. With an RPM-based system, the package containing the kernel source files are normally named `kernel-source-<version>-<distribution patch level>.rpm`. You will also need to install the kernel header files, normally stored in `kernel-headers-<version>-<distribution patch level>.rpm`. If you use Debian, you can get the kernel source by issuing the following:

```
apt-get install kernel-source-<version>
```

and

```
apt-get install kernel-headers-<version>
```

New precompiled kernels are also available with `apt-get`, but you should use those at your own risk.



Even if you never plan to recompile your kernel, you may still need to install the kernel source files because some applications require them when compiling.

Getting new source

If you plan to update to a new kernel version, you will probably need to get it from a main kernel site. Most distribution makers do not keep updated kernels for their distributions because of support reasons. The official kernel source site is <http://www.kernel.org>. You should use a close mirror site to keep traffic off the main site. The list of mirrors is available at <http://www.kernel.org/mirrors/>.

On the site you will see a directory for each major kernel tree from v1.0 up to the current. The kernel source code is in each directory compressed with either `gzip` or `bzip2`. Usually the `bzip2` files are smaller and take less time to download. You may also notice patch files. Once you download a full kernel source package, you need to download only the patches to stay current. This can greatly reduce download time.



Tip

A full kernel package is about 15MB, so you can save a lot of time by using patches.

Updating your source with patches

Patches are much smaller than a full kernel package. A kernel package updated to the latest version with patches is exactly the same as the latest version you can download as a full package. If you need to increment up more than one patch level, you must download each patch level up to the version you want. For example, if you currently have kernel v2.2.13 and want to move to v2.2.17, you must download patches 2.2.14, 2.2.15, 2.2.16, and 2.2.17.

To apply a patch you use the `patch` command. The following is an example:

```
debian:/usr/src# zcat patch-2.2.18.gz | patch -p0
patching file `linux/net/socket.c'
patching file `linux/net/sunrpc/auth.c'
patching file `linux/net/sunrpc/auth_null.c'
patching file `linux/net/sunrpc/auth_unix.c'
patching file `linux/net/sunrpc/clnt.c'
patching file `linux/net/sunrpc/pmap_clnt.c'
patching file `linux/net/sunrpc/sched.c'
patching file `linux/net/sunrpc/stats.c'
```

```

patching file `linux/net/sunrpc/sunrpc_syms.c'
patching file `linux/net/sunrpc/svc.c'
patching file `linux/net/sunrpc/svcsock.c'
patching file `linux/net/sunrpc/sysctl.c'
patching file `linux/net/sunrpc/xdr.c'
patching file `linux/net/sunrpc/xprt.c'
patching file `linux/net/unix/af_unix.c'
patching file `linux/net/unix/garbage.c'
patching file `linux/net/x25/af_x25.c'
patching file `linux/net/x25/x25_dev.c'
patching file `linux/net/x25/x25_out.c'
patching file `linux/scripts/Lindent'
patching file `linux/scripts/ksymloops/ksymloops.c'
patching file `linux/scripts/kwhich'
patching file `linux/scripts/mkdep.c'

```

Either `cat` or `zcat` the patch file to the `patch -p0` command, depending on if the file is compressed with `gzip` or not.

Adding features with patches

Occasionally you may need a feature not supplied in the standard kernel distribution. Many times enhancements can be added to the kernel using a patch, similar to the patches in the previous section. These are written for specific kernel versions and should be used only with the correct kernel source. They are applied the same way as other patches. The following is an example.

```

debian:/usr/src# zcat linux-2.2.18-reiserfs-3.5.29-patch.gz |
patch -p0
patching file `linux/Documentation/Configure.help'
patching file `linux/fs/Config.in'
patching file `linux/fs/Makefile'
patching file `linux/fs/buffer.c'
patching file `linux/fs/filesystems.c'
patching file `linux/fs/inode.c'
patching file `linux/fs/reiserfs/Makefile'
patching file `linux/fs/reiserfs/README'
patching file `linux/fs/reiserfs/bitmap.c'
patching file `linux/fs/reiserfs/buffer.c'
patching file `linux/fs/reiserfs/buffer2.c'
patching file `linux/fs/reiserfs/dir.c'
patching file `linux/fs/reiserfs/do_balan.c'
patching file `linux/fs/reiserfs/file.c'
patching file `linux/fs/reiserfs/fix_node.c'
patching file `linux/fs/reiserfs/hashe.c'
patching file `linux/fs/reiserfs/ibalance.c'
patching file `linux/fs/reiserfs/inode.c'
patching file `linux/fs/reiserfs/ioctl.c'
patching file `linux/fs/reiserfs/journal.c'
patching file `linux/fs/reiserfs/lbalance.c'
patching file `linux/fs/reiserfs/namei.c'
...

```

Verifying the source

The kernel mirror sites supply a checksum file so you can verify the kernel source code and patches you download. This requires the use of GnuPG, which is available with Red Hat in the `gnupg-<version>-<patch>.rpm` file. Debian users can use the `apt-get install gnupg` command. Before you can do a check with GnuPG you need to get the kernel maintainer's public key. This is done with the following command:

```
debian:~# gpg --keyserver wwwkeys.pgp.net --recv-keys
0x517D0F0E
gpg: requesting key 517D0F0E from wwwkeys.pgp.net ...
gpg: key 517D0F0E: public key imported
gpg: /root/.gnupg/trustdb.gpg: trustdb created
gpg: Total number processed: 1
gpg:             imported: 1
```

The following is an example of a successful signature check. You may receive other information after the first two lines, but just ignore that subsequent information.

```
debian:~# gpg --verify linux-2.2.17.tar.bz2.sign linux-
2.2.17.tar.bz2
gpg: Signature made Tue Oct 10 02:39:07 2000 EDT using DSA key
ID 517D0F0E
gpg: Good signature from "Linux Kernel Archives Verification
Key <ftpadmin@kernel.org>"
```

The following is an example of an unsuccessful signature check. If you receive this error, download the kernel source and signature file again, and check them again.

```
debian:~# gpg --verify patch-2.2.18.gz.sign patch-2.2.18.gz
gpg: Signature made Sun Dec 10 19:46:22 2000 EST using DSA key
ID 517D0F0E
gpg: BAD signature from "Linux Kernel Archives Verification Key
<ftpadmin@kernel.org>"
```

Configuring the kernel

The most confusing part of installing a new kernel is going through the many configuration options. This quickly overwhelms new users and if done incorrectly can cause the system not to boot or cause it to lack required hardware support. The configuration process can be done from one of four different interfaces. These interfaces are the following:

- ♦ `config`
- ♦ `menuconfig`
- ♦ `xconfig`
- ♦ `manual`

The first three are `make` commands that are run from the `/usr/src/linux` directory. Which one you use depends on the graphical capability of the system, the installed support packages, and your comfort level with the interface.

Using `make config`

The most basic menu interface to configuring the kernel is by using `make config`. The following is an example of the `make config` interface.

```

debian:/usr/src/linux# make config
rm -f include/asm
( cd include ; ln -sf asm-i386 asm)
/bin/sh scripts/Configure arch/i386/config.in
#
# Using defaults found in arch/i386/defconfig
#
*
* Code maturity level options
*
Prompt for development and/or incomplete code/drivers
(CONFIG_EXPERIMENTAL) [N/y/?]
*
* Processor type and features
*
Processor family (386, 486/Cx486, 586/K5/5x86/6x86,
Pentium/K6/TSC, PPro/6x86MX) [PPro/6x86MX]
    defined CONFIG_M686
/dev/cpu/microcode - Intel P6 CPU microcode support
(CONFIG_MICROCODE) [N/y/m/?] (NEW)
/dev/cpu/*/msr - Model-specific register support
(CONFIG_X86_MSR) [N/y/m/?] (NEW)
/dev/cpu/*/cpuid - CPU information support (CONFIG_X86_CPUID)
[N/y/m/?] (NEW)
Maximum Physical Memory (1GB, 2GB) [1GB]
    defined CONFIG_1GB
Math emulation (CONFIG_MATH_EMULATION) [N/y/?]
MTRR (Memory Type Range Register) support (CONFIG_MTRR) [N/y/?]
Symmetric multi-processing support (CONFIG_SMP) [Y/n/?]
*
* Loadable module support
*
Enable loadable module support (CONFIG_MODULES) [Y/n/?]

```

As you can see, the output is not exactly user friendly. Each new kernel version has more options to go through, and the `config` method can take considerable time to work through. The benefit of this option is that no other interface packages are required. Answer Y or N to each option, and if you need help, use the ? key. Usually the help is extensive and informative. The following is an example:

CONFIG_MODULES:

```
Kernel modules are small pieces of compiled code which can be
inserted in or removed from the running kernel, using the
programs insmod and rmmod. This is described in the file
Documentation/modules.txt, including the fact that you have
to say "make modules" in order to compile the modules that
you chose during kernel configuration. Modules can be device
drivers, file systems, binary executable formats, and so on.
If you think that you may want to make use of modules with
this kernel in the future, then say Y here. If unsure, say Y.
```

Since `make config` is very cumbersome to use, a `make oldconfig` option is available to keep from having to go through the entire process every time. The `make oldconfig` will take an existing configuration file and ask only questions that were added to the new kernel level you are compiling.

Using `make menuconfig`

The next level of interface for configuring a new kernel is `make menuconfig`. Figure 13-1 shows the interface. The `menuconfig` interface is much easier and faster to use than `config`, but requires the `ncurses` library to be installed. If you try to use this interface and get the following error, it means you are missing this library.

```
debian:/usr/src/linux# make menuconfig
rm -f include/asm
( cd include ; ln -sf asm-i386 asm)
make -C scripts/lnxdialog all
make[1]: Entering directory `/usr/src/linux/scripts/lnxdialog'
gcc -Wall -Wstrict-prototypes -O2 -fomit-frame-pointer -DLOCALE
-DCURSES_LOC="<curses.h>" -c -o lnxdialog.o lnxdialog.c
In file included from lnxdialog.c:22:
dialog.h:29: curses.h: No such file or directory
make[1]: *** [lnxdialog.o] Error 1
make[1]: Leaving directory `/usr/src/linux/scripts/lnxdialog'
make: *** [menuconfig] Error 2
```

Debian users can use `apt-get install libncurses5-dev` to get the needed package, and Red Hat users should install the `ncurses-devel-5.<minor version>-<patch level>.i386.rpm` package.

As you can see, this interface provides an easy-to-use menu. The arrow keys navigate the menu. Features, options, and drivers are chosen with the spacebar. An `*` means the item will be compiled into the kernel, while an `M` means it will be compiled as a module.

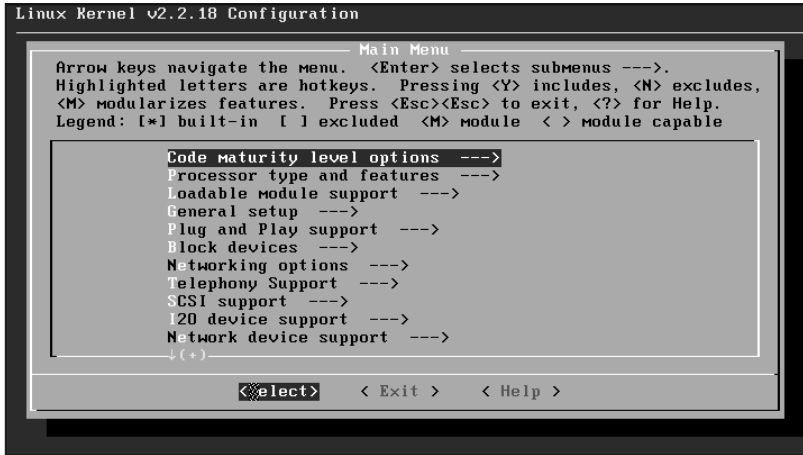


Figure 13-1: menuconfig Interface

Using make xconfig

The easiest interface to use is `make xconfig`. Figure 13-2 shows an example of this interface. This provides a nice point-and-click interface, but requires the X Window system to be installed and functional. Like `menuconfig`, `xconfig` is a menu structure, and options are selected with `*` meaning compiled into the kernel and `M` meaning compiled as a module.

The `make xconfig` interface requires the installation of the TCL/TK toolkit. Most Linux distributions install this toolkit by default.

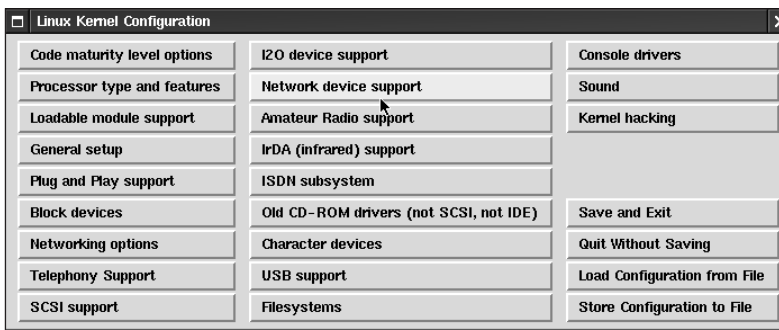


Figure 13-2: xconfig Interface

Manually editing the .config file

By default, all of the configuration options save the final result as `/usr/src/linux/.config`. The following is sample output from this file.

```
#
# Processor type and features
#
# CONFIG_M386 is not set
# CONFIG_M486 is not set
# CONFIG_M586 is not set
# CONFIG_M586TSC is not set
CONFIG_M686=y
CONFIG_X86_WP_WORKS_OK=y
CONFIG_X86_INVLPG=y
CONFIG_X86_BSWAP=y
CONFIG_X86_POPAD_OK=y
CONFIG_X86_TSC=y
CONFIG_X86_GOOD_APIC=y
# CONFIG_MICROCODE is not set
# CONFIG_X86_MSR is not set
# CONFIG_X86_CPUID is not set
CONFIG_1GB=y
# CONFIG_2GB is not set
# CONFIG_MATH_EMULATION is not set
# CONFIG_MTRR is not set
CONFIG_SMP=y
```

The format is simple, but it is not recommended you edit this file unless you need to enable or remove a simple feature.



The kernel configuration settings are stored in `/usr/src/linux/.config`.

Compiling the kernel

The actual process of compiling the kernel is very simple. A few steps need to be followed, but they are almost always the same sequence with the same result. Some of the steps may take a considerable amount of time, depending on the speed of the system and the amount of available memory.

Building the dependency list

Depending on your choices during configuration, some source code will depend on other source code. This dependency setup is configured with the `make dep` command.

Cleaning up temporary files

Some temporary files are created during the compile process, so any time you are doing a new compile you should use the `make clean` command. If you do not do this and have leftover temporary files, you can get very strange errors during the compile. So if in doubt, use the `make clean` command.

Compiling the kernel

Finally, the moment of truth! The kernel compile process can take a large amount of time, depending on the system. Several commands are available to use, depending

on if you want the kernel automatically installed after compile. Table 13-6 lists the common build commands.

Most current kernels are too big, even with modules, to `gzip` and will not install. Therefore, you should always use the `bzip2` option when creating a new kernel. You should also install the kernel manually, so you can give it an appropriate name and put it in a known place instead of the default the `Makefile` uses. The following is an example of what happens if you create a `gzip`-compressed kernel, and it is too large.

```
Root device is (3, 1)
Boot sector 512 bytes.
Setup is 1292 bytes.
System is 533 kB
System is too big. Try using bzImage or modules.
make[1]: *** [zImage] Error 1
make[1]: Leaving directory `/usr/src/linux/arch/i386/boot'
make: *** [zImage] Error 2
```

Table 13-6
Kernel Build Commands

<i>Command</i>	<i>Function</i>
<code>make zImage</code>	Creates a <code>gzip</code> 'd kernel image that must be installed manually.
<code>make bzImage</code>	Creates a <code>bzip2</code> 'd kernel image that must be installed manually.
<code>make zlilo</code>	Creates a <code>gzip</code> 'd kernel image that is installed automatically into LILO.
<code>make bzlilo</code>	Creates a <code>bzip2</code> 'd kernel image that is installed automatically into LILO.
<code>make bzdisk</code>	Creates a boot disk with the new kernel.

Compiling and installing modules

Drivers configured to be modules are not compiled with the rest of the kernel. To compile the modules, use the `make modules` command. Depending on the number of modules to be compiled, this process may take a considerable amount of time.

Finally, to install the modules use the `make modules_install` command. This command will place the modules in the correct directory structure under `/lib/modules/<new kernel version>`. You should also run a `depmod -a` to create a new module dependency list.

Installing the kernel

The next step is to install the new kernel, unless you used the `make zlilo` or `bzlilo` commands. Installing the new kernel manually so that nothing gets

overwritten by accident is much safer than automatic installation. The new kernel should be located in `/usr/src/linux/arch/i386/boot/bzImage`, assuming you are on an i386 system. To move the kernel to the correct place, follow these steps:

1. Copy the new kernel to the appropriate location, which is normally `/boot`. Some other systems store the kernel in `/`. If you are unsure, check your `/etc/lilo.conf` file for the kernel location.
2. Rename the new kernel to a more informative name than `bzImage`. Something like `vmlinuz-2.2.18-test` is a good idea. If the kernel has a special feature, you could use that in the name, such as `vmlinuz-2.2.18-reiserfs`.
3. Copy the `/usr/src/linux/System.map` file to the new kernel directory. This file is used during booting.

Creating a ramdisk

Some systems require an `initrd` image to be created. An `initrd` image is an initial ramdisk that contains any modules that are needed to boot the system and get to the root file system. If you have all needed drivers to boot and get to the root file system compiled into the kernel, you do not need to create this image. But, if your main disk controller driver was compiled as a module, you must create the `initrd` image. Usually, compiling the drivers needed to boot into the kernel is a good idea. To create the `initrd` image use the following command:

```
mknitrd /boot/image-name kernel-version
```

You will see the `initrd` image loaded at boot, but it is removed from memory when no longer needed.



You should make an `initrd` image whenever a driver needed to access the root volume is compiled as a module.

Configuring LILO

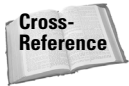
Before you can reboot to use your new kernel, you have to configure LILO to boot to the new image. The smart idea is to keep the current kernel configuration and add the new kernel so it can be fully tested before removing the old. Assume the current `/etc/lilo.conf` file has the following entry:

```
image=/boot/vmlinuz-2.2.14-5.0
    label=linux
    read-only
    root=/dev/hda5
```

To install the new kernel you would add a second entry, for example:

```
image=/boot/vmlinuz-2.2.14-5.0
    label=linux-old
    read-only
    root=/dev/hda5
```

```
image=/boot/vmlinuz-2.2.18-reiserfs
label=linux
read-only
root=/dev/hda5
```



LILO is covered in detail in Chapter 8.

Depending on the current configuration, you may need to add `prompt` and `timeout` settings so that you can choose which kernel to boot. For example:

```
prompt
timeout=50
```

This way you can boot to the old kernel from the LILO prompt, if needed. Be sure to run `/sbin/lilo -v` so that the new boot sector is written.



`/sbin/lilo` must be run whenever a new kernel is installed or `/etc/lilo.conf` is edited.

Testing the new kernel

The final step is to reboot the system and test out the new kernel. Make sure the kernel boots correctly with no errors. Check the `syslog` for any kernel errors. Make sure all hardware devices work as expected, as well as any other required options.

If the kernel checks out and no problems are found, you can remove the old kernel and change the `/etc/lilo.conf` file. Be sure to run `/sbin/lilo -v` again to re-create the boot sector.

If there is a problem with the kernel, you can boot back to the old version, and work to fix the problem. Usually a kernel problem relates to a configuration option that was not enabled or set incorrectly.

Key Point Summary

Working with the Linux kernel can be one of the most daunting and scary tasks for a new user. Module support greatly enhances the ease of use for adding new hardware and cuts down on reboots and downtime. Recompiling the kernel is a task that most users end up doing some time or another. The configuration interfaces have been greatly improved, but the configuration decisions still remain the same and important.

- ♦ The kernel manages the system's memory and CPU time for the user processes.
- ♦ The kernel version numbers follow the format of `<major>.<minor>.<patch level>`.

- ♦ Even-numbered minor numbers represent stable kernels, while odd-numbered minor numbers represent development kernels.
- ♦ A monolithic kernel has the drivers compiled into the kernel.
- ♦ A modular kernel has the drivers compiled as modules.
- ♦ The module driver files are stored in `/lib/modules/<kernel version>`.
- ♦ The `insmod` command is used to load modules.
- ♦ The `rmmod` command is used to remove modules.
- ♦ The `modinfo` command is used to display module info.
- ♦ The `modprobe` command can be used to perform many module tasks.
- ♦ The `modules.conf` or `conf.modules` file is used to change how `modprobe` functions.
- ♦ The module dependency list is created with `depmod`.
- ♦ Kernel patches can be used to update a kernel source package to the latest.
- ♦ All patches between kernel versions must be applied.
- ♦ GnuPG can be used to verify the integrity of the kernel source code.
- ♦ `make config` provides a simple kernel configuration interface.
- ♦ `make menuconfig` provides a text mode kernel configuration interface.
- ♦ `make xconfig` provides a graphical kernel configuration interface.
- ♦ The kernel configuration file is saved to `/usr/src/linux/.config`.
- ♦ The commands to compile a kernel, in order, are as follows: `make config/menuconfig/xconfig`, `make dep`, `make clean`, and `make bzImage/zImage/zlilo/bzlilo/bzdisk`.
- ♦ The compiled kernel is stored in `/usr/src/linux/arch/i386/boot/bzImage/zImage`.
- ♦ The kernel and `System.map` should be copied to the boot directory.
- ♦ The `/etc/lilo.conf` file should be edited to add the new kernel.
- ♦ `/sbin/lilo -v` should be run to re-create the new boot sector for the new kernel.
- ♦ The `mkinitrd` command should be run to make an initial ramdisk, if needed.
- ♦ An initial ramdisk is needed if the disk controller driver to access the root volume was compiled as a module.



STUDY GUIDE

The following questions and exercises will allow you to review the information covered in this chapter. Take your time when completing this section of the chapter, carefully reviewing any questions that you may have. Answering the question correctly is not as important as understanding the answer, so review any material that you might still be unsure of. Being comfortable with the questions and answers presented here will help you be more prepared for the certification exam questions.

Assessment Questions

1. Which command should be run when the module's configuration file is edited?
 - A. `depmod -a`
 - B. `modprobe -d`
 - C. `moddep -a`
 - D. `echo 1 > /proc/modules`
2. A _____ kernel has everything compiled into the main kernel file.
3. Which kernel is released under the stable branch?
 - A. 2.3.14
 - B. 3.3.14
 - C. 3.2.14
 - D. 4.3.13
4. The command `modinfo -d sound-driver` would display what?
 - A. Module size
 - B. Module description
 - C. If the module is loaded
 - D. If the module is set as autoclean

5. Which directive in the module's configuration file will give the module a new name?
- A. `alias newname oldname`
 - B. `oldname alias newname`
 - C. `oldname option=alias newname`
 - D. `alias oldname newname`
6. How many patches would be required to update from kernel 2.1.13 to 2.1.21?
- A. 8
 - B. 7
 - C. 2
 - D. 1
7. Which command is used to check the integrity of the kernel source package?
- A. PGP
 - B. MD5
 - C. DES
 - D. GPG
8. To configure the kernel using the most basic interface, you would type _____.
9. The kernel configuration is stored in a file named what?
- A. `.config`
 - B. `config`
 - C. `.kconfig`
 - D. `kconfig`
10. The _____ file must be edited if a second kernel is installed on a system.
11. Which command should be run to create a new boot sector when a new kernel is installed?
- A. `mkinitrd`
 - B. `make install`
 - C. `lilo`
 - D. `kerneld`

12. Which module directory holds the sound drivers?
- A. sound
 - B. dsp
 - C. misc
 - D. media
13. Which command(s) will remove all autoclean-able modules?
- A. `rmmod -r`
 - B. `rmmod -a`
 - C. `modprobe -r`
 - D. `modprobe -a`
14. Which command is used to apply kernel patches?
- A. update
 - B. patch
 - C. upgrade
 - D. `mkinitrd`
15. Which library must be installed to use `make menuconfig`?
- A. TCL/TK
 - B. GTK
 - C. PLIB
 - D. ncurses
16. Which library must be installed to use `make xconfig`?
- A. TCL/TK
 - B. GTK
 - C. PLIB
 - D. ncurses
17. Which command creates a source code dependency list when compiling a new kernel?
- A. `depmod`
 - B. `make depmod`
 - C. `make depend`
 - D. `make dep`

18. The _____ command lists all currently loaded modules.
19. Which `menuconfig/xconfig` menu has the disk configuration options?
 - A. Block devices
 - B. File systems
 - C. SCSI devices
 - D. General setup
20. Which file holds the module dependency list (no path)?

Scenarios

1. You have compiled a new kernel, but it will not load. The system is a standard server with a SCSI RAID controller as its only disk interface. The kernel was set up to be a completely modular kernel to help increase performance and reduce downtime. What could be the problem?
2. You have compiled a new kernel on a system and edited the `/etc/lilo.conf` file. After copying the kernel files to the `/boot` directory and rebooting the system, you find that you cannot choose the new kernel at the LILO prompt. What could be the problem?
3. While running `make zImage` you get an error. The error reports that the image file is too large. What can you do to fix this problem?

Lab Exercises

Lab 13-1 Installing the Kernel

This lab will walk you through obtaining, configuring, and compiling a new kernel. Be warned that some of the configuration decisions will vary by system, so use your best judgment.

Obtaining the kernel

In this part of the lab you will get a new kernel package and a patch so that you can practice fully.

1. FTP to `ftp.kernel.org` and go to the `/pub/linux/kernel/v2.2` directory. If your distribution shipped with kernel v2.4 or later, use a package from that directory.

2. Download a kernel package that is one earlier than the latest, and a patch to update it to the latest. The examples shown will be with package 2.2.17 updated to 2.2.18.
3. Uncompress the kernel package to the `/usr/src` directory to make a `/usr/src/linux` directory.
4. Apply the patch using a command such as the following:

```
zcat patch-2.2.18.gz | patch -p0
```

or

```
patch -p0 < patch-<version>
```

Configuring the kernel

The examples in this section use the X Window configuration interface. Use the most appropriate interface for your system. At any time during the configuration you can choose the help option to get valuable information. Use this information in determining if you need to enable or disable an option.

1. From inside of the X Window system, execute `make xconfig` from the `/usr/src/linux` directory. Figure 13-3 shows the main menu.

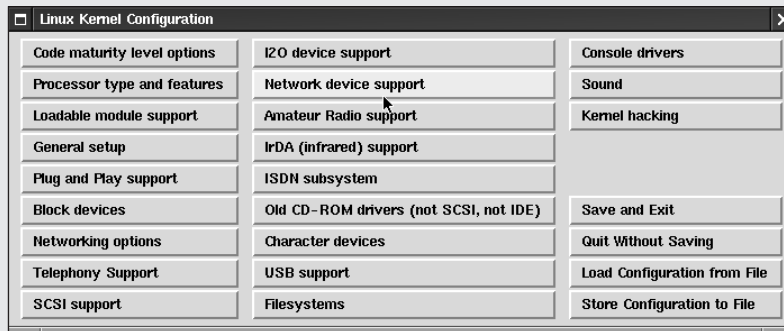


Figure 13-3: Main menu

2. Choose the “Code maturity level options” choice from the main menu. It is shown in Figure 13-4.

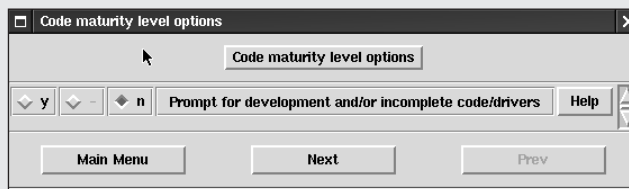


Figure 13-4: Code maturity level options

- If you want to use new drivers that may not be stable, select Y from the menu. The new drivers may work fine, or they may cause problems so choose wisely.
- Choose Next to go to the Processor types and features screen, as shown in Figure 13-5. Choose the correct processor type here so the kernel optimization can be done correctly. Setting the correct processor can increase performance; however, a kernel compiled for a new processor may not work on an older one, so this should be done system-by-system.

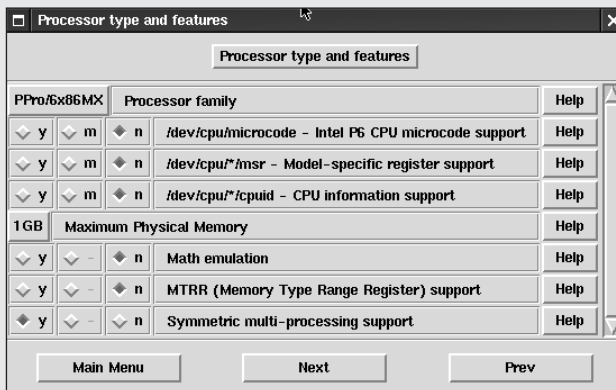


Figure 13-5: Processor types and features

- Continue through the rest of the screens and be sure to read the help next to an option if you have any questions. Table 13-7 shows the purpose of the main menu options.

Table 13-7
Main Menu Options

<i>Menu</i>	<i>Options</i>
Code maturity level options	Set whether or not to display incomplete or untested options during configuration.
Processor type and features	Optimize the kernel for the specific processor in the system, as well as enable multiprocessor support.
Loadable module support	Enable module support is enabled, as well as autoloading support.
General setup	Enable things such as network and PCI support, as well as support for other bus types and internal settings.
Plug and Play support	Enable or disable Plug-and-Play support.

Continued

Table 13-7 (continued)

<i>Menu</i>	<i>Options</i>
Block devices	Configure block devices, usually disks and media.
Networking options	Configure the network features supported by the kernel.
Telephony support	Preliminary telephony support is now included in the kernel.
SCSI support	Configure SCSI options and available drivers.
I2O device support	Enable the Intelligent Input/Output architecture.
Network device support	Go through the many supported network devices. It is recommended to configure most of the adapters as modules, in case you need to switch network cards later.
Amateur radio support	Enable and configure amateur radio support.
IrDA support	Enable and configure infrared support for those systems that have IrDA ports.
ISDN subsystem	Configure Linux's excellent ISDN support.
Old CD-ROM drivers	Configure and enable non-IDE and non-SCSI CD-ROM support. These are older CD-ROM drives.
Character devices	Configure terminal support devices.
USB support	Configure and select the USB chipset used on the system. If you are unsure, you may want to compile the chipsets as modules and try loading each one.
Filesystems	Enable file system support. Many of these drivers can be done as modules, which is a good idea in case you should ever need them later.
Console drivers	Allow configuration of graphics mode at the console. This is normally used for graphical boot packages.
Sound	Enable sound support and configure sound hardware.
Kernel hacking	Used for debugging purposes. Be careful with any changes.

Compiling the kernel

The following steps compile the kernel to a `bzip2` image, but do not automatically install it.

1. Run the `make dep` command.
2. Run the `make clean` command.

3. Run the `make bzImage` command.
4. The kernel will be compiled and placed in `/usr/src/linux/arch/i386/boot`, with the name of `bzImage`.

Compiling and installing modules

These steps compile and install the module files you configured.

1. Run the `make modules` command.
2. Run the `make modules_install` command.
3. Run the `depmod -a` command.
4. Check the `/lib/modules/<new kernel version>` directory to be sure the module files are installed correctly.

Copying the kernel

The next step is to copy the kernel to the correct location.

1. Copy the `/usr/src/linux/arch/i386/boot/bzImage` file to the directory of the current kernel, normally `/boot`. If you are unsure, check the `/etc/lilo.conf` file for the current kernel directory.
2. Copy the `/usr/src/linux/System.map` to the directory with the new kernel.

Reconfiguring LILO

Before a kernel can be booted you must reconfigure LILO.

1. Add a new kernel block to `/etc/lilo.conf`. Be sure to rename the old kernel label if needed. For example:

```
image=/boot/vmlinuz-2.2.18-reiserfs
label=linux
read-only
root=/dev/hda5
```
2. Add a prompt and timeout statement if needed so that you have a chance to boot to the old kernel if needed.
3. Run a `/sbin/lilo -v` to create the new boot sector.

Testing the new kernel

The final step is to reboot the system and choose the new kernel. Make sure it boots correctly with no errors. Also check all hardware to make sure it is supported as expected.

Answers to Chapter Questions

Chapter Pre-Test

1. A modular kernel is a kernel with all drivers compiled into the kernel file.
2. A new kernel should be compiled when you need new hardware or feature support.
3. 5 patches are required: 2.2.14, 2.2.15, 2.2.16, 2.2.17, and 2.2.18.
4. `make modules_install`
5. `/etc/lilo.conf`
6. TCL/TK
7. `rmmmod`
8. Use the `modinfo -a <module name>` command.
9. `/usr/src/linux/.config`
10. `make bzImage`

Assessment Questions

1. **A.** Whenever the `modules.conf` file is modified, you should update the module dependencies by running `depmod`. For more information, see the “Using `depmod`” section.
2. **monolithic.** Monolithic kernels have all drivers compiled into the kernel instead of using modules. For more information, see the “Kernel types” section.
3. **C.** The minor version number specifies if the kernel is stable or development. For more information, see the “Kernel development” section.
4. **B.** `modinfo` is used to display module information. The `-d` parameter shows the description. For more information, see the “Displaying module information” section.
5. **D.** The `alias` directive specifies a second name for a module. For more information, see the “Using `modprobe`” section.
6. **A.** Patches 2.2.14, 2.2.15, 2.2.16, 2.2.17, 2.2.18, 2.2.19, 2.2.20, and 2.2.21 would be required. For more information, see the “Updating your source with patches” section.
7. **D.** The kernel sources are signed with GnuPG. This way you can be sure they have not been tampered with. For more information, see the “Verifying the source” section.
8. `make config`. This interface provides a simple text mode mechanism for configuring the kernel. For more information, see the “Using `make config`” section.

9. **A.** The saved kernel configuration is stored in the `.config` file. For more information, see the “Manually editing the `.config` file” section.
10. **A.** `/etc/lilo.conf`. If this file is not edited, the system will not know about the new kernel. Do not forget to run `lilo` to install the new boot sector. For more information, see the “Configuring LILO” section.
11. **C.** Running `lilo` installs a new boot sector after the `lilo.conf` file is modified. The `mkinitrd` command creates an initial ramdisk, if needed. The `make install` command is used to install compiled software, and answer `D` is invalid. For more information, see the “Configuring LILO” section.
12. **C.** The other choices are invalid. For more information, see the “Locating module files” section.
13. **A and C.** The `rmmod -a` command removes all unused modules and the `modprobe -a` command loads all matching modules. For more information, see the “Removing modules” section.
14. **B.** Answers `A` and `C` are invalid. The `mkinitrd` command is used to create an initial ramdisk image. For more information, see the “Updating your source with patches” section.
15. **D.** `TCL/TK` is required to run `make xconfig`. For more information, see the “Using `make xconfig`” section.
16. **A.** The `ncurses` library is required for `make menuconfig`. For more information, see the “Using `make menuconfig`” section.
17. **D.** The `depmod` command is used to check module dependencies, after they are compiled. For more information, see the “Using `depmod`” section.
18. `lsmod`. For more information, see the “Listing installed modules” section.
19. **A.** Because disks are considered block devices, they are configured under this item. For more information, see the “Configuring the kernel” section.
20. `modules.dep`. The `depmod` command creates this module dependency list file. For more information, see the “Using `depmod`” section.

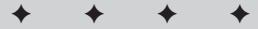
Scenarios

1. You need to create an initial ramdisk by using the `initrd` command. Without this `initrd` image, the system cannot get to the root file system, since the needed driver was compiled as a module.
2. You forgot to run `/sbin/lilo`, and the new boot sector was not created.
3. You need to run `make bzImage`, which will result in a smaller kernel file.

Using Shells and Scripts

14

C H A P T E R



EXAM OBJECTIVES

Exam 102 ♦ General Linux, Part 2

1.7 Text editing, Processing, Printing

- **Perform basic file editing operations using vi.** Edit text files using vi. Includes vi navigation, basic modes, inserting, editing and deleting text, finding text, and copying text.

1.9 Shells, Scripting, Programming, Compiling

- **Customize and use the shell environment.** Customize your shell environment: set environment variables (e.g. PATH) at login or when spawning a new shell; write bash functions for frequently used sequences of commands. Involves editing these files in your home directory: `.bash_profile` | `.bash_login` | `.profile` ; `.bashrc` ; `.bash_logout` ; `.inputrc`.
- **Customize or write simple scripts.** Customize existing scripts (like paths in scripts of any language), or write simple new (ba)sh scripts. Besides use of standard sh syntax (loops, tests), be able to do things like: command substitution and testing of command return values, test of file status, and conditional mailing to the superuser. Make sure the correct interpreter is called on the first (`#!`) line, and consider location, ownership, and execution- and suid-rights of the script.

CHAPTER PRE-TEST

1. Which file is executed when a user logs out?
2. How many named buffers does vi have?
3. Which file customizes the readline?
4. Which command tells vi to insert text?
5. How many modes does vi have?
6. Which command is used to make a variable usable by subprocesses?
7. Which variable specifies the prompt when given after an incomplete command?
8. Which command is used to read user input from a shell script?
9. What is the first line of a shell script?
10. Which commands are used to set shell options that change the features of bash?

Most Linux distributions now ship with easy-to-use, point-and-click GUI tools, but they are not always the best to use or the best available. The other way to configure the system is by manually editing the configuration files using a text editor. Several text editors exist. Some of the most popular are vi, emacs, Joe, Pico, and ae. The best editor is often hotly debated on the Internet, and the choice is up to your personal preference. But, the vi editor is usually installed by default with most Linux distributions and other UNIX systems. Given that, it is well worth the time invested to learn it. You never know; you may find that it is your favorite editor after all. Later in the chapter we go over writing shell scripts to help ease the network administrators burden. These scripts can be used to automate tasks and reduce repetitive work. Linux also uses many different scripts to start up and configure the system. These are covered and their functions are explained.

Using vi

Objective

1.7 Text editing, Processing, Printing

- **Perform basic file editing operations using vi.** Edit text files using vi. Includes vi navigation, basic modes, inserting, editing and deleting text, finding text, and copying text.

William Joy originally wrote vi, pronounced *vee eye*, at the University of Berkeley in California. It is a text mode editor with a command set that not everyone is fond of; however, with a little experience that command set becomes second nature. Several clones of vi exist that add functionality, or even a graphical interface, but you can never be sure to find them on a system. So, it is best to learn using the basic vi editor.

Text editing practices

There are a couple of best practices to remember when editing text files that are worth repeating. First, be sure to have a backup copy of any file that you are editing, especially important configuration files. Copy the original to another file for safekeeping, remembering to reset the file permissions as necessary so those stay the same as well.



When editing a file for the first time we usually rename the original to the same name but with a `.dist` extension. This signifies that it is the version that shipped with the distribution. After that, we normally add a date extension.

Save early and save often. Network connections drop, lines are accidentally deleted, or important configuration options are overwritten. Be sure to save your work often so that you do not have to start over in case of such an accident.

Opening files for editing

To have `vi` open a file for editing, use the following command:

```
vi filename
```

If no filename is specified, `vi` opens with a blank slate, and you can specify the filename later when saving. Be sure that you have read and write permissions to the file, or you will not be able to save changes and may find that out after spending a lot of time editing the file. If you know which line needs to be edited in a file, you can jump straight to that line using the following syntax:

```
vi +10 filename
```

This tells `vi` to start editing the file at line 10. Another option for quickly jumping to a point in the file is to specify a search string to start at. This is done with the following syntax:

```
vi +/telnet inetd.conf
```

This would start `vi`, open the `inetd.conf` file, and then jump to the first instance of the string `telnet`.

There are three modes in `vi`. They are as follows:

- ♦ Command mode
- ♦ Insert mode
- ♦ Last line mode

When `vi` starts up, it defaults to command mode. When in command mode, you can move the cursor around, enter commands, delete and append text, or enter insert mode. To return to command mode you press the Esc key. When in insert mode, all you can do is insert text and use the Delete key to remove mistyped text. You cannot move the cursor around the document. You must go back to command mode for that. The last line mode is used to enter complex commands. To enter last line mode you type a colon from command mode. After the command is completed you are returned to command mode.



Caution

Remember that in `vi` the commands are case-sensitive. `J` is different from `j`.

Exiting vi and saving files

Exiting `vi` is done from the last line mode, so the commands start with a colon from command mode.

- ♦ **Saving without closing**—To save the current file, you would use the `:w` command. You should do this often in case of big mistakes or system problems.

- ♦ **Saving and closing**—To exit vi and save the changes to the file, you can either use the `:wq` command or just `ZZ`.
- ♦ **Closing without saving**—If you need to redo the changes you have made to a file, you should exit vi without saving your changes. This is done with the `:q!` command. The exclamation overrides the requirement to save the file before you can exit.



In vi, to save a file without closing, use the `:w` command. To exit and save changes, use `:wq` or `ZZ`. To exit without saving, use `:q!`.

Moving the cursor

vi is a text mode editor, so mouse support isn't available. This is sometimes daunting to new users, but is easily overcome. vi offers numerous shortcuts to quickly move through files, so users are not bogged down in simple cursor movements.

In most cases you can just use the arrow keys to move around the document. But, not all systems have these keys, and sometimes terminal emulation incompatibilities cause them not to function as expected. The cursor movement keys are shown below, diagramming their function:

```

      k
     h l
      j
  
```

The `-` key can be substituted for `k`, and the `+` key for `j`. You should learn these keys even if you plan to always use the arrow keys. Many other applications use the vi cursor movement keys since they are popular and well known. By themselves, the keys move the cursor one line or character. To move more, you can use a number, such as `4j` to move down four lines.

To move around the file more quickly, you can use the full-page and half-page commands. These are `Ctrl+f` and `Ctrl+b` to move forward and backward a page, and `Ctrl+d` and `Ctrl+u` to move forward and backward half of a page.

You can also use line numbers to move around the file. To display the number of the current line you would use the `Ctrl+g` command. To move to a specific line you would use the `nG` command, where `n` is the line number.

There are many other commands to move the cursor around. Most of these are shown in Table 14-1. Remember that you can cause a command to repeat by putting a number in front of it.

Table 14-1
Cursor Movement Commands

Key	Function
h	Moves the cursor one character to the left.
j	Moves the cursor down one line.
k	Moves the cursor up one line.
l	Moves the cursor one character to the right.
Ctrl-G	Displays the current line number.
nG	Goes to the line specified by <i>n</i> .
Ctrl-f	Moves forward one screen.
Ctrl-b	Moves backward one screen.
Ctrl-d	Moves forward one-half screen.
Ctrl-u	Moves backward one-half screen.
w	Moves to the start of the next word.
e	Moves to the end of the next word.
E	Moves to the end of the next word. Unlike the <i>e</i> command, <i>E</i> considers punctuation to be part of the word.
b	Moves to the start of the previous word.
O	Moves to the beginning of the line.
^	Moves to the first word of the current line.
\$	Moves to the end of the line.
Enter	Moves to the next line.
-	Moves to the start of the previous line.
G	Moves to the end of the file.
%	Moves to the matching bracket.
H	Moves to the top line of the screen.
M	Moves to the middle of the screen.
L	Moves to the bottom of the screen.
n	Moves the cursor to the <i>n</i> column.

Adding text

You can add new text in several ways in vi. Each of these requires the user to be in command mode when issuing the command.

Inserting text

To enter insert mode, you would use the `i` command. This causes text to be inserted in to the document wherever the cursor is located. To exit back to command mode, use the Esc key.

Appending text

To append text after current cursor location, use the `a` command. To append text at the end of the current line, use the `A` command.

Creating a new line

Two commands are used to begin a new line of text. The first one, `o`, opens a new line of text below the current cursor location. The second, `O`, creates a new line of text above the current cursor location.

Changing text

vi uses a two-part command to change text. When the change command is used, vi marks the sections to be changed. After the text is marked, you enter the new text to replace the old.

It is started with the `c` command. The second part tells vi how much text to change. To change one word, use `cw`, and to change an entire sentence, use `cs`. vi uses a space to end a word and punctuation to end a sentence. To change the text from the current position to the end of the line, use `c$` or `C`.

Like most other commands, you can use the repeat function. For example, to change three words you would enter `3cw`.

Replacing text

To replace existing text you do not always need to count the number of words or sentences you want to change. The `r` command lets you replace a single character at the current cursor position, and the `R` command lets you replace all text until you return to command mode with Esc.

The full list of commands for adding text is shown in Table 14-2.

Table 14-2
Adding Text Commands

<i>Command</i>	<i>Function</i>
i	Inserts text to the left of the cursor.
I	Inserts text before the first nonspace character in the line.
a	Appends text to the right of the cursor.
A	Appends text at the end of the current line.
o	Begins a new line of text below the current line.
O	Begins a new line of text above the current line.
cw	Changes one word.
cs	Changes the current sentence.
c\$ or C	Changes the current line.
r	Replaces a single character.
R	Replaces text until the Esc key is pressed.
s	Substitutes text for the current character.

Deleting text

As you've seen, vi offers many different ways to accomplish things, and deleting text is no exception.

Deleting single characters

To delete single characters of text you use the `x` and `X` commands. The `x` command deletes the character currently selected by the cursor while the `X` command deletes the character preceding the cursor.

The repeating function also works with these commands. To delete the eight characters before the cursor, use the `8X` command.

Deleting words

The `dw` command deletes the currently selected word. To select a word just put the cursor on the first letter. To delete multiple words to the right of the cursor put a number before or after the `d`, such as `3dw` or `d3w`.

Deleting lines

To delete the line of text where the cursor resides, use the `dd` command. To delete multiple lines add a repeating number, such as `3dd` to delete three lines of text.

Deleting text to the cursor

Two commands in *vi* are used to delete all text before and after the cursor. To delete all text after the cursor to the end of the line, use the `D` command, and to delete all text from the beginning of the line to the cursor, use `d^`. To delete all text from the current cursor position to the end of the screen, use `dL`, and to delete all text to the end of the file, use `dG`. To delete all of the text from the cursor to the beginning of the file, use `d1G`. To delete all text from the cursor to a specified line you would use the `d#` command, where `#` is the number of the specified line.

Table 14-3 summarizes the various delete commands.

Table 14-3
Delete Commands

Command	Function
<code>x</code>	Deletes a single character located at the cursor's location.
<code>X</code>	Deletes a single character located before the cursor.
<code>dw</code>	Deletes a single word. This actually moves the text to an unnamed buffer, which is covered in the next section.
<code>dd</code>	Deletes a single line.
<code>D</code>	Deletes all text after the cursor to the end of the line.
<code>dL</code>	Deletes all text after the cursor to the end of the screen.
<code>dG</code>	Deletes all text from the cursor to the end of the file.
<code>d^</code>	Deletes all text from the beginning of the line to the cursor.
<code>d1G</code>	Deletes all text from the beginning of the file to the cursor.
<code>d#</code>	Deletes all text from the cursor to the specified line number.

Copying and pasting

Like most other modern word processors, *vi* offers the ability to cut and paste text. This can greatly help to speed up repetitive work, when doing things such as shell scripts.

Yanking and pasting text

Yanking and *pasting* are the *vi* terms for copying and pasting. To copy one line of text, use either `yy` or `Y`. Add a number to the front to copy multiple lines, such as `5Y` for five lines. To paste text, you would use the `p` command.

Just using `yy` or `Y` places text in the unnamed buffer. *vi* supports up to 26 named buffers. The unnamed buffer is useful for quick copies, but may be accidentally lost,

and it is sometimes useful to have multiple buffers. To differentiate between a command and a named buffer, the " symbol is used. Named buffers are represented by a single character. When copying text, the lowercase letter is used to replace the current text in the buffer, and the uppercase character is used to append text to the buffer. For example, to move the current line to buffer b, you would type the following:

```
"bY
```

or

```
"byy
```

To copy three lines into the buffer, you would use the following:

```
"by3
```

Or to append the three lines to the current buffer, you would use the following:

```
"By3
```

This is a good example of how commands can be stacked in vi. To copy words instead of lines, use the yw command. To move three words to the buffer C, you would use the following:

```
"Cy3w
```

To paste text from the C buffer, you would use the following:

```
"Cp
```

Moving text

To move text around the document, you use the dd command. By itself, the command places text in the unnamed buffer, but a buffer name can also be specified. To delete text and put it in the a buffer, you would use the following:

```
"add
```

Or to add five lines you would use the following:

```
"a5dd
```

Pasting moved text uses the same commands as when pasting copied text.

Copying or moving between files

The unnamed buffer can be used only within a single file, but named buffers can be used to move text between files. To do this, put text in a named buffer and then

open another file with the `:e` command. To open a file named `list.txt` you would use the following:

```
:e list.txt
```

Once the new document is open you can paste text as you normally would. For example, to move five lines of text from the file `oldlist.txt` to `newlist.txt`, you would use the following sequence:

```
"a5Y
:e newlist.txt
"ap
:w
:e oldlist.txt
```

This copies five lines of text, opens the new file, pastes the buffer, saves the file, and then opens the older file again.

Table 14-4 summarizes the copying and pasting commands.

Table 14-4
Copying and Pasting Commands

Command	Function
<code>yy</code> or <code>Y</code>	Copies a single line of text into the unnamed buffer.
<code>nyy</code> or <code>nY</code>	Copies <i>n</i> number of lines to the unnamed buffer.
<code>yw</code>	Copies a single word to the unnamed buffer.
<code>ynw</code> or <code>nyw</code>	Copies the specified number of words to the unnamed buffer.
<code>y\$</code>	Copies the text from the cursor to the end of the line into the unnamed buffer.
<code>"ayy</code>	Copies a single line of text to the buffer named <i>a</i> .
<code>"Ayy</code>	Appends a single line of text to the buffer named <i>A</i> .
<code>p</code>	Pastes text to the right of the cursor.
<code>P</code>	Pastes text to the left of the cursor.
<code>np</code>	Pastes <i>n</i> copies of text to the right of the cursor.
<code>"ap</code>	Pastes the contents of buffer <i>a</i> to the right of the cursor.
<code>"c3P</code>	Pastes three copies of text in the <i>c</i> buffer to the left of the cursor.
<code>"add</code>	Moves the current line to the buffer named <i>a</i> .
<code>"a5dd</code>	Moves five lines to the <i>a</i> buffer.
<code>dw</code>	Deletes a single word and places it in the unnamed buffer.

Searching for text

Beyond its abilities to manipulate text, vi also offers very powerful mechanisms for searching through text. Unlike most word processors, vi gives you the ability to search with the power of regular expressions, and not just simple phrases.

Pattern matching

Special characters vi uses with searches extend the text editor's capabilities beyond simple word or phrase searches. These characters are known as *metacharacters*.

Matching a single character

To match a single character, use a period. For example, to match *deer* and *dear*, you would use the following:

```
de.r
```

Matching multiple characters

The `*` is used to match multiple characters. It will also match no character. For example, `m*n` would match *mean*, *man*, *moon*, and even *mn*.

Beginning and ending of lines

You can also dictate that a pattern matches only if it is at the beginning or ending of a line. The `^` defines the beginning of a line, and `$` defines the end. Remember that searches are case-sensitive. For example, `^The` would match the first two lines of the following:

```
The Linux kernel is very powerful.  
Then you must restart the daemon.  
the file is corrupt.
```

The pattern `kernel$` would match the first two of the following lines.

```
I recompiled the kernel  
He had a problem when he booted to the new kernel  
I downloaded the new kernel.
```

Escaping a metacharacter

In some cases you may need to search for a character that could be interpreted as a metacharacter, such as a period. To do this, you need to tell vi that you want to treat the character as a regular character, not as a metacharacter. This is called *escaping* the character and is accomplished by using the backslash character before the character you want to escape. For example, to search for the following phrase:

```
kernel. But then,
```

you would use the following pattern:

```
kernel\. But then,
```

Matching a range of characters

It is sometimes useful to match a single character from a given range. This is done by bracketing the possible range of characters. For example, to search for all `v2.x` entries in the file, you would use:

```
v2\.[1-9]
```

Remember you need to escape the period. All numbers between one and nine inclusive would match. The caret character can be used to do an inverse match. For example, the following would match any character that is not a 2.

```
[^2]
```

Table 14-5 lists some of the common metacharacters.

Table 14-5 Common Metacharacters	
<i>Metacharacter</i>	<i>Function</i>
<code>\</code>	Escapes a metacharacter so that the symbol can be searched for.
<code>.</code>	Matches any single character.
<code>*</code>	Matches 0 or more unknown characters.
<code>[]</code>	Matches a single character contained in the brackets.
<code>^</code>	Matches characters at the beginning of a line.
<code>\$</code>	Matches characters at the end of a line.
<code>[^a]</code>	Matches anything that is not a letter <i>a</i> .
<code>[a-b]</code>	Matches anything in the given range of <i>a</i> to <i>b</i> .

Searching text

`vi` provides two methods for searching through text. The first method is very simple and searches only for a single character. To search for the next instance of the character *a*, you would use the command `f a`. To find the first instance of *a* before the cursor you would use the command `F a`.

If you do not want to have the cursor land on the searched character, you can use the `T` and `t` commands. If you search for `t a`, you will land on the character to the left of the next *a*. If you search for `T a` you will land on the character to the right of the first preceding *a*.

To search for more than a single character, you would use the `/` and `?` commands. When in command mode, you would enter `/string` to search for the first instance of *string* to the right of the cursor. To search to the left, use `?`. You can use regular expressions when searching with `/` or `?`.

Repeating a search

To repeat the last simple search command, use the `;` key. To reverse the direction of the first simple search, use the `,` key.

To repeat a `/` or `?` search, enter the symbol again and press enter without any search string. You can switch the symbol used to reverse the direction of the search. A shortcut to repeat the search in the same direction is to use the `n` command, and to reverse the search direction, use the `N` command.

Searching and replacing text

The mechanisms vi offers for performing search and replace functions are powerful. These functions can be done on certain blocks of text or across the entire file. These commands must be run from the colon prompt.

To search and replace through the entire file, begin the command with `:%`, and to search from the current cursor position to the end of the file, use `:$`. To search a range of lines, use `:1,5`, which would designate lines one through five. Lines relative to the current cursor can be done with a command such as `:/+3`, which designates to search the third line from the current position.

The syntax for a search and replace is as follows:

```
:<start>,<finish>s/<find>/<replace>
```

For example, to replace *kernel* with *module* on the first instance of every line, you would use the following:

```
:%s/kernel/module
```

To do the same with every instance of *kernel* on every line you would add a `g` directive, such as the following:

```
:%s/kernel/module/g
```

This does a complete search and replace automatically. To have vi ask for confirmation on every change, use a `c` directive, such as the following:

```
:%s/kernel/modules/gc
```

Table 14-6 lists the search commands.

Table 14-6
Search Commands

Command	Function
<code>fa</code>	Searches for the first instance of <i>a</i> to the right of the cursor.
<code>Fa</code>	Searches for the first instance of <i>a</i> to the left of the cursor.
<code>ta</code>	Moves to the character to the left of the next <i>a</i> .
<code>Ta</code>	Moves to the character to the right of the first preceding <i>a</i> .
<code>;</code>	Repeats the last <code>f</code> , <code>F</code> , <code>t</code> , or <code>T</code> command.
<code>,</code>	Same as <code>;</code> , but reverses the direction of the search.
<code>/string</code>	Searches to the right for the first instance of <i>string</i> .
<code>?string</code>	Searches to the left for the first instance of <i>string</i> .
<code>n</code>	Repeats the last <code>/</code> or <code>?</code> search.
<code>N</code>	Repeats the last <code>/</code> or <code>?</code> search, but in the opposite direction.
<code>:%</code>	Searches the entire file.
<code>:\$</code>	Searches from the current cursor position to the end of the file.
<code>:1,5</code>	Searches lines one through five.

Undoing changes

We all make mistakes, and luckily `vi` lets you undo changes you have made. To undo the last command, use the `u` command. To undo all changes made to the current line use the `U` command. If you have really messed up, you can revert back to the last saved copy of a file with the `:e!` command.

Customizing the Shell Environment

Objective

1.9 Shells, Scripting, Programming, Compiling

- **Customize and use the shell environment.** Customize your shell environment: set environment variables (e.g. `PATH`) at login or when spawning a new shell; write bash functions for frequently used sequences of commands. Involves editing these files in your home directory: `.bash_profile` | `.bash_login` | `.profile` ; `.bashrc` ; `.bash_logout` ; `.inputrc`

The bash shell environment can be customized to suit your needs and tastes. The four methods we cover on how to customize the environment are as follows:

- ♦ Environment variables
- ♦ Aliases
- ♦ Special files
- ♦ Options

Environment variables

The shell variables store information and settings that can be adjusted by the user. The bash shell has several built-in commands, and the user or software can add others. The syntax for assigning a new variable is as follows:

```
variable_name=value
```

Notice that there is no space before or after the equal sign. If the value contains a space it should be enclosed in quotes. For example, to set the variable *NAME*, you would use the following:

```
NAME="Robert Mowlds"
```

Usually variable names are in all caps, but this is not always the case with user-defined variables. When you want to use a variable name in a command or script, you should precede its name with a \$ sign, such as the following:

```
echo $NAME
```

The `unset` command is used to remove a variable. For example:

```
unset NAME
```



The `unset` command is used to remove a variable.

Quoting variables

It is sometimes useful to expand variables inside of another string. Attention must be paid on how to do this, or you will end up just echoing the variable name. To understand this, you need to know how to use quotes in bash.

Consider the following line:

```
debian:~$echo 2 + 3 > 1 + 1
```

The `echo` command is used to display the information or expression given to it. When you enter this command, you get no response, but instead you end up with a

file named 1. To tell `bash` to pass the string literally to the `echo` command, you would use the following:

```
debian:~$echo '2 + 3 > 1 + 1'
2 + 3 > 1 + 1
```

You do not need to quote the entire line, only what you want to be taken literally. For example:

```
debian:~$echo 2 + 3 '>' 1 + 1
```

Double-quotes can also be used in some situations. They bypass some of the translation by the shell, mainly the following:

- ♦ Ignore pipe characters
- ♦ Ignore aliases
- ♦ Ignore tilde substitution
- ♦ Ignore wildcard expansion
- ♦ Do not split words via delimiters, such as spaces

Double quotes allow the following:

- ♦ Parameter substitution
- ♦ Command substitution
- ♦ Arithmetic expression evaluation

Table 14-7 shows some examples of quoting. We'll assume that the command `NAME=Jason` has been run, and `~jason` is `/home/jason`.

Table 14-7
Quoting Examples

<i>Expression</i>	<i>Result</i>
<code>\$NAME</code>	Jason
<code>"\$NAME"</code>	Jason
<code>\\$NAME</code>	<code>\$NAME</code>
<code>'\$NAME'</code>	<code>\$NAME</code>
<code>" '\$NAME' "</code>	<code>'Jason'</code>
<code>"~jason"</code>	<code>~jason</code>
<code>'~jason'</code>	<code>~jason</code>
<code>~jason</code>	<code>/home/jason</code>

Another method of making a special character be taken literally is by using the backslash. For example:

```
debian:~$echo 2 + 3 \> 1 + 1
```

This produces the same output as if you enclosed it in single quotes. To have a literal backslash in a string, either enclose it in single quotes or escape it with `\\`.

The final type of quoting is command substitution. Consider the following command:

```
debian:~# echo `date`
Sun Dec 31 19:39:02 EST 2000
```

When a command is enclosed in ``` symbols, the result of the command is passed back as the input string.



The ``` character is for command substitution. The `'` character passes the string literally. The `"` character literally passes some special characters.

Built-in variables

There are several useful built-in variables used by the shell. The most common of these will be discussed in the following sections.

Editing mode variables

The first group of built-in variables deal with the editing mode of the command line. They control the size and behavior of the command history. The first is `HISTFILE`, which holds the name of the file that the history is saved to. The default value is `~/.bash_history`. The `HISTCMD` variable tells you the history number of the current command. For example:

```
debian:~$echo $HISTCMD
570
```

The `HISTCONTROL` variable is used to filter which commands are added to the history list. The available values for this variable are as follows:

- ♦ `ignorespace`—Do not add commands that start with a space.
- ♦ `ignoredups`—Do not add a command if it is the same as the one that precedes it.
- ♦ `ignoreboth`—Ignores both types of commands.

The `HISTIGNORE` variable is a list of colon-separated patterns that causes matching commands to not be added to the history list. Duplicate commands can be filtered by using the `&` symbol. For example, to block all commands starting with `R` and duplicates, you would use the following:

```
debian:~$HISTIGNORE=R*:\&
```

The backslash must be used to escape the normal use of the `&` command, which would just cause the command to be run in the background.

The `HISTFILESIZE` variable defines the number of entries of the history file. If the size is set to a value smaller than that of the current file, the file is truncated to the new value. The default setting is 500. The `HISTSIZE` variable specifies the size of the command history. Table 14-8 shows these variables.

Table 14-8
History Variables

<i>Variable</i>	<i>Function</i>
HISTFILE	Holds the filename of the history file.
HISTCMD	Holds the history number of the current command.
HISTCONTROL	Defines which commands will not be added to the history list. The valid values are <code>ignorespace</code> , which ignores commands that start with a space; <code>ignoredups</code> , which ignores duplicate commands; and <code>ignoreboth</code> , which ignores both types of commands.
HISTIGNORE	Holds a colon-separated list of patterns that filters commands added to the history list.
HISTFILESIZE	Defines the number of entries that can be added to the history file.
HISTSIZE	Specifies the number of entries that are added to the resident history list.

Mail variables

The mail variables tell the shell where and how often to check for new mail so it can notify you. The shell just checks the defined files and determines if they have been modified since the last check.

The `MAILCHECK` variable controls how often the shell checks for new mail. The default is every 60 seconds. The `MAIL` variable holds the name of the mail file to check. This is usually the user's mail file in the spool directory.

```
debian:~$echo $MAIL
/var/spool/mail/root
```

The `MAIL` variable holds the location of only one mail file to check. Some mail programs use multiple mail files, or you may set up filters or rules to move where mail is stored. The `MAILPATH` variable holds a colon-separated list of filenames to check for new mail. For example:

```
MAILPATH=~/.mail/work:~/mail/personal:~/mail/spam
```

When a new message arrives, the notification message will tell you the file it is in, such as the following:

```
You have new mail in /home/Jason/mail/spam
```

You can also specify a special message to be displayed by enclosing it in quotes. For example:

```
MAILPATH='~/mail/work?"You have new work
mail":~/mail/personal?"You have new personal mail''
```

The `$_` variable can be used to display the referenced mail file in a message. For example:

```
MAILPATH='~/mail/work?"You have new mail in $_''
```

would display

```
You have new mail in /home/jason/mail/work
```

Table 14-9 covers these variables.

Table 14-9
Mail Variables

<i>Variable</i>	<i>Function</i>
MAILCHECK	How often, in seconds, the shell checks for new mail. The default is 60 seconds.
MAIL	The name of the file to check for new mail.
MAILPATH	Colon-separated list of filenames to check for new mail. Overrides the MAIL variable if set.



The MAILPATH variable overrides the MAIL variable.

Prompt variables

The bash prompt that you see at the command line is defined by variables. Table 14-10 lists the prompt variables and their use.

Table 14-10
Prompt Variables

<i>Variable</i>	<i>Function</i>
PS1	This is the primary prompt and is the one you normally see.
PS2	This is the secondary prompt and is used when you type an incomplete line and hit Enter. By default this is set to >.
PS3	This is used for debugging and programming.
PS4	This is used for debugging and programming.

The PS1 variable is the primary prompt string and the one you see most often. In most current distributions it is set by default to show the hostname, user name, and either a \$ or # depending on whether or not you are root. For example, the prompt for a normal user would be as follows:

```
jason@debian:~$
```

The prompt for root would be as follows:

```
debian:~#
```

The PS2 variable is the secondary prompt used for an incomplete command. For example:

```
debian:~$echo "This is an example
> of an incomplete command"
This is an example
of an incomplete command
debian:~$
```

To change the prompt variables you can use the many customization characters supported by bash. Table 14-11 shows these variables. The following are several examples:

```
debian:~$PS1=\d:\h\$
Sun Dec 31:debian$

Sun Dec 31:europa$PS1="\t:\u:\W"
12:39:44:root:/root

12:39:44:root:/rootPS1="\t:\w:\!"
12:40:48:~:513
```

As you can see, you can customize the prompt to suit your needs exactly. Using some strings such as /\$ and /h to make sure you know when you are root and which system you are connected to is a good idea. Such a strategy can keep you from making a mistake.



The main prompt is set with the `PS1` variable.

Table 14-11
Prompt Customization Strings

<i>String</i>	<i>Function</i>
<code>\a</code>	The ASCII bell character. Used to make a beep. Annoy at your own risk.
<code>\d</code>	The date in “Weekday Month Date” format: for example, “Fri May 14”.
<code>\e</code>	An ASCII escape character.
<code>\h</code>	The hostname up to the first period.
<code>\H</code>	The full hostname.
<code>\j</code>	The number of jobs currently being managed by the shell.
<code>\l</code>	The basename of the shell’s terminal device name.
<code>\n</code>	Starts a new line.
<code>\r</code>	Carriage return.
<code>\s</code>	The name of the shell.
<code>\t</code>	The current time in 24-hour format.
<code>\T</code>	The current time in 12-hour format.
<code>\@</code>	The current time in 12-hour a.m./p.m. format.
<code>\u</code>	The current user name.
<code>\v</code>	The version of bash being used.
<code>\V</code>	The release of bash, which is the version and patch level.
<code>\w</code>	The current working directory.
<code>\W</code>	The basename of the current working directory.
<code>\!</code>	The history number of this command.
<code>\#</code>	The command number of this command.
<code>\\$</code>	Puts a <code>#</code> if the effective UID is 0 (root), or a <code>\$</code> otherwise.
<code>\nnn</code>	The character corresponding to octal <i>nnn</i> .
<code>\\</code>	A backslash.
<code>\[</code>	Begins a sequence of nonprintable characters. Used to send terminal control sequences.
<code>\]</code>	Ends the nonprintable sequence.

Search paths

The `PATH` variable is a colon-separated list of directories to be searched when an executable is run. For example:

```
PATH=/sbin:/bin:/usr/sbin:/usr/bin:/usr/bin/X11:/usr/local/sbin
:/usr/local/bin
```

Each of the directories in the list is checked in order when a command is entered at the command prompt.

The `CDPATH` variable is a colon-separated list of directories that are checked when the `cd` command is used. By default this variable is not set. Consider the following setting:

```
CDPATH=~/Documents:~/Graphics
```

If the user types `cd Project`, the shell will first check for a `Project` directory in the current directory, then in `~/Documents`, and then `~/Graphics`.

Exporting variables

Before a subprocess, such as an application or another shell, can use a variable, the variable must be exported. By default, a variable can only be used only in the shell where it was created; this is called the variable's *scope*. Exporting the variable increases its scope. This is done with the following syntax:

```
export variable_name
```

You can also combine commands and export a variable while naming it, such as the following:

```
export variable_name=value
```

An environment variable can also be set up only for a given process, without changing the variable in the current shell. This is done with the following syntax:

```
variable_name=value command
```

To list all current environment variables and their values, use the `export` command with no options or with the `-p` option. To list all shell variables, even those not exported as environment variables, use the `set` command.



Tip

When a variable is declared, it is a shell variable. When it is exported, it becomes an environment variable.

Aliases

bash provides the ability to create an alias for a command. The syntax is as follows:

```
alias new_name=command
```

Aliases with command-line options should be surrounded by quotes, such as in the following:

```
alias home='cd /home/jason'
```

Aliases can also point to other aliases, but in case of a loop, bash knows to execute the loop only one time.



Many Linux distributions have aliases set up to help the user migrating from other operating systems, mainly MS-DOS.



It may seem like a good idea to use an alias for frequently misspelled commands, but it is better to learn the real command. For example, many Linux distributions alias `cd .` to `cd ..`, but you should learn the correct command, because you cannot always count on the alias being set.

Special files

There are several script files that get run at login, when starting a new shell, or when logging out. These script files help set up and configure the shell environment automatically for the user. These files are the following:

- ♦ `.bash_profile`
- ♦ `.bashrc`
- ♦ `.profile`
- ♦ `/etc/profile`
- * `.bash_logout`
- ♦ `.inputrc`
- ♦ `/etc/inputrc`

Login files

When a user logs into the system, the user is given a shell, which is known as a *login shell*. When a user starts another instance of a shell from the command-line or within the X Window system, that is a *nonlogin shell*. This is important because some files are read only in certain circumstances. During a login shell, the shell reads the `/etc/profile` file, if it exists. After that it checks for the following files in order and stops looking after the first one is found.

1. ~/.bash_profile
2. ~/.bash_login
3. ~/.profile



Exam Tip Remember that once one of the three files is read, the others are not checked.

The first file, `.bash_profile`, is read and executed every time you log in to the system. The following is an example of this file.

```
# .bash_profile
# Get the aliases and functions
if [ -f ~/.bashrc ]; then
    . ~/.bashrc
fi
# User specific environment and startup programs
PATH=$PATH:$HOME/bin
BASH_ENV=$HOME/.bashrc
USERNAME=""
export USERNAME BASH_ENV PATH
```

Any permanent changes you want to make to your environment should be added here. If you make a change, it does not take effect until you log in again. To get around this, you can read the file again using the `source` command, which has a shortcut of the period character:

```
source .bash_profile
```

or

```
. .bash_profile
```

The other two files, `.bash_login` and `.profile`, are files derived from other shells. This helps users migrate from one to another without having to move scripts around or learn new startup procedures.

The `.bashrc` file

The `.bashrc` file is read and executed by nonlogin bash sessions. An example of this file is as follows:

```
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files for examples
# If running interactively, then:
if [ "$PS1" ]; then
    # enable color support of ls and also add handy aliases
    eval `dircolors`
```

```

alias ls='ls --color=auto '
#alias ll='ls -l'
#alias la='ls -A'
#alias l='ls -CF'
#alias dir='ls --color=auto --format=vertical'
#alias vdir='ls --color=auto --format=long'
# set a fancy prompt
PS1='\u@\h:\w\$ '
fi

```

As you can see in the previous example of `.bash_profile`, it is common to source the `.bashrc` file so that it is read at login, too. Normally the `.bashrc` holds user-defined aliases and functions.



Many Linux distributions have the `.bash_profile` file source the `.bashrc` file. Aliases are normally stored in `.bashrc`.

The `.bash_logout` file

The `.bash_logout` file is executed when the user logs out. The following is an example:

```

# ~/.bash_logout: executed by bash(1) when login shell exits.
# when leaving the console clear the screen to increase privacy
case "`tty`" in
    /dev/tty[0-9]) clear
esac

```

By default this file usually clears the screen so the next user does not see what the previous user did. Put any cleanup tasks that you want done at logout in this file.

The `.inputrc` file

The `~/.inputrc` file is used to customize the readline. If it is not found, the `/etc/inputrc` file is used. Below is an example of the `/etc/inputrc` file.

```

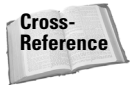
# do not bell on tab-completion
#set bell-style none
set meta-flag on
set input-meta on
set convert-meta off
set output-meta on
"\e0d": backward-word
"\e0c": forward-word
# for linux console
"\e[1~": beginning-of-line
"\e[4~": end-of-line
"\e[5~": beginning-of-history
"\e[6~": end-of-history

```

```

"\e[3~": delete-char
"\e[2~": quoted-insert
# for xterm
"\e0H": beginning-of-line
"\e0F": end-of-line
#for freebsd console
"\e[H": beginning-of-line
"\e[F": end-of-line

```



Chapter 2 covers the Readline Library in detail.

Options

Shell options control many different parameters that define how the shell operates. Some examples of these settings are the style of text editing, file overwrite protection, and variable handling. To enable an option, use the following:

```
set -o optionname
```

To disable an option, use the following:

```
set +o optionname
```

No, that is not a typo. The dash enables an option, and the plus disables an option. Most options are disabled by default. Table 14-12 lists some of the most common options.



The `-o` parameter is used to enable an option, while `+o` is used to disable an option.

Table 14-12
set Options

Option	Function
allexport	Automatically exports all variables created.
braceexpand	The shell does brace expansion. This is on by default.
emacs	Uses the emacs-style mode for command-line editing. This is on by default.
errexit	Exits immediately if a simple command exits with a nonzero status.
hashall	Remembers the location of commands when they are executed to speed future searches. This is on by default.
histexpand	Enables ! style history substitution. This is on by default.

Continued

Table 14-12 (continued)

Option	Function
history	Enables command history.
ignoreeof	Ignores the Ctrl-D command, which normally exits the shell.
noclobber	Does not allow file redirection such as > to overwrite an existing file.
noexec	Reads commands but does not execute them.
noglob	Does not expand wildcards like * and ?.
nounset	Reports an error when an unnamed variable is used. Normally an unnamed variable is reported as having a value of "".
vi	Uses the vi-style command-line editing mode.

Another way to set shell options is with the `shopt` command, which was introduced with bash version 2. Table 14-13 lists the command-line options for `shopt`. Using the `-o` option you can use `shopt` to set those options previously set with the `set` command.

Table 14-13
shopt Command-Line Options

Option	Function
-s	Sets each option.
-u	Unsets each option.
-q	Suppresses normal output. Enables quiet mode.
-o	Allows the values of the option names to be those used by the <code>set</code> command. Used for backward compatibility.
-p	Displays a list of the settable options and their current values.

A list of the most common options is shown in Table 14-14. Most options are disabled by default.

Table 14-14
shopt Options

Option	Function
<code>cdable_vars</code>	If set, an argument to the <code>cd</code> command that is not a valid directory is assumed to be a variable with a value of the directory you want to change to.
<code>cdspell</code>	If set, minor spelling errors when using <code>cd</code> in directory names will be corrected.
<code>checkhash</code>	If set, <code>bash</code> first checks to see if an executable listed in the hash table exists before executing. If it no longer exists, a standard path search is done.
<code>cmdhist</code>	If set, <code>bash</code> attempts to save all lines of a multiple-line command in the same history entry.
<code>dotglob</code>	If set, <code>bash</code> includes filenames beginning with a <code>.</code> in the results of path name expansion.
<code>histappend</code>	If set, the history list is appended to the history file at exit, instead of overwriting it.
<code>mailwarn</code>	If set, if <code>bash</code> does a mail check and notices that the mail file has been accessed since the last check, a warning is issued.

Writing Simple Scripts

Objective

1.9 Shells, Scripting, Programming, Compiling

- **Customize or write simple scripts.** Customize existing scripts (like paths in scripts of any language), or write simple new (ba)sh scripts. Besides use of standard sh syntax (loops, tests), be able to do things like: command substitution and testing of command return values, test of file status, and conditional mailing to the superuser. Make sure the correct interpreter is called on the first (`#!`) line, and consider location, ownership, and execution- and `suid`-rights of the script.

Shell scripts are used to execute several commands in a sequence. They are similar to batch files in the Microsoft world, but provide much more power and control. They are stored as text files and can be as simple as a list of commands to execute or can utilize much more complex functions and structures.

Starting a shell script

Shell scripts are written differently for different shells. This is important to remember if you use the more complex functions of scripting. The scripts in this section are geared toward bash, since it is the default shell for Linux and is the most used.

The first line of a shell script tells the system which interpreter to use. It begins with `#!/`, which denotes the interpreter. An example is the following:

```
#!/bin/bash
```

There is no space anywhere in the line. This line should be found only once in any script. It tells the system that the `/bin/bash` interpreter should be used when executing. If a script is written for another interpreter such as the `ksh` shell, or `perl`, this line will be different. If you are unsure which language a script is written in, it is often useful to just check the first line.



The first line of a shell script tells the system which interpreter to use when the script is run.

Writing a basic script

You can start with a basic script, as shown in the following example. For this example, pretend that you do not know about aliases.

```
#!/bin/bash
# This is a simple script that can be used to display
# my documents in a folder, while pausing the output.
ls -al *.doc | more
```

As you can see, this is a simple script that runs `ls` with basic parameters. To make this script executable, you need to do the following:

```
chmod u+x script_name
```



`chmod` is covered in Chapter 6.

The script can also be run manually without making it executable. For example:

```
bash script_name
```

The `#` signs at the beginning of lines denote comments, except for the first line, which tells the system which interpreter to use. Be sure to use comments often in scripts so that later when editing them again, you can remember what you were doing.

Scripts can take input as well. The following example redoes the script so that you can input any extension to list.

```
#!/bin/bash
# This is a simple script that can be used to display
# any files in long format, and pausing the output.
ls -al *.$1 | more
```

The variable \$1 reads in the first command-line option, \$2 reads the second, \$3 reads the third, and so on. Variables in scripts begin with the \$ symbol. By running the script with a command-line parameter of a file extension, such as doc, it will list all files with that extension.

Testing conditions

Many times in scripts you need to test the condition of a value or file. This is done with the `test` command. The `test` command is used in the next several sections to check whether conditions are met and what actions to take based on them.

Normally, the syntax for `test` is as follows:

```
test -flag file/variable/value
```

Or, to check strings you can use the following:

```
test -flag string = string
test -flag string != string
```

A shortcut is to put the flag and item to be tested in brackets, as in the following:

```
[ -flag file/variable/value ]
```

Table 14-15 lists the available flags for `test`.

Table 14-15
test Flags

Flag	Function
-a <i>file</i>	True if the file exists.
-b <i>file</i>	True if the file exists and is a block device.
-c <i>file</i>	True if the file exists and is a character device.
-d <i>file</i>	True if the file exists and is a directory.
-e <i>file</i>	True if the file exists.
-f <i>file</i>	True if the file exists and is not a special file.

Continued

Table 14-15 (continued)

Flag	Function
<code>-g file</code>	True if the file exists and is set GID.
<code>-h file</code>	True if the file exists and is a symbolic link.
<code>-k file</code>	True if the file exists and has the sticky bit set.
<code>-p file</code>	True if the file exists and is a named pipe.
<code>-r file</code>	True if the file exists and is world readable.
<code>-s file</code>	True if the file exists and has a size greater than zero.
<code>-t file</code>	True if the descriptor exists and refers to a terminal.
<code>-u file</code>	True if the file exists and is set UID.
<code>-w file</code>	True if the file exists and is writeable.
<code>-x file</code>	True if the file exists and is executable.
<code>-0 file</code>	True if the file exists and is owned by the effective user ID.
<code>-G file</code>	True if the file exists and is owned by the effective group ID.
<code>-L file</code>	True if the file exists and is a symbolic link.
<code>-S file</code>	True if the file exists and is a socket.
<code>-N file</code>	True if the file exists and has been modified since the last access.
<code>file1 -nt file2</code>	True if file1 is newer than file2.
<code>file1 -ot file2</code>	True if file1 is older than file2.
<code>file1 -ef file2</code>	True if file1 and file2 have the same device and inode numbers.
<code>-o option</code>	True if the shell option is enabled.
<code>-z string</code>	True if the length of the string is zero.
<code>-n string</code>	True if the length of the string is nonzero.
<code>string1 == string2</code> or <code>string1 = string2</code>	True if the strings are equal.
<code>string1 != string2</code>	True if the strings do not match.
<code>string1 < string2</code>	True if string1 sorts before string2.
<code>string1 > string2</code>	True if string1 sorts after string2.

Flow control

For managing the flow of a script, bash supports several different mechanisms. These include the following:

- ♦ `if/else`
- ♦ `for`
- ♦ `case`
- ♦ `while` and `until`

The `if/else` statement

The `if/else` statement is probably the simplest flow control method to use. The `if/else` command comprises several levels of complexity. In either case the `if/else` statements end with the `fi` directive. This closes the logical block. In the first case, if the condition in the `if` statement is true, the command in the `then` statement is executed. If the condition is false, nothing happens.

```
if condition
then statement(s)
fi
```

In the second case, if the condition in the `if` statement is true, the command in the `then` statement is executed, just as in the first case. If the condition is false, the command in the `else` statement is executed. There is no scenario in which nothing happens.

```
if condition
then statement(s)
else statement(s)
fi
```

In the third case, if the condition in the `if` statement is true, the command in the `then` statement is executed, just as in the first case. If the condition is false, the condition in the `elif` (short for “else if”) statement is checked. If the second condition is true, the `then` statement that follows the `elif` statement is executed. If both the conditions are false, the command in the `else` statement is executed. There is no scenario in which nothing happens. Note that you can have more than one `elif` statement, which gives you as many conditions as you want.

```
if condition
then
    statement(s)
[elif condition
then
    statement(s)]
[else
    statement(s)]
fi
```

An example of an `if/else` statement would be:

```
if [ -a "/etc/passwd" ]
then
    echo "The password file is still there!"
else
    echo "The password file is gone!" | mail root
fi
```

As you can see in table 14-15, the `-a` flag checks to see if the `/etc/passwd` file exists. Therefore, the `if` statement checks to see if the file is there, and if so, it executes the `then` statement which prints out, “The password file is still there!” to the console. If the file does not exist, the `else` statement executes the command to print out “The password file is gone!” to the `mail` command. This causes mail to be sent to root.

The for loop

The `for` loop is used to go through a sequence of values. The syntax is as follows:

```
for item [in list]
do
    statements
done
```

Note that the loop must end with the `done` statement. If the optional `in list` is omitted, the given command-line parameters for the script are used as the list. An example of a `for` loop is the following:

```
#!/bin/bash
IFS=:
for dir in $PATH
do
    echo "Directory: $list"
done
```

The first line in this example sets the Internal Field Separator variable to colon. This causes the script to separate each field at a colon. The second line sets up a `for` loop that goes through one iteration for each directory in the `$PATH` variable. The script then outputs a separate line for each directory found in the `$PATH` variable.

The case statement

The `case` statement lets you test strings against patterns and then execute a function based on the match. The syntax for a case statement is as follows:

```
case string in
    pattern1 )
        statements ;;
    pattern2 )
```

```

        statements ;;
    pattern3 )
        statements ;;
    ...
esac

```

Note that each pattern ends with a right parenthesis and each statement ends with two semicolons. The case statement ends with the `esac` statement (which is *case* spelled backward). Patterns can also be made up of a list of pipe-separated patterns. A pattern of `*` can be used as a default, in case no other patterns are matched.

Here is an example:

```

case $filename in
*.doc )
    echo "This is a document file!" ;;
*.gif|*.jpg|*.png )
    echo "This is a graphic file!" ;;
*.txt
    echo "This is a text file!" ;;
esac

```

In this example, the `case` statement checks the name of the file represented by the variable `$filename`. The first case checks to see if the filename ends in `.doc` and prints a message if it does. The second case checks to see if the file ends in `.gif`, `.jpg`, or `.png`. The pipe character enables the statement to check all three possibilities at once. The third case is similar to the first.

The while and until statements

The `while` and `until` loops run until a condition is met, not for a set number of iterations. The syntax for the `while` loop is as follows:

```

while condition
do
    statement(s)
done

```

An example would be the following:

```

while [ $value -lt 50 ]
do
    $value = (($value + 1))
done

```

The first line of this example specifies that this `while` loop runs as long as `$value` is less than 50. The arithmetic tests are shown in Table 14-16. During each iteration the `$value` variable is incremented by one. Once the `$value` variable hits 50, the loop will stop executing.

The `until` loop is run until the defined condition returns true. The syntax for `until` is as follows:

```
until condition
do
    statements(s)
done
```

An example would be the following:

```
until [ $value -eq 50 ]
do
    value=$((value + 1))
    echo $value
done
```

This example performs a similar function to the previous example. The first line sets up the `until` loop and specifies that the loop runs until `$value` equals 50. Keep in mind how the values are tested. The `until` loop runs as long as the test is false, and the `while` loop runs as long as the test is true.

Table 14-16
Arithmetic Tests

Test	Function
-lt	Less than
-le	Less than or equal to
-eq	Equal to
-ge	Greater than or equal to
-gt	Greater than
-ne	Not equal to

Reading user input

You can prompt a user for input from within a shell script by using the `read` command. The syntax is as follows:

```
read var1 var2 var3...
```

If a user enters more data than you have set variables for, all of the extra data goes to the last variable. The following is an example:

```
debian:~# read first last
lee johnson
debian:~# echo $first
lee
debian:~# echo $last
johnson
```

Script considerations

The security on shell scripts should be set correctly and checked often. If the scripts are stored in a world-readable location there can be serious security implications if the permissions are set incorrectly. If a nontrusted user can change a script file, they can modify its function to access data from any user that runs it. It is best never to set scripts to run SUID, as this can open very large security holes in any system.

Key Point Summary

The Linux environment can be greatly customized to suit your needs and tastes. You can combine the startup script files with the powerful shell scripting abilities, and if a function isn't built in, you can write your own easily.

- ♦ vi is a popular text mode editor that is installed by default on most Linux and UNIX systems.
- ♦ To open a file at line ten, use the syntax `vi +10 filename`.
- ♦ To open a file at the first instance of string, use the syntax `vi +/string filename`.
- ♦ vi has three different modes: command, insert, and last line.
- ♦ To save a file in vi, use the `:w` command.
- ♦ To exit vi and save changes, use either `ZZ` or `:wq`.
- ♦ To close vi without saving changes, use the `:q!` command.
- ♦ The vi cursor movement keys are `k` for up, `j` for down, `h` for left, and `l` for right.
- ♦ Most commands can be repeated by preceding them with a number.
- ♦ To insert text in vi, use the `i` command.
- ♦ To append text in vi, use the `a` command.
- ♦ To create a new line in vi, use either the `o` or `O` commands.
- ♦ To change text in vi, use either the `c` or `C` commands.
- ♦ To replace text in vi, use either the `r` or `R` commands.
- ♦ To delete text in vi, use either the `x` or `X` commands.

- ♦ Many actions can be applied to words by adding a `w` to the end.
- ♦ To delete a line of text in `vi`, use the `dd` command.
- ♦ To copy and paste text in `vi`, you use the `y` and `p` commands.
- ♦ `vi` supports one unnamed buffer and up to 26 named buffers.
- ♦ To search for a single letter of text in `vi`, use the following commands: `f`, `F`, `t`, and `T`.
- ♦ To do a complex search in `vi`, use the `/` command.
- ♦ To repeat a search in `vi`, use the `n` and `N` commands, as well as `/` and `?`.
- ♦ To undo a change in `vi`, use the `u` command.
- ♦ The ``` is used for command substitution, `'` for literal quoting, and `"` to pass some things literally.
- ♦ `bash` provides several built-in variables to control history, editing, mail checking, and prompt customization.
- ♦ The `PATH` variable specifies where `bash` looks for executable files, and `CDPATH` specifies which directories to check when running a `cd` command.
- ♦ A variable must be exported before it can be used by a subprocess.
- ♦ Aliases are used to give commands easier to remember names.
- ♦ During login, `bash` first reads the `/etc/profile` file. After that, it looks for the following files in order and stops at the first one found: `~/.bash_profile`, `~/.bash_login`, and `~/.profile`.
- ♦ The `.bashrc` file is run only with nonlogin `bash` sessions, unless it is called from another script.
- ♦ The `.bash_logout` command is run when the user exits.
- ♦ The `.inputrc` file customizes the readline configuration.
- ♦ Many shell customization options can be changed with the `set` and `shopt` commands.
- ♦ Shell scripts must begin with a line that specifies the command interpreter.
- ♦ Scripts must be set executable before they can be run.
- ♦ Conditions can be tested in scripts with the `test` command.
- ♦ `bash` supports several flow control mechanisms, including `if/else`, `for`, `case`, `while`, and `until`.
- ♦ User input can be accepted with the `read` command.
- ♦ Scripts should not be world-writable and should be stored in a secure directory.



STUDY GUIDE

The following questions and exercises will allow you to review the information covered in this chapter. Take your time when completing this section of the chapter, carefully reviewing any questions that you may have. Answering the question correctly is not as important as understanding the answer, so review any material that you might still be unsure of. Being comfortable with the questions and answers presented here will help you be more prepared for the certification exam questions.

Assessment Questions

1. Which script is executed when a nonlogin bash shell is started?
 - A. `.bashrc`
 - B. `.bash_nologin`
 - C. `/etc/profile`
 - D. `.cshrc`
2. To get input from a user and place it into a variable named `NAME`, you would use the _____ command.
3. Which command inserts text into `vi` at the current cursor location?
 - A. `a`
 - B. `f`
 - C. `T`
 - D. `i`
4. Which option corrects some spelling mistakes when changing directories?
 - A. `cdspell`
 - B. `cdvar`
 - C. `cdspchk`
 - D. `spellcd`
5. To make the shell script named `sortdoc` executable by only the user, you would type _____.

6. Enter the command that is used to delete five lines of text in vi:

7. If NUMBER is set to 4, what will be the result of the following?

```
echo "$NUMBER"
```

- A. \$NUMBER
- B. 'NUMBER'
- C. 4
- D. '4'

8. Which variable tells you the history number of the current command?

- A. HISTORY
- B. HISTCMD
- C. HISTNUM
- D. HISTCTRL

9. To put the date in the prompt, which string would you add?

- A. \D
- B. /d
- C. /D
- D. \d

10. Given the following commands, which directory will the user be taken to, assuming that there is a Jason directory off each one?

```
debian:~# CDPATH=/documents:/shared:/music  
debian:~# cd Jason
```

- A. /documents/Jason
- B. /shared/Jason
- C. /music/Jason
- D. ~/Jason

11. If you want to clean up a directory every time you log out, you would edit the _____ file.

12. Which `set` option will keep you from accidentally overwriting an existing file with file redirection?
- A. `nowrite`
 - B. `noclobber`
 - C. `noklobber`
 - D. `nonuke`
13. Which parameter is used to enable an option with `set`?
- A. `+s`
 - B. `-s`
 - C. `+0`
 - D. `-0`
14. The first line of a script file for the `/bin/bash` shell should be _____.
15. Which `test` parameter returns true if a string is 0 bytes long?
- A. `-z`
 - B. `-n`
 - C. `-s`
 - D. `-0`
16. Which scripting clause matches a string to a pattern?
- A. `case`
 - B. `if/else`
 - C. `for loop`
 - D. `while loop`
17. Which command moves five lines of text into a vi buffer named *a*?
- A. `a5dd`
 - B. `A5dd`
 - C. `"a5dd`
 - D. `"add5`

18. Which metacharacter matches any single character?
- A. .
 - B. *
 - C. ?
 - D. ,
19. To find the next instance of the character *B* after the cursor in *vi*, you would enter _____.
20. Which variable sets the mail check time to be every 10 minutes?
- A. MAIL=10
 - B. MAIL=600
 - C. MAILCHECK=10
 - D. MAILCHECK=600

Scenarios

1. You have been using the standard UNIX `mail` command to read your e-mail, but you have now installed a new client. This new client stores mail in several different places. You notice that you no longer get notified when new mail arrives like you used to. How can you fix this?
2. A user occasionally overwrites files when doing redirection. He or she often forgets the difference between `>` and `>>`. How can you keep this user from deleting his or her files again?
3. Create a simple shell script that will prompt the admin for a user name and then create the user account and change the password.
4. Write a shell script using a `select` statement that takes a filename as input and moves the file to another directory based on its extension. Have it support `.doc`, `.txt`, `.pdf`, and `.ps` files.

Answers to Chapter Questions

Chapter Pre-Test

1. `.bash_logout`
2. 26
3. `.inputrc`

4. `i`
5. Three
6. `export`
7. `PS2`
8. `read`
9. `#!/command/interpreter`
10. `set` and `shopt`

Assessment Questions

1. **A.** Answer B is invalid. The `/etc/profile` is read whenever a user logs in, and they do not have a custom profile file. The `.cshrc` is not used by the bash shell. For more information, see the “Special files” section.
2. **read \$NAME.** The `read` command takes input from a user and places it in to variables. The `.cshrc` is not used by the bash shell. For more information, see the “Reading user input” section.
3. **D.** The `a` command appends text, and can also insert text to the right of the cursor. The other answers are invalid. For more information, see the “Using vi” section.
4. **A.** The `cdspell` shell option can fix some spelling mistakes when changing directories. The other answers are invalid. For more information, see the “Options” section.
5. `chmod u+x sortdoc.` You must set the script as executable before a user can run it. For more information, see the “Writing a basic script” section.
6. `5dd.` For more information, see the “Deleting lines” section.
7. **D.** The double quotes around the variable cause it to be translated to its value. The forward ticks are displayed literally. Remember that back ticks (```) are used for command substitution. For more information, see the “Quoting variables” section.
8. **B.** The `HISTCMD` variable holds the command number for the current command. All of the other answers are invalid. For more information, see the “Built-in variables” section.
9. **D.** The other answers are invalid. For more information, see the “Prompt variables” section.
10. **A.** The `cd` command will go through each directory in the `CDPATH` variable, and change to the first correct one found. For more information, see the “Search paths” section.

11. `.bash_logout`. This file is executed each time a user logs out. For more information, see the “Special files” section.
12. **B.** By enabling this option, you can prevent accidentally overwriting a file when redirecting text. For more information, see the “Options” section.
13. **D.** Remember that `-o` enables an option, while `+o` disables the option. For more information, see the “Options” section.
14. `#!/bin/bash`. This line specifies the command interpreter to use for the script. For more information, see the “Starting a shell script” section.
15. **A.** The `-n` test checks to see if a string is non-zero in length. The `-s` parameter tests to see a file exists and has a size greater than 0. There is no `-0` test. For more information, see the “Testing conditions” section.
16. **A.** The `case` clause matches patterns, while the other clauses test expressions for true or false. For more information, see the “Flow control” section.
17. **C.** The command must start with a `"` symbol, and the buffer name is case sensitive. For more information, see the “Yanking and pasting text” section.
18. **A.** The period matches any single character. The `*` will match multiple characters. The other answers are invalid. For more information, see the “Testing conditions” section.
19. **fB.** For more information see the “Searching text” section.
20. **D.** The `MAILCHECK` variable defines how often to check for new mail in seconds. The `MAIL` variable stores the filename to check for new mail. For more information, see the “Mail variables” section.

Scenarios

1. Set the `MAILPATH` variable with a colon-separated list of the files where mail is stored.
2. You should set the `noclobber` option with `set -o noclobber` in the user’s `.bash_login`.
3. An example would be the following:

```
#!/bin/bash
# Simple script to add users.
#
echo "Enter a userid to be added:"
read NEWUSER
echo "Adding a new user..."
adduser $NEWUSER
echo "Enter a new password when prompted..."
passwd $NEWUSER
```

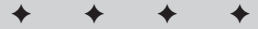
4. An example would be the following:

```
#!/bin/bash
# Simple script to move files based on extensions.
#
case $filename in
*.doc )
    mv $1 ~/doc ;;
*.pdf )
    mv $1 ~/pdf ;;
*.txt )
    mv $1 ~/txt ;;
*.ps )
    mv $1 ~/ps ;;
esac
```


Managing the Network

The final part of the book covers managing the network. In this section you are introduced to networking on a Linux system and provided with detailed information on the various networking servers and their configuration. Chapter 15 provides an introduction to networking including configuration of network adapters and basic tools used on a network. Chapter 16 then expands on this by covering the various network services include Web, mail, FTP, and DNS services. Chapter 17 concludes this part with information on securing a networked Linux system.

IV

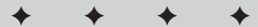


In This Part

Chapter 15
Networking
Fundamentals

Chapter 16
Managing Network
Services

Chapter 17
Managing Security



Networking Fundamentals

EXAM OBJECTIVES

Exam 102 ♦ General Linux, Part 2

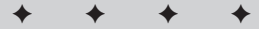
1.12 Networking Fundamentals

- **Fundamentals of TCP/IP.** Demonstrate an understanding of network masks and what they mean (i.e. determine a network address for a host based on its subnet mask), understand basic TCP/IP protocols (TCP, UDP, ICMP) and also PPP, demonstrate an understanding of the purpose and use of the more common ports found in `/etc/services` (20, 21, 23, 25, 53, 80, 110, 119, 139, 143, 161), demonstrate a correct understanding of the function and application of a default route. Execute basic TCP/IP tasks: FTP, anonymous FTP, telnet, host, ping, dig, traceroute, whois.

Continued

15

CHAPTER



EXAM OBJECTIVES (CONTINUED)

- **TCP/IP Troubleshooting & Configuration.** Demonstrate an understanding of the techniques required to list, configure and verify the operational status of network interfaces, change, view or configure the routing table, check the existing route table, correct an improperly set default route, manually add/start/stop/restart/delete/reconfigure network interfaces, and configure Linux as a DHCP client and a TCP/IP host and debug associated problems. May involve reviewing or configuring the following files or directories: `/etc/HOSTNAME` | `/etc/hostname`, `/etc/hosts`, `/etc/networks`, `/etc/host.conf`, `/etc/resolv.conf`, and other network configuration files for your distribution. May involve the use of the following commands and programs: `dhcpcd`, `host`, `hostname` (`domainname`, `dnsdomainname`), `ifconfig`, `netstat`, `ping`, `route`, `traceroute`, the network scripts run during system initialization.
- **Configure and use PPP.** Define the chat sequence to connect (given a login example), setup commands to be run automatically when a PPP connection is made, initiate or terminate a PPP connection, initiate or terminate an ISDN connection, set PPP to automatically reconnect if disconnected.

CHAPTER PRE-TEST

- 1.** What is each host on the Internet uniquely known by?
- 2.** Which protocol in the TCP/IP suite handles error reporting?
- 3.** What service uses port 139?
- 4.** Which application provides terminal emulation?
- 5.** Which file is used to manually map names to IP addresses?
- 6.** Which tool allows you to view the path a packet takes through a network?
- 7.** What method is used to automatically configure a workstation's IP information?
- 8.** Which tool is used to assign an IP address to a network card?
- 9.** What type of file is used to supply the user name and password when making a PPP connection?
- 10.** Which option is used to automatically initiate the reconnection of a PPP connection that has been terminated?

Networking is a fundamental part of Linux, as it is with most other UNIX-derived operating systems. This is the major reason that Linux works so well as an Internet server. This chapter covers how to configure and troubleshoot the networking features in Linux. Special attention will be paid to the sometimes overwhelming number of configuration files and the excellent tools available to track down problems. Finally, we cover how to get a Linux system on the Internet via a PPP connection.

The TCP/IP Protocol Suite

Objective

1.12 Networking Fundamentals

- **Fundamentals of TCP/IP.** Demonstrate an understanding of network masks and what they mean (i.e. determine a network address for a host based on its subnet mask), understand basic TCP/IP protocols (TCP, UDP, ICMP) and also PPP, demonstrate an understanding of the purpose and use of the more common ports found in `/etc/services` (20, 21, 23, 25, 53, 80, 110, 119, 139, 143, 161), demonstrate a correct understanding of the function and application of a default route. Execute basic TCP/IP tasks: FTP, anonymous FTP, telnet, host, ping, dig, traceroute, whois.

UNIX and TCP/IP have grown up and evolved together over many years. Where once the world was filled with a network protocol for every network vendor, it is now settling on one large suite, TCP/IP. TCP/IP survives because of how well it adapts and because of its scalability, which lets it work on both tiny networks and the enormous Internet.

Addresses

Every device on a TCP/IP network is known as a *host*. Each host must have a way to be uniquely identified, and that way is by IP address. Computers are connected to the network using network interface adapters. A computer can have multiple network adapters, and each one connected to the network must have at least one IP address.

IP addresses are 32-bit numbers that are usually written as four octets (8 bits) separated by periods. For example, your server may have the IP address of 192.168.100.100. It could also be known by its 32-bit number of 3232261220, and in fact, some TCP/IP applications can take input in this form (`http://3232261220`). To see how these numbers relate, you need to change the IP address to binary:

192 = 1100 0000

168 = 1010 1000

100 = 0110 0100

100 = 011 00100

If you take all four binary octets together you get 11000000101010000110010001100100, which is 3232261220 when expressed as a 32-bit number. As you will soon learn, it is sometimes very important to consider IP addresses in terms of binary digits.

Network classes

When TCP/IP addresses are handed out to organizations, they are not provided in sequential order. The addresses are assigned through Internet service providers, and the ISPs get them from `www.arin.net`. The addresses are segmented up into what are known as network classes. The five classes are as follows:

- ♦ **Class A:** 1.0.0.0–126.0.0.0
- ♦ **Class B:** 128.0.0.0–191.0.0.0
- ♦ **Class C:** 192.0.0.0–223.0.0.0
- ♦ **Class D:** 224.0.0.0–239.0.0.0
- ♦ **Class E:** 240.0.0.0–254.0.0.0

Addresses used by computers on the Internet are typically class A, class B, and class C addresses. Class D addresses are used for multicasting, and class E addresses are for experimental purposes. The determination of IP address class may seem arbitrary, but it has a very specific basis. To better understand this class assignment you need to convert these decimal numbers to binary. Table 15-1 contains examples of binary-to-decimal conversion of IP addresses.

Table 15-1
Binary-to-Decimal Conversion

<i>Decimal Address</i>	<i>Class</i>	<i>Binary Address</i>
10.254.90.135	A	0000 1010.1111 1110.0101 1010.1000 0111
192.168.1.3	C	1100 0000.1010 1000.0000 0001.0000 0011
172.16.116.98	B	1010 1100.0001 0000.0111 0100.0110 0010
4.2.2.1	A	0000 0100.0000 0010.0000 0010.0000 0001
164.109.153.102	B	1010 0100.0110 1101.1001 1001.0110 0110
216.254.90.131	C	1101 1000.1111 1110.0101 1010.1000 0011

By examining these addresses you can discover the reasoning behind the address classifications. Class A addresses have a leading bit of 0 (zero), class B addresses start with 10, and class C addresses begin with 110. These bits are used for the network address leaving the rest of the bits as host addresses. Class A addresses

allow for a greater number of hosts than a Class B address, while Class B addresses allow for a greater number of hosts than Class C addresses.

These ranges of addresses include some reserved addresses as well. These addresses are not valid for use on the Internet. The address 127.0.0.1 is known as the *loopback* address. This address is assigned to the local network interface. This address can be useful when troubleshooting network problems on a local system.

Also, each class of IP addresses includes a range of addresses specifically for use on private networks. These addresses are not valid on the Internet but are useful for networks not connected to the Internet. Those addresses are as follows:

- ♦ **Class A:** 10.0.0.0–10.255.255.255
- ♦ **Class B:** 172.16.0.0–172.31.255.255
- ♦ **Class C:** 192.168.0.0–192.168.255.255

Dividing networks with subnet masks

The network address includes two important pieces of information. It contains both the network on which a machine is located and the unique identifier for that network interface. A *subnet mask* (also called a *netmask*) is used to separate these two pieces of information. When working with subnet masks, it is helpful to convert decimal numbers to binary. By examining the address from the previous table 10.254.90.135 and the subnet mask of 255.255.255.224, you can determine the network ID and the host ID. This process is known as *anding*.

First, you convert both the IP address and the subnet mask to binary form.

```
0000 1010.1111 1110.0101 1010.1000 0111 — IP address
1111 1111.1111 1111.1111 1111.1110 0000 — subnet mask
```

Notice that when the subnet mask is converted to binary, the first 27 bits are ones, and the last 5 bits are zeroes. This means that the first 27 bits of the IP address are used to indicate the network ID, and the last 5 bits are used to indicate the host ID.

To and these two, you simply examine the two side-by-side. All positions that contain a one in both addresses are anded to a 1. All others become zero. So, the result of anding these two numbers is the following:

```
0000 1010.1111 1110.0101 1010.1000 0000 — network address
```

After converting these numbers back to decimal, you have the following:

```
10.254.90.135 — IP address
255.255.255.224 — subnet mask
```

10.254.90.128 — network ID

0.0.0.7 — host ID

Another way to express this is 10.254.90.135/27 because the first 27 bits of this address are used to specify the network ID.

Table 15-2 shows the default subnet masks used with each address class, as well as the number of hosts per network supported. It is important to remember that routers separate networks. This means that networks on different sides of a router are on different subnets and must have different network addresses. Many physical limitations can affect the size of a network, and because of these limitations, many of the available IP addresses would be wasted without subnetting. Subnetting enables you to segment network addresses so that many different network segments can be supported. When determining the correct subnet mask to use it is important to consider the number of network segments needed along with the number of hosts per network. As demonstrated in Table 15-2, as you increase the number of network segments, the number of hosts per network is decreased.

Table 15-2
Address Classes, Subnet Masks, and Hosts

<i>Address Class</i>	<i>Number of Network ID Bits</i>	<i>Default Subnet Mask</i>	<i>Number of Hosts per Network</i>
Class A	8	255.0.0.0	16,777,214
Class B	16	255.255.0.0	65,534
Class C	24	255.255.255.0	254

Keep in mind a few important considerations for network and host IDs. Neither can consist of all 0s or 1s. A network or host ID of all 1s represents the broadcast address, and an ID of all 0s indicates a local network. These addresses are not routed and will not function as valid network addresses. This is why only 254 hosts are possible with a class C network address.

You can use a simple formula when attempting to determine the correct subnet mask to use when segmenting a network. The formula used to determine the number of possible IDs is as follows: $2^n - 2 = \text{available IDs}$, where n is the number of bits used. This formula can be used for the number of network bits or the number of host bits. One bit has two possible positions, 1 or 0. Two bits have four possible positions: 00, 01, 10, and 11. Three bits have eight possible positions: 000, 001, 010, 011, 100, 101, 110, and 111. Because IDs of all 1s and 0s aren't valid, you need to subtract two from the number of possible bit combinations.

If you know the number of IDs that you need and are trying to determine the number of bits required, you can simply increase the number of IDs needed by two and then determine how many bits would be required to allow that number of combinations.

An example of how to subnet a network is as follows:

You have decided to use the reserved class C network address of 192.168.1.0 for your local network, which will not be connected to the Internet. Your company will have offices on four of the floors in the building. A router will separate each floor. Because of this, you will need four different network IDs for use on the network. This means that you will require six possible bit combinations. Two bits allow for only four possible combinations, but three bits allows for eight. Therefore, you will require three bits to specify the network ID for all of the subnets on your network. This will create a subnet mask of 255.255.255.224 and allow for six bits to be used for host IDs. Using the above formula above you can determine the number of hosts allowed on each subnet.

$$2^6 - 2 = 62$$

The possible network IDs for the subnets on this network are as follows:

- ♦ 192.168.1.32/27
- ♦ 192.168.1.64/27
- ♦ 192.168.1.128/27
- ♦ 192.168.1.160/27
- ♦ 192.168.1.192/27

It is also possible to determine the subnet mask based on the number of host IDs required. Taking the same network address used in the previous example (192.168.1.0) and the requirement of eight hosts per segment, you can determine the correct subnet mask. As you can see by examining bits, three bits allows for six hosts per segment, and four bits allows for 14 hosts per segment. This is also calculated as $2^4 - 2 = 14$. Therefore, the subnet mask used to allow for eight hosts per segment is 255.255.255.240. Using this subnet mask, the following subnets are possible:

- ♦ 192.168.1.16/28
- ♦ 192.168.1.32/28
- ♦ 192.168.1.48/28
- ♦ 192.168.1.64/28
- ♦ 192.168.1.80/28
- ♦ 192.168.1.96/28
- ♦ 192.168.1.112/28

- ♦ 192.168.1.128/28
- ♦ 192.168.1.144/28
- ♦ 192.168.1.160/28
- ♦ 192.168.1.176/28
- ♦ 192.168.1.192/28
- ♦ 192.168.1.208/28
- ♦ 192.168.1.224/28

This allows for 14 network segments that each support up to 14 network interfaces. These interfaces must include all routers, print devices, and network adapters on the segment. Some machines may contain more than one network adapter, and each must have a unique host ID. By carefully examining the needs of your network and the potential growth, it is possible to determine the correct subnetting scheme that will work for you now and in the future.

Protocols

TCP/IP is actually a suite of protocols including the TCP and IP. There are many different protocols in this suite that are used for specific network communication needs. The primary protocols used in the TCP/IP suite are IP (Internet Protocol), ARP (Address Resolution Protocol), ICMP (Internet Control Message Protocol), TCP (Transmission Control Protocol), and UDP (User Datagram Protocol). These protocols work together transmitting message units across the network. The TCP/IP suite of protocols is scalable, allowing them to function well whether on a small, home network or on the Internet.

All of these protocols work together to provide transmission of data across the network. They each manage data differently and when combined provide end-to-end data delivery for all situations.

Internet Protocol

The Internet Protocol (IP) handles the delivery of datagrams across the network; this includes the packet fragmentation and routing. IP is an unreliable protocol because it does not establish a direct connection between the sending and receiving hosts. Instead the packets are sent on the assumption that they will reach the intended recipient. This procedure can result in datagrams being received out of order, duplicated, delayed, and even lost. Other protocols are used to ensure reliable delivery of data, so IP doesn't duplicate the work of these protocols. IP is responsible for delivering datagrams to hosts. When the host is located on a remote network, IP makes use of routers to get the datagram to the receiving network and host. IP uses *IP headers* to label the datagram with information about the data contained in the datagram. These headers also contain information about the sender, recipient, and protocol for which the datagram is intended. This allows the datagram to be transmitted to the correct recipient and allows for the correct protocol to handle the datagram once it has been delivered.

Address Resolution Protocol

TCP/IP addresses are easy-to-remember addresses assigned by the user or systems administrator to a system. They provide unique identifiers for each network interface. Each network interface also has a physical hardware address that has been assigned by the manufacturer. This address is unique to all other network interfaces that exist. However, these addresses are very long and difficult to remember, and they aren't *routable*, meaning that the sending interface can locate the receiving interface only if they're on the same subnet. If there's a router in between the sender and the receiver, the transmission will be lost. So, when TCP/IP addresses are assigned to a network interface, the system allows that TCP/IP address to represent the underlying physical address because TCP/IP is a routed protocol. This makes it easy for users and applications to communicate using the TCP/IP address. However, during the physical transmission of data across the network media, this TCP/IP address must be resolved to a matching hardware address. This is where ARP (Address Resolution Protocol) comes in. ARP uses broadcasts on the local network to find the matching physical address for each TCP/IP address that is sending or receiving data. An ARP cache is maintained on each system for a period of time, which allows for a reduction in the number of broadcasts that must occur on the network.



The unique address assigned to a network card, known as a *MAC address* for media access control, can be changed via software on many adapters.

For example, suppose, the router connecting your network to the Internet is sending requests to the Apache Web server located on your network. The ARP cache on the router is first checked to see whether the cache contains a matching hardware address for the TCP/IP address of the server. If no matching entry is found, the router will then send an ARP request as a network broadcast message. This request includes the IP and hardware address of the router, along with the IP address of the server. This message is examined and then ignored by all other network interfaces on the network segment except for the Web server. The server with the matching IP address updates its local ARP cache with the information for the router. Then the server sends an ARP reply to the router. The router then updates its ARP cache with the IP address and hardware address of the server.

To prevent excessive network traffic, broadcast messages are not routed between network segments. This means that ARP cannot provide matching of IP address and hardware addresses to systems beyond the local subnet. However, this isn't necessary. If the machine is not located on the local subnet, the data is sent to the router on the local subnet. Therefore, ARP is used to discover the mapping of the IP address to the hardware address of the router.

Internet Control Message Protocol

The Internet Control Message Protocol (ICMP) can be used to report errors in delivery of datagrams. It does not provide reliability to IP but instead can report on some errors that are a result of a failure in an IP action. ICMP messages include the following: echo request, echo reply, redirect, source quench, and destination unreachable. Source quench is used when a sender is transmitting data faster than the receiver can handle. A source quench signals the sender to slow the transmission rate to the receiver.

This protocol can be useful for troubleshooting TCP/IP connection problems. The utilities `ping` and `tracert` use this protocol. Use of these utilities is discussed later in this chapter.

Transmission Control Protocol

The Transmission Control Protocol (TCP) provides reliability that is missing in the Internet Protocol. TCP provides *flow control* and *package sequencing* to ensure reliable transfer of data. When a TCP transmission occurs, a session between the communicating hosts is established. This session allows for error recovery and flow control as an acknowledgement from the receiving system is required for each segment transmitted. Segments are numbered before they are sent, so any missing segment can be detected and resent. When the transmission of data has completed and all acknowledgements are received, the session is closed.

This process provides for reliable transmission of data, but has a cost. An overhead is associated with TCP, caused by establishing the session, sending the acknowledgements, and numbering the segments. However, for important transmissions across the network, the reliability provided by TCP can be essential.

User Datagram Protocol

User Datagram Protocol (UDP) also provides datagram delivery. However, unlike TCP, this protocol provides unreliable transmissions. UDP does not perform error checking; applications that use UDP often use other methods of error checking to ensure the integrity of the data. UDP uses connectionless delivery, with no direct connection between communicating systems, to provide fast transmission of small messages, without the overhead associated with connection-oriented delivery methods. Because broadcast messages are not sent to a specific host, they must use the connectionless delivery provided by UDP.

Ports

TCP and UDP use *ports*, similar to apartment mailboxes that receive packets for a specific application, for network communications. This strategy allows many simultaneous UDP and TCP transmissions to occur as each transmission is directed to a specific port. These port numbers are predefined by networking standards and

correspond to specific services. Some available port numbers can be defined by the user. Port numbers are combined with the source and destination addresses of the data to create a *socket*. Each network interface may have multiple open sockets at a given time that are both transmitting and receiving data. These sockets organize all of these transmissions and prevent them from being scrambled.

Because the port numbers for commonly used services are predefined, they are stored on each computer. On Linux systems the port assignments are located in the `/etc/services` file, which is discussed later in this chapter. Table 15-3 lists the most commonly used applications and their default port assignments.



It is important to memorize the standard list of port assignments for the exam.

Table 15-3
Port Assignments

<i>Application</i>	<i>Port</i>
FTP-DATA	20
FTP	21
SSH	22
Telnet	23
SMTP	25
DNS	53
finger	79
HTTP	80
WWW	80
POP2	109
POP3	110
NNTP	119
NETBIOS-SSN (session service)	139
IMAP	143
SNMP	161
IRC	194
HTTPS	443

Note that in Table 15-3 FTP-DATA is assigned to port 20 while FTP is port 21. When an FTP command is sent, port 21 is used. However, when sending data, FTP uses port 20.

Applications

The ports listed in the previous section are all assigned to services used in the TCP/IP suite. These services that are provided on the network include mail, remote access, file transfer, and Web browsing. Along with the services are some applications that can help reveal information about network or user status. Some of the most commonly used applications are the following: `ftp`, `telnet`, `host`, `ping`, `traceroute`, `whois`, and `finger`. Each of these applications is discussed in more detail in the sections that follow.

FTP

The File Transfer Protocol (FTP) service and application does what the name implies. It is used to transfer files across the network. FTP requires both an FTP server and an FTP client for file transfers. The FTP server requires a login to grant access to files, which can be through a user account on the server or through an anonymous account. The anonymous account allows users to access files without having an account on the server. This can also be a security measure for the user, because the user IDs and passwords sent to an FTP server are not secured. Therefore, to prevent an intruder from gaining access to a valid user account on the system, which can be used to cause damage or compromise the system, servers are often configured for anonymous access. Users logging into an anonymous FTP server use the user name of *anonymous* and the password of their e-mail address. Any time remote access to files on the system is granted, carefully examining the security and file permissions of the server is important. If these permissions aren't set correctly, key system files may be altered causing problems for all users. When you are working in an environment with a firewall that prevents direct connections to remote machines, the `ftp` command can be used with the `-p` option, which specifies passive mode.



Tip

FTP makes a connection to the remote server on port 21, but when a file is transferred the remote server makes a connection back to the client on port 20. If a firewall blocks inbound connections, you can cause the client to initiate the data connection by enabling passive mode.

To initiate an FTP session to `ftp.redhat.com` the following command would be used:

```
# ftp ftp.redhat.com
```

An example of an FTP session is displayed below. As you can see, information such as IP address, the specific FTP server version, and file transfer mode are displayed. When downloading files via FTP, pay careful attention to the transfer mode. Binary

files, such as executables and compressed files, must be downloaded using binary mode. If these files are obtained using ASCII mode, the default on some systems, they will be unusable. Once you have reached an FTP prompt, shown in the last line of the following session, you can use the commands in Table 15-4.

```
# ftp deedee.the-nashes.net
Connected to 10.254.90.135.
220 deedee.the-nashes.net FTP server (Version wu-2.6.0(1) Mon
Feb 28 10:30:36 EST 2000) ready.
Name (10.254.90.135:angie):
331 Password required for angie.
Password:
230 User angie logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

Table 15-4
FTP Commands

FTP Command	Use
<code>cd</code>	Changes directory on the remote machine.
<code>lcd</code>	Changes directory on the local machine.
<code>ascii</code>	Switches to ASCII file transfer mode.
<code>binary</code>	Switches to binary file transfer mode.
<code>bye</code>	Exits the FTP session.
<code>ls</code>	Prints a listing of the directory contents.
<code>get file</code>	Downloads the specified file from the remote machine to the local machine.
<code>mget file</code>	Downloads multiple matching files; can be used with wildcards such as <code>*</code> .
<code>put file</code>	Uploads the specified file from the local machine to the remote machine.
<code>mput file</code>	Uploads multiple matching files; can be used with wildcards such as <code>*</code> .
<code>open host</code>	Establishes an FTP connection to the specified host.
<code>prompt</code>	Toggles interactive prompting.
<code>pwd</code>	Displays the present working directory on the remote system.
<code>verbose</code>	Toggles verbose mode.

Telnet

The Telnet service and application allows remote access to a machine. Using Telnet, you can access the shell on a remote system. Telnet requires the user to log in with a valid user name and password, as though they were logging on the machine locally. All of the user's configuration files are used, and the user has access to all of the files and programs that can normally be accessed through the shell. By default the root or superuser account cannot log in via Telnet. If a user needs to access the system as root via Telnet, he or she must first log in using a standard login ID and then use the `su` command to access the root account. This procedure is done because the user name and password sent to the remote computer aren't secure. As with FTP, the Telnet program could allow unauthorized users to access the system if they locate a valid user name and password. For this reason, use caution when allowing remote users to Telnet to a machine. Another, more secure alternative is SSH, which provides a more secure remote shell. The following is an example of the output produced by the Telnet utility:

```
# telnet deedee.the-nashes.net
Trying 10.254.90.135...
Connected to 10.254.90.135.
Escape character is '^]'.

Red Hat Linux release 6.2 (Zoot)
Kernel 2.2.14-5.0 on an i686
login:
```

host

The `host` utility is used to look up hostnames on domain name servers. Information such as IP address, hostname, and mail servers can be discovered using this command. The following are examples of output produced by the `host` command:

```
# host www.the-nashes.net
www.the-nashes.net has address 216.254.90.131

# host 216.254.90.131
216.254.90.131.in-addr.arpa domain name pointer brain.the-nashes.net

# host the-nashes.net
the-nashes.net has address 10.254.90.131
the-nashes.net has address 216.254.90.131
the-nashes.net mail is handled (pri=10) by mail.the-nashes.net
```



The `host` command is covered in more detail in Chapter 16.

ping

The `ping` utility is used to verify TCP/IP connectivity. This utility uses the ICMP protocol to send small packets of data to the specified host. The host then sends a response packet back to the local host. This is a useful utility for verifying network

connectivity between two hosts. The ping utility accepts either the IP address or the hostname of the target machine. Table 15-5 shows the common ping command-line options. The following is an example of results produced by the ping command:

```
# ping -c 4 bugs.the-nashes.net
PING bugs.the-nashes.net (216.254.90.129) from 10.254.90.153 :
56(84) bytes of data.
64 bytes from bugs.the-nashes.net (216.254.90.129): icmp_seq=0
ttl=240 time=97.4 ms
64 bytes from bugs.the-nashes.net (216.254.90.129): icmp_seq=1
ttl=240 time=95.4 ms
64 bytes from bugs.the-nashes.net (216.254.90.129): icmp_seq=2
ttl=240 time=98.4 ms
64 bytes from bugs.the-nashes.net (216.254.90.129): icmp_seq=3
ttl=240 time=99.0 ms

--- bugs.the-nashes.net ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 95.4/97.5/99.0 ms
```



Pay close attention to the parameters for the networking tools, as you often encounter them on the exam. In other words, do not just use the tools in default form or you may be surprised.

Table 15-5
ping Options

Option	Function
-c <i>NUMBER</i>	Specifies number of packets to send and receive.
-d	Enables debug option.
-f	Sends packets as fast as possible. Use this option with caution as it can cause data loss.
-i	Waits a set number of seconds between each packet.
-n	Displays addresses as IP only, not resolved to DNS names.
-p <i>pad</i>	Specifies pad bytes to test for pattern problems.
-q	Displays only summary lines.
-s <i>size</i>	Specifies the packet size.
-v	Verbose output. Displays responses other than ECHO_RESPONSE.

traceroute

The `traceroute` utility also verifies TCP/IP connectivity, but provides more detail than `ping`. The `traceroute` utility also uses the ICMP and UDP discussed earlier in the chapter. As you can see in the output that follows, detailed information concerning every “hop” between two locations on the network is displayed. This includes information such as hostname, IP address, and the time between hops. The ability to view the route of data between two hosts can be helpful for locating any interruption in connectivity. Table 15-6 shows the most common `traceroute` options. The following is an example of use of the `traceroute` command:

```
# traceroute bugs.the-nashes.net
1  10.254.90.140 (10.254.90.140)  1.590 ms  1.787 ms  1.504 ms
2  10.88.250.1 (10.88.250.1)  12.069 ms  21.029 ms  13.017 ms
3  24.25.1.61 (24.25.1.61)  11.369 ms  14.079 ms  13.044 ms
4  24.25.1.49 (24.25.1.49)  11.187 ms  10.865 ms  11.797 ms
5  24.93.64.97 (24.93.64.97)  16.053 ms  18.973 ms  11.807 ms
6  sgarden-roc-gsr.carolina.rr.com (24.93.64.18)  14.299 ms  14.663 ms  14.019
ms
7  sa-gsr-sgarden-gsr.carolina.rr.com (24.93.64.29)  16.924 ms  16.793 ms
24.107 ms
8  ip137.203.54.216.in-addr.arpa (216.54.203.137)  38.879 ms  34.015 ms  31.446
ms
9  207.67.50.5 (207.67.50.5)  35.844 ms  43.260 ms  32.573 ms
10 207.67.50.126 (207.67.50.126)  35.213 ms  33.769 ms  33.335 ms
11 sl-gw21-pen-6-0-0.sprintlink.net (144.228.178.17)  52.788 ms  54.032 ms
53.209 ms
12 sl-bb13-pen-2-2.sprintlink.net (144.232.5.133)  50.938 ms  51.764 ms  51.741
ms
13 sl-bb21-pen-13-0.sprintlink.net (144.232.5.233)  61.732 ms  53.964 ms
51.882 ms
14 sl-bb21-nyc-13-0.sprintlink.net (144.232.18.66)  51.441 ms  52.157 ms
52.384 ms
15 sl-gw9-nyc-9-0.sprintlink.net (144.232.7.98)  50.764 ms  50.827 ms  51.524
ms
16 sl-internapnyc-12-0-0.sprintlink.net (144.232.173.34)  51.722 ms  51.849 ms
54.715 ms
17 border10.fe2-0-fenet2.nyc.pnap.net (209.191.128.142)  52.111 ms  50.979 ms
53.034 ms
18 spk-1-nyc.dsl-isp.net (209.191.175.193)  58.072 ms  65.246 ms  68.805 ms
bugs (216.254.90.129)  92.255 ms  95.289 ms  90.429 ms
```

Table 15-6
traceroute Options

<i>Option</i>	<i>Function</i>
-f	Sets the initial time-to-live value.
-F	Sets the don't fragment bit on the packet.
-I	Uses ICMP echo instead of UDP.
-m	Sets the maximum time-to-live on the outgoing packet. The default is 30.
-n	Displays addresses as IP only. Does not resolve the names.
-v	Verbose mode. Displays responses other than TIME_EXCEEDED and UNREACHABLE.
-w	Sets the time to wait for a reply in seconds. The default is 5.

whois

The `whois` utility searches the InterNIC database for a matching domain. The InterNIC database contains the DNS entries for domains used on the Internet. Detailed information regarding the name servers, contact information, and the registrar is displayed. This information is gathered by querying a `whois` server and can be useful when investigating questions regarding a specific domain. The following is an example of the output from a `whois` utility search.

```
# whois the-nashes.net
Whois Server Version 1.3
```

```
Domain names in the .com, .net, and .org domains can now be
registered
with many different competing registrars. Go to
http://www.internic.net
for detailed information.
```

```
Domain Name: THE-NASHES.NET
Registrar: DOTSTER, INC.
Whois Server: whois.dotster.com
Referral URL: http://www.dotster.com/help/whois
Name Server: NS1.SPEAKEASY.NET
Name Server: NS1.THE-NASHES.NET
Updated Date: 03-aug-2000
```

```
>>> Last update of whois database: Tue, 21 Nov 2000 09:31:36
EST <<<
```

```
The Registry database contains ONLY .COM, .NET, .ORG, .EDU
domains and Registrars.
```

```
Found InterNIC referral to whois.dotster.com.
```

```
Registrant:
```

```
NashNet  
123 Some Street  
Our Town, ST 12345  
United States
```

```
Registrar: Dotster (http://www.dotster.com)
```

```
Domain Name: THE-NASHES.NET  
Created on: 02-NOV-98  
Expires on: 01-NOV-01  
Last Updated on: 03-AUG-00
```

```
Administrative Contact:
```

```
Nash, Angie angie@the-nashes.net  
123 Some Street  
Our Town, ST 12345  
United States  
555-555-0442
```

```
Technical Contact:
```

```
Nash, Jason jason@the-nashes.net  
123 Some Street  
Our Town, ST 12345  
United States  
555-555-0442
```

```
Domain servers in listed order:
```

```
NS1.THE-NASHES.NET  
NS1.SPEAKEASY.NET
```

finger

The `finger` utility is used to display information about a user on a system. This command displays information concerning the user's home directory, name, shell, login times, and mail. The `.plan` file, stored in the user's home directory, is also displayed. This file is created by the user and can display any information that the user wishes to make public. The following is an example use of the `finger` command:

```
# finger angie  
Login: angie Name: Angie Nash  
Directory: /home/angie Shell: /bin/bash2  
On since Tue Nov 21 17:40 (EST) on pts/0 from 10.254.90.151  
No mail.  
Plan:  
Angie Nash  
System Administrator  
angie@the-nashes.net  
555-0442
```



For information on the `dig` utility, see Chapter 16.

Configuration and Troubleshooting

Objective**1.12 Networking Fundamentals**

- **TCP/IP Troubleshooting & Configuration.** Demonstrate an understanding of the techniques required to list, configure and verify the operational status of network interfaces, change, view or configure the routing table, check the existing route table, correct an improperly set default route, manually add/start/stop/restart/delete/reconfigure network interfaces, and configure Linux as a DHCP client and a TCP/IP host and debug associated problems. May involve reviewing or configuring the following files or directories: `/etc/HOSTNAME` | `/etc/hostname`, `/etc/hosts`, `/etc/networks`, `/etc/host.conf`, `/etc/resolv.conf`, and other network configuration files for your distribution. May involve the use of the following commands and programs: `dhcpd`, `host`, `hostname` (`domainname`, `dnsdomainname`), `ifconfig`, `netstat`, `ping`, `route`, `traceroute`, the network scripts run during system initialization.

Network connectivity is essential to most Linux systems. This can include connectivity both to local networks and to the Internet using either phone lines or network adapters. This section explains some of the tools used to successfully connect a Linux system to a network. A thorough understanding of these tools is essential both for the test and for the job of system administrator.

Managing network interfaces

Several components compose the proper configuration of network interfaces on a Linux system. The IP address, hostname, route, and network adapter all work together to provide network connectivity. The tools covered in the following sections are used to assign these values and verify network connectivity.

DHCP

The Dynamic Host Configuration Protocol (DHCP) allows for dynamically assigned IP addresses. Along with the IP address and netmask, the default route and name servers can be assigned using DHCP. This eases the administrative headache of manually assigning an IP address to each system on the network and prevents duplicate IP addresses on the network. A DHCP server running the `dhcpd` daemon is configured with the information, which is allocated to machines as it is requested. This information can include IP addresses, subnet addresses, DNS servers, and gateway information. DHCP clients use broadcasts to search for a DHCP server. The server then responds with the appropriate configuration information. The IP information is leased to the client. The lease must be renewed, or the address is returned to the pool of available addresses. The client sends a lease renewal request to the DHCP server at various intervals in the lease period. Once the lease is renewed, this process is repeated. If the lease is not renewed before it expires, the DHCP client will request a new lease. To configure a Red Hat DHCP

client for the first Ethernet adapter, which has the name `eth0`, the following lines should be included in the `/etc/sysconfig/network-scripts/ifcfg-eth0` file:

```
DEVICE=eth0
BOOTPROTO=dhcp
ONBOOT=yes
```

To configure a Debian DHCP client for the first Ethernet adapter, the following line should be included in the `/etc/network/interfaces` file:

```
iface eth0 inet dhcp
```

hostname

The `hostname` command can be used to display the hostname of the local system. This capability can be useful for applications that need to use the hostname of a system. This information is configured in the `/etc/hosts` file.

```
# hostname
deedee.the-nashes.net
```

domainname and dnsdomainname

The `domainname` command on Red Hat systems and `dnsdomainname` command on Debian and Red Hat systems are used to view the fully qualified DNS domain name of the local system. A fully qualified DNS domain name includes the host name, such as `www`, and the second-level domain name, which is the organization name, such as `the-nashes`. It ends with the top-level domain. These include `.com`, `.net`, `.org`, `.edu`, `.gov`, as well as country codes such as `.ja`. These commands can be used only to view the DNS domain name, not to make changes. An example of this command is as follows:

```
# dnsdomainname
redhat.the-nashes.net
```

ifconfig

The `ifconfig` utility can be used to view and set the configuration of a network interface. Without options, the `ifconfig` utility simply displays the current configuration. An example of this output is the following:

```
# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:A0:CC:53:A7:91
          inet addr:10.254.90.135  Bcast:10.254.90.159  Mask:255.255.255.224
          UP BROADCAST RUNNING MTU:1500  Metric:1
          RX packets:246630 errors:0 dropped:0 overruns:0 frame:0
          TX packets:147531 errors:9 dropped:0 overruns:0 carrier:18
          collisions:31007 txqueuelen:100
          Interrupt:5 Base address:0xe400
```

```

lo          Link encap:Local Loopback
            inet addr:127.0.0.1  Mask:255.0.0.0
            UP LOOPBACK RUNNING  MTU:3924  Metric:1
            RX packets:359 errors:0 dropped:0 overruns:0 frame:0
            TX packets:359 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:0

```

When used with the interface name, the `ifconfig` utility can down or up the network interface. The `down` command is used to disable the network interface while the `up` command is used to enable the interface. This capability can be useful when removing the system from the network for maintenance and for reinitializing the network configuration. Other options that can be configured using the `ifconfig` utility include IP address, broadcast address, netmask address, and adapter settings. `ifconfig` is the primary command used when making changes to the network configuration on a system.

The `ifconfig` utility is used with the following syntax and the options shown in Table 15-7.

```
ifconfig interface options
```

Table 15-7
Options Used with `ifconfig`

Option	Use
<code>up</code>	Used to activate the specified interface.
<code>down</code>	Used to deactivate the specified interface.
<code>netmask ADDRESS</code>	Specifies the <i>ADDRESS</i> used for the subnet mask.
<code>ADDRESS</code>	Specifies the <i>ADDRESS</i> used for the IP address.
<code>irq ADDRESS</code>	Specifies the <i>IRQ ADDRESS</i> used by the interface.
<code>io_addr ADDRESS</code>	Specifies the <i>IO ADDRESS</i> used by the interface.
<code>media TYPE</code>	Specifies the media <i>TYPE</i> used by the interface, such as 10Base-T, 100Base-T, and so on.
<code>mem_start ADDRESS</code>	Specifies the start <i>ADDRESS</i> for the shared memory address used by the interface.

Following is an example of using the `ifconfig` utility to disable the `eth0` interface, to change the IP address to 10.254.90.153, to assign the netmask of 255.255.255.224 to the `eth0` interface, and to enable the `eth0` interface.

```
# ifconfig eth0 down
# ifconfig eth0 10.254.90.153 netmask 255.255.255.224
# ifconfig eth0 up
```

It is also possible to assign more than one IP address to a network interface using this utility. This is done by including a colon and number after the interface name so that the first IP address is assigned to `eth0:0`, the second assigned to `eth0:1`, and so on. The following example shows how to add a second IP address to the `eth0` interface.

```
# ifconfig eth0:1 10.254.90.144 netmask 255.255.255.224
```

netstat

The `netstat` utility is used to display network connections, routing tables, and interface statistics. Table 15-8 lists some of the options that can be used with the `netstat` utility. The information displayed with the `netstat` utility can be useful for viewing current network statistics and information. An example of the use of the `netstat` command follows. In this example the open sockets are displayed. There are two open TCP connections to the local address of 10.254.90.145 using the `ssh` ports and two remote connections. One of these is to the 10.254.90.146 address at port 1123, and the other is to 10.254.90.143 at the 2987 port. Both connections are currently established. Information is also displayed for the active UNIX domain sockets. This information includes the protocol (here they are all using the `unix` protocol), the attached processes known as the reference count, the type of socket access (in this case it is all stream), the connection state, and the process ID and path to the process using the socket.

```
# netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      20 10.254.90.145:ssh       10.254.90.146:1123     ESTABLISHED
tcp        0       0 10.254.90.145:ssh       10.254.90.143:2987     ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags   Type       State         I-Node Path
unix  1      [ ]     STREAM    CONNECTED    2544  @00000038
unix  1      [ ]     STREAM    CONNECTED    284   @00000007
unix  1      [ ]     STREAM    CONNECTED    237   @00000002
unix  1      [ ]     STREAM    CONNECTED    1436  @00000026
unix  1      [ ]     STREAM    CONNECTED    2545  /dev/log
unix  1      [ ]     STREAM    CONNECTED    1438  /dev/log
unix  1      [ ]     STREAM    CONNECTED    285   /dev/log
unix  1      [ ]     STREAM    CONNECTED    238   /dev/log
```



On some utilities, such as `netstat` and `route`, the `-n` option can be used to display the addresses in numeric form instead of as the DNS name.

Table 15-8
netstat Options

Option	Use
-r	Displays routing tables.
-i	Displays a table of all network interfaces.
-n	Displays numerical addresses.
-t	Displays TCP connection information.
-u	Displays UDP connection information.
-i p	Displays IP connection information.
-a	Displays all socket information.

ping

Earlier in this chapter, the operation of the `ping` utility was discussed. This utility is useful for troubleshooting problems in connectivity between hosts and on a local system. The `ping` utility when used with the loopback address (127.0.0.1, as mentioned earlier in this chapter) can verify the proper functioning of the TCP/IP protocol on a system. A successful ping of this address verifies that TCP/IP is installed and working on the local system. Successfully pinging the IP address of the local system verifies that the network adapter and TCP/IP are configured properly. Once you have pinged the local system successfully, the next step is pinging the default gateway; a response from that interface ensures that there are no cabling issues with the system and that it is able to access to the network.

All of these steps combined help isolate any problems that may exist in the network configuration of a system. Knowing which addresses to ping and what the response means can save valuable time in locating and resolving any network configuration problems.

tracert

The `tracert` utility can also be useful when troubleshooting connectivity issues between hosts. By viewing the path data takes between the two hosts, you can see the point at which the communication is interrupted. Routing issues can be located using this command. One thing to look for is repeatedly seeing the same routers listed. Such repetition can mean that there is a routing loop due to an incorrect configuration. This information can be especially useful when calling your ISP or bandwidth provider with a problem. This information can also be used to determine where routing problems exist on your own network.

Following is an example of the use of the `traceroute` utility. The example shows the information that is provided about each stop along the path between a destination and recipient.

```
# traceroute www.hungryminds.com
traceroute to www.hungryminds.com (38.170.216.15), 30 hops max, 38 byte packets
 1 192.168.1.1 (192.168.1.1) 0.028 ms 1.791 ms 1.476 ms
 2 10.233.32.1 (10.233.32.1) 12.678 ms 12.859 ms 13.707 ms
 3 24.25.1.61 (24.25.1.61) 8.093 ms 8.917 ms 45.217 ms
 4 24.25.1.49 (24.25.1.49) 8.488 ms 10.062 ms 15.093 ms
 5 24.93.64.97 (24.93.64.97) 17.706 ms 11.147 ms 9.593 ms
 6 12.124.234.37 (12.124.234.37) 25.965 ms 32.363 ms 31.102 ms
 7 gbr5-p80.wswdc.ip.att.net (12.123.9.58) 21.789 ms 24.097 ms 23.419 ms
 8 gbr3-p100.wswdc.ip.att.net (12.122.5.194) 32.894 ms 33.464 ms 42.197 ms
 9 gbr1-p70.wswdc.ip.att.net (12.122.1.158) 31.698 ms 22.994 ms 56.648 ms
10 gr1-p340.wswdc.ip.att.net (12.123.8.218) 33.614 ms 30.625 ms 29.107 ms
11 204.6.117.65 (204.6.117.65) 32.978 ms 21.781 ms 35.766 ms
12 nw.transit.tier1.us.psi.net (154.13.2.97) 88.099 ms 113.275 ms 130.073 ms
13 rc7.nw.us.psi.net (38.1.23.199) 95.938 ms 100.910 ms 108.484 ms
14 ip20.ci2.sanfrancisco.ca.us.psi.net (38.146.152.20) 91.448 ms 107.134 ms
150.364 ms
15 www.idgbooks.com (38.170.216.15) 99.785 ms 93.408 ms 108.695 ms
```

route

The `route` utility is used to display, delete, and add a route. The most common options for `route` are shown in Table 15-9. Having a default gateway configured in the routing table is imperative. Without this entry, no data will be delivered beyond the local network segment. The `route` command can also be used to configure the system to act as a router when multiple network interfaces are installed. With proper configuration, Linux is capable of routing information between connected network segments. When Linux routes information, it maintains a cache of destinations. This allows it to quickly route information to frequently requested destinations. The following is an example of the data displayed using the `route` command.

```
# route
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
localnet * 255.255.255.0 U 0 0 0 eth0
default 10.254.90.140 0.0.0.0 UG 0 0 0 eth0
```

This example shows a single route to the default gateway. The IP address of the default gateway is 10.254.90.140. The metric of 0 means this is the default route. Redundant routes will have higher metrics, which means the system will try them only if a lower metric route is unavailable. Table 15-10 shows the possible values for the Flags field.

Table 15-9
route Options

<i>Option</i>	<i>Function</i>
-v	Verbose mode.
-n	Displays addresses as IP. Does not resolve IP addresses to DNS names.
-N	Resolves hardware addresses to names.
-e	Displays extended information.
-C	Displays cache instead of the normal routing table.

Table 15-10
Flags Values

<i>Value</i>	<i>Meaning</i>
U	The route is available (Up).
H	The destination is a host.
G	Use gateway.
R	Reinstate, for dynamic routing.
D	Dynamic route. This route was installed by a routing daemon.
M	This route was modified by a routing daemon, or redirect.
C	Cache entry.
!	Rejected route.

It is also possible to add and delete information from the routing table. The following examples show how to delete and add a default route using the `route` command.

```
# route del default
# route add default gw 10.254.90.140
```

Managing network configuration files

On Red Hat systems the network configuration files are stored in `/etc/sysconfig/network-scripts`. The `ifcfg-eth0` file stored here contains the device name and IP address information for the first Ethernet adapter installed on the system. Information files for other network adapters including the loopback and PPP adapters are also stored here.

On Debian systems the network configuration file is located at `/etc/network/interfaces`. This file contains information on IP address assignment for all adapters configured on the system. The loopback adapter, Ethernet adapters, and PPP adapters all have sections in this file.

/etc/hosts

The `/etc/hosts` file is used on both Debian and Red Hat systems to provide IP address-to-domain name resolution. This file is checked before any DNS servers, so adding frequently accessed systems to this file can reduce network traffic. However, it is important to keep in mind that this file must be updated when there are any changes to the IP address or hostname configuration of the systems listed. This file can only contain hostnames that have a static IP address; those using DHCP shouldn't be listed here because the IP address may change. The local hostname is also stored in this file. The following are sample entries from the `/etc/hosts` file:

```
127.0.0.1          localhost.localdomain localhost
10.254.90.133     norbert.the-nashes.net
10.254.90.135     deedee.the-nashes.net
```

/etc/services

The `/etc/services` file contains information concerning port mappings to specific services. As you can see from the following example, mappings to both TCP and UDP are included for most ports. Any changes to port mappings should be done using this file. You may wish to change a port assignment as a means of security. If you wish to run an FTP server but want to hide it, you can change the port assignment. This will prevent people from connecting to the server without knowing the correct port. When an application is run, it checks the `/etc/services` file to see which port to use.

```
ftp-data          20/tcp
ftp               21/tcp
fsp              21/udp          fspd
ssh              22/tcp          # SSH Remote Login Protocol
ssh              22/udp          # SSH Remote Login Protocol
telnet           23/tcp
# 24 - private
smtp             25/tcp          mail
# 26 - unassigned
time            37/tcp          timserver
time            37/udp          timserver
rlp             39/udp          resource        # resource
nameserver      42/tcp          name            # IEN 116
whois           43/tcp          nickname
```

/etc/resolv.conf

The `/etc/resolv.conf` file contains the name servers to be used for hostname resolution along with the domain names to search when no fully qualified domain name has been entered. These servers are required to translate hostnames and domain names to IP addresses, such as `redhat.the-nashes.net` to the IP address of 10.254.90.145. The network administrator or the Internet service provider supplies these name servers. An example of the contents of this file is as follows:

```
search the-nashes.net
nameserver 10.254.90.131
nameserver 4.2.2.1
```

/etc/HOSTNAME and /etc/hostname

`/etc/HOSTNAME` on Debian systems and `/etc/hostname` on Red Hat systems contain the hostname information for the local machine. These files contain the fully qualified domain name for the system. Following is an example of an entry in the `/etc/hostname` file on a Red Hat-based system.

```
deedee.the-nashes.net
```

Configuring PPP

Objective

1.12 Networking Fundamentals

- **Configure and use PPP.** Define the chat sequence to connect (given a login example), setup commands to be run automatically when a PPP connection is made, initiate or terminate a PPP connection, initiate or terminate an ISDN connection, set PPP to automatically reconnect if disconnected.

PPP, or Point-to-Point Protocol, is used for dial-up connections. This protocol allows the TCP/IP suite of protocols to work over telephone wires. When using PPP on Linux, the PPP connection is configured as a network interface. A variety of tools available for Linux can help with the configuration and use of the PPP interface.

Chat scripts

Chat scripts are used by PPP to make the appropriate dial-up connections. The chat script first establishes the connection, and then the `pppd` program configures the connection. The chat script contains all the connection information, such as telephone number, user ID, password, and any necessary connection strings. The script contains expected strings followed by response string. It is recommended that you supply only enough expected string information to identify the string. If you supply more information and the strings don't match, an error will occur. Following is an example of a chat script entry:

```
ogin: ppp ssword: hello2u2
```

This entry will send the string `ppp` in response to the expected string of `ogin` at the login prompt. For the expected string `ssword` from the password prompt, the string `hello2u2` is sent. Following is a more detailed chat script that will dial the ISP and enter the login information when prompted:

```
ABORT BUSY
ECHO OFF
SAY "Dialling your ISP...\n"
' ATDT5551212
TIMEOUT 120
SAY "Waiting up to 2 minutes for connection ... "
CONNECT ''
SAY "Connected, now logging in ...0
ogin: account
ssword: pass
$ SAY "Logged in OK ...0 etc ...
```

This script can be stored anywhere on the system as long as the user has the proper permissions to access it. The user specifies the script to run as an argument with the `pppd` command. The modem control codes can be found in the documentation that comes with your hardware. The Internet service provider provides the other information.

pppd

The `pppd` program works with the chat script to establish the PPP connection. Once the chat script has made the connection, `pppd` configures it based on the options entered. The program is run with a specified chat script and connection device. The `pppd` command is run at the command line using the following syntax:

```
pppd -option argument
```

A static address can be specified or DHCP can be used to configure the address. Some of the options used with `pppd` are listed in Table 15-11.

Table 15-11
pppd Options

<i>Option</i>	<i>Use</i>
<code>connect SCRIPT</code>	Establishes a PPP connection with the chat script specified.
<code>disconnect SCRIPT</code>	Disconnects a PPP connection using the script specified.
<code>holdoff TIME</code>	Specifies how long to wait after a link has been terminated before it is reinitialized.
<code>idle SEC</code>	Specifies how many seconds to wait before the PPP connection is disconnected after no data has been sent.
<code>modem</code>	Dictates that the modem control lines are used.

pppd reads options from the files `/etc/ppp/options`, `~/.ppprc`, and `/etc/ppp/options.ttyname` (in that order) before processing the options on the command line. This allows the options to be configured in one of these scripts instead of entering them at the command line.

The command to initiate a PPP connection using an ISDN line is as follows:

```
# pppd connect 'chat -f /etc/ppp/ppp_isdn.chat' holdoff 60
/dev/isdn 64000
```

Key Point Summary

This chapter covers many important concepts and utilities used to provide network connectivity on a Linux system. Some of the most important items covered in this chapter include the following:

- ♦ TCP/IP addresses are used to uniquely identify each network interface on the network. These addresses can be viewed using binary or decimal format.
- ♦ The three most commonly used network classes and matching network masks are as follows:
 - **Class A:** 1.0.0.0–126.0.0.0, with the mask 255.0.0.0
 - **Class B:** 128.0.0.0–191.0.0.0, with the mask 255.255.0.0
 - **Class C:** 192.0.0.0–223.0.0.0, with the mask 255.255.255.0
- ♦ The TCP/IP protocol suite comprises the following primary protocols: IP, ARP, ICMP, TCP, and UDP. IP is responsible for the delivery of datagrams across the network. ARP provides IP address-to-hardware address mapping and translation, and ICMP reports errors in the delivery of packets. TCP is a connection-oriented protocol that provides reliable transmission of data using sessions. UDP provides connectionless transmissions with less overhead than TCP.
- ♦ TCP and UDP make use of ports and sockets to manage multiple connections to a variety of applications simultaneously. Applications are mapped to specific ports using the `/etc/services` file.
- ♦ The FTP application allows for file transfer from remote systems, while Telnet allows remote access to a system's shell.
- ♦ The `host` command allows you to discover IP or hostname information for the specified system. The `whois` command provides domain, name server, and contact information for the specified domain. The `finger` utility can be used to discover information about a specific user on a system.

- ♦ The `ping` and `traceroute` utilities use ICMP and can reveal information concerning the connectivity between a local and remote system. This information can be useful for locating the source of a communication failure between two systems.
- ♦ DHCP is used to allow dynamic IP address assignment; `dhcpd` allows a Linux machine to function as a DHCP server. DHCP clients are configured using network configuration files.
- ♦ The `ifconfig`, `hostname`, and `route` commands are all used to specify network configuration options. These commands along with `netstat`, `domainname` (Red Hat), and `dnsdomainname` (Red Hat and Debian) also allow the present configuration to be viewed.
- ♦ The configuration file on Debian that is used to configure the network adapter is `/etc/network/interfaces`, and the file used to configure the hostname is `/etc/HOSTNAME`.
- ♦ Red Hat systems use the `/etc/sysconfig/network-scripts` to store files for each network interface. The hostname on these systems is specified in the `/etc/hostname` file.
- ♦ Both systems use the `/etc/resolv.conf` file to store name servers and domain search strings, the `/etc/hosts` file to store hostname-to-IP address mappings, and `/etc/services` to provide port mappings.
- ♦ PPP connections on Linux systems utilize `chat` scripts to dial out and log in to the PPP server. These files are configured with expected strings and responses that are sent to provide the login name and password.
- ♦ Once the `chat` script establishes the connection, `pppd` configures it. The `pppd` utility is used along with options that specify the device and `chat` script to use. Options can also be used to specify IP address, subnet mask, idle time-out, and automatic reconnection of the PPP connection.



STUDY GUIDE

The following questions and exercises will allow you to review the information covered in this chapter. Take your time when completing this section of the chapter, carefully reviewing any questions that you may have. Answering the question correctly is not as important as understanding the answer, so review any material that you might still be unsure of. Being comfortable with the questions and answers presented here will help you be more prepared for the certification exam questions.

Assessment Questions

1. Which of the following is an example of a class C network address?
 - A. 10.254.90.135
 - B. 172.16.98.42
 - C. 255.255.255.255
 - D. 192.168.90.131
2. The _____ is used to separate the network ID from the host ID in a TCP/IP address.
3. How many available network IDs are provided by 12 binary bits?
 - A. 254
 - B. 4,094
 - C. 4,096
 - D. 65,534
4. Which protocol provides connection-oriented delivery of packets using sessions?
 - A. IP
 - B. ARP
 - C. TCP
 - D. UDP

5. By default, SMTP operates on which port?
 - A. 21
 - B. 23
 - C. 25
 - D. 119

6. The _____ protocol provides IP address-to-hardware address translation.

7. Which command is used to upload a file from the local machine to a remote FTP server?
 - A. put
 - B. cp
 - C. mv
 - D. place

8. Which utility allows you to view the domain name of the specified IP address?
 - A. hostname
 - B. domainname
 - C. dnsdomainname
 - D. host

9. The _____ utility provides information about the path of data between a local and remote host.

10. Which utility provides contact and name server information about the specified domain?
 - A. finger
 - B. host
 - C. nameserver
 - D. whois

11. A Linux machine can provide DHCP configuration information to other hosts when running which application?
 - A. dhcpd
 - B. dhcpcd
 - C. bootproto
 - D. dhcp

12. Which utility is used to change the IP address of a network interface?
- A. ipconfig
 - B. ifconfig
 - C. hostname
 - D. netstat
13. Which two utilities can be used to view the routing table?
- A. netstat
 - B. traceroute
 - C. route
 - D. ifconfig
14. On a Debian system, which file contains the hostname of the local machine?
- A. /etc/hosts
 - B. /etc/HOSTS
 - C. /etc/hostname
 - D. /etc/HOSTNAME
15. Which file contains the name servers to be used by the local machine for hostname resolution?
- A. /etc/hosts
 - B. /etc/resolv.conf
 - C. /etc/services/
 - D. /etc/network/interfaces
16. Which of the following is a correct entry in the /etc/hosts file?
- A. deede 10.254.90.135/ip
 - B. nameserver 10.254.90.131
 - C. search the-nashes.net
 - D. 10.254.90.135 deede.the-nashes.net
17. Chat scripts are used to perform which of the following functions?
- A. Specify connection device
 - B. Configure a PPP connection
 - C. Establish a PPP connection
 - D. Specify idle timeout of a PPP connection

18. The `pppd` application must receive its IP address via DHCP.
- A. True
 - B. False
19. Which of the following is NOT a valid option of the `pppd` command?
- A. `reconnect`
 - B. `connect`
 - C. `disconnect`
 - D. `holdoff`
20. Which of the following utilities is used to stop a network interface?
- A. `ifconfig`
 - B. `netstat`
 - C. `down`
 - D. `stop`

Scenarios

1. Your company has decided to install a small network in a remote office. This office will not be connected to the Internet, so you decide to use the 192.168.1.0 network address. Which subnet mask would allow you to have 32 computers on each subnet of the network? How many subnets would this address allow?

Answers to Chapter Questions

Chapter Pre-Test

1. Each host on the Internet is known by a unique IP addresses.
2. The ICMP protocol handles error reporting.
3. The NetBIOS session service uses port 139.
4. The Telnet application provides terminal emulation.
5. The `/etc/hosts` file provides manual IP address-to-hostname mappings.
6. The `traceroute` tool allows you to view the path of a packet through the network.
7. DHCP is used to automatically configure a workstation's IP information.

8. The `ifconfig` tool is used to assign an IP address to a network adapter.
9. A `chat` script is used to supply information such as telephone number, user name, and password that are used when making a PPP connection.
10. The `holdoff` option is used to automatically reconnect a PPP connection in the case of an interruption.

Assessment Questions

1. **D.** The address 192.168.90.131 is a class C address between the ranges of 192.0.0.0 and 223.0.0.0. See the “Dividing networks with subnet masks” section for more information.
2. **subnet mask.** The subnet mask is anded with the IP address to provide the network ID and the host ID. See the “Dividing networks with subnet masks” section for more information.
3. **B.** There are 4,094 available network IDs provided by 12 bits. The formula for calculation is $2^{12} - 2$. See the “Dividing networks with subnet masks” section for more information.
4. **C.** The TCP protocol is connection-oriented, using sessions to provide reliable transfer of packets. See the “Transmission Control Protocol” section for more information.
5. **C.** By default, SMTP operates on port 25. See the “Ports” section for more information.
6. **ARP.** ARP provides hardware address-to-IP address mapping and translation. See the “Address Resolution Protocol” section for more information.
7. **A.** The `put` command is used to upload files to an FTP server. See the “FTP” section for more information.
8. **D.** The `host` utility allows you to view the host and domain name when an IP address is entered. See the “host” section for more information.
9. **traceroute.** The `traceroute` utility displays every hop between the local and remote hosts. See the “traceroute” section for more information.
10. **D.** The `whois` utility provides information about the registration, name servers, and contacts for the specified domain. See the “whois” section for more information.
11. **A.** The `dhcpcd` daemon allows a Linux server to provide DHCP information to clients. See the “DHCP” section for more information.
12. **B.** The `ifconfig` utility is used to change the IP address of a network interface. The `ipconfig` utility is used by Windows-based systems. See the “ifconfig” section for more information.

13. **A and C.** The routing table can be viewed using the `route` command and using `netstat -r`. See the “route” and “netstat” sections for more information.
14. **D.** Debian systems use the `/etc/HOSTNAME` file to store the hostname of the local machine. The `/etc/hostname` file is used by Red Hat. See the “`/etc/HOSTNAME` and `/etc/hostname`” section for more information.
15. **B.** The `/etc/resolv.conf` file contains the domain search listing and the name servers to use for hostname resolution. See the “`/etc/resolv.conf`” section for more information.
16. **D.** Entries in the `/etc/hosts` file contain an IP address followed by the corresponding hostname. See the “`/etc/hosts`” section for more information.
17. **C.** Chat scripts are used to establish a PPP connection; `pppd` is used to configure the connection. See the “`pppd`” section for more information.
18. **B.** `pppd` can utilize either static or dynamic IP addresses. See the “`pppd`” section for more information.
19. **A.** `reconnect` is not a valid option of `pppd`. See the “`pppd`” section for more information.
20. **A.** The `ifconfig` utility is used to stop a network interface. See the “`ifconfig`” section for more information.

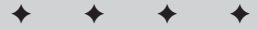
Scenarios

1. The 192.168.1.0 network address uses the default mask of 255.255.255.0 to allow for 254 hosts on one subnet. Five bits allow for only 30 hosts, so six bits must be used for hosts. This allows for up to 62 hosts on a subnet. The two bits that are left for subnets will allow for two possible subnets. This would appear as 1100 0000 and would create the subnet mask of 255.255.255.192.

Managing Network Services

16

C H A P T E R



EXAM OBJECTIVES

Exam 102 ♦ General Linux, Part 2

1.12 Networking Fundamentals

- **Fundamentals of TCP/IP.** Demonstrate an understanding of network masks and what they mean (i.e. determine a network address for a host based on its subnet mask), understand basic TCP/IP protocols (TCP, UDP, ICMP) and also PPP, demonstrate an understanding of the purpose and use of the more common ports found in `/etc/services` (20, 21, 23, 25, 53, 80, 110, 119, 139, 143, 161), demonstrate an correct understanding of the function and application of a default route. Execute basic TCP/IP tasks: FTP, anonymous FTP, telnet, host, ping, dig, traceroute, whois.

1.13 Networking Services

- **Configure and Manage inetd and related services.** Configure what services are available through inetd, use tcp-wrappers to allow or deny services on a host-by-host basis, manually start, stop, and restart internet services, configure basic network services including telnet and ftp. Includes managing `inetd.conf`, `hosts.allow`, and `hosts.deny`.

Continued

EXAM OBJECTIVES (CONTINUED)**1.13 Networking Services** (*Continued*)

- **Operate and perform basic configuration of sendmail.** Modify simple parameters in sendmail config files (modify the DS value for the “Smart Host” if necessary), create mail aliases, manage the mail queue, start and stop sendmail, configure mail forwarding (.forward), perform basic troubleshooting of sendmail. Does not include advanced custom configuration of sendmail. Includes commands mailq, sendmail, and newaliases. Includes aliases and mail/ config files.
- **Operate and perform basic configuration of apache.** Modify simple parameters in apache config files, start, stop, and restart httpd, arrange for automatic restarting of httpd upon boot. Does not include advanced custom configuration of apache. Includes managing httpd conf files.
- **Properly manage the NFS, smb, and nmb daemons.** Mount remote filesystems using NFS, configure NFS for exporting local filesystems, start, stop, and restart the NFS server. Install and configure Samba using the included GUI tools or direct edit of the /etc/smb.conf file (Note: this deliberately excludes advanced NT domain issues but includes simple sharing of home directories and printers, as well as correctly setting the nmbd as a WINS client).
- **Setup and configure basic DNS services.** Configure hostname lookups by maintaining the /etc/hosts, /etc/resolv.conf, /etc/host.conf, and /etc/nsswitch.conf files, troubleshoot problems with local caching-only name server. Requires an understanding of the domain registration and DNS translation process. Requires understanding key differences in config files for bind 4 and bind 8. Includes commands nslookup, host. Files: named.boot (v.4) or named.conf (v.8)

CHAPTER PRE-TEST

1. Which tool is used to control the Apache daemon?
2. Which DNS record type is used to route mail?
3. Which file controls the directories exported by NFS?
4. Which file is used to set logging options on the wu-ftpd server?
5. Which tool displays the current connections to an FTP server?
6. What is the name of the script in Red Hat that controls sendmail?
7. Which directive controls the log format in Apache?
8. Which tool displays statistics about an NFS server?
9. What is the name of the Samba client tool?
10. What is the name of the main configuration file for BIND v4?

This chapter covers the basic information you need to manage the most popular network services normally used with Linux. Each section will cover a different application and go over the major points of configuration and management. The exam does not delve deeply into the different applications, but you should know the function of each and the major points of the configuration files.

Using the Internet Super Server

Objective

1.13 Networking Services

- **Configure and Manage inetd and related services.** Configure what services are available through inetd, use tcpwrappers to allow or deny services on a host-by-host basis, manually start, stop, and restart internet services, configure basic network services including telnet and ftp. Includes managing inetd.conf, hosts.allow, and hosts.deny.

The `inetd` daemon is sometimes known as the Internet Super Server because it has the job of managing many other services. Normally, a special daemon is started for each service that a server offers to its clients. This daemon resides in memory until it is manually unloaded. If a server offers many services the different daemons can consume a considerable amount of memory. Memory is not as scarce in servers today as it once was, but many daemons still remain idle most of the time.

This problem was resolved with `inetd`, short for *Internet daemon*. `inetd` monitors all of the ports for the services it manages and starts the appropriate daemon when a client requests a specific service. This reduces memory overhead by keeping only `inetd` loaded all of the time. Some simple services are now handled by `inetd` instead of requiring separate daemons.

`inetd` should not be used with services that take a long time to start or that service many users. While a Web server could be run from `inetd`, it is not usually advisable to because Web servers take several seconds to begin and often get many users. Other packages, such as Secure Shell (SSH), generate encryption keys when they start, which would cause significant delays any time a user connects. The process of starting up services for a heavily used service can cause serious performance degradation on the server.

Configuring inetd

`inetd` is configured using the `/etc/inetd.conf` file. The following is an example of this file:

```
#echo stream tcp nowait root internal
#echo dgram udp wait root internal
#discard stream tcp nowait root internal
```

```

#discard      dgram  udp    wait   root   internal
#daytime      stream tcp    nowait root   internal
#daytime      dgram  udp    wait   root   internal
#chargen      stream tcp    nowait root   internal
#chargen      dgram  udp    wait   root   internal
#time         stream tcp    nowait root   internal
#time         dgram  udp    wait   root   internal
#
# These are standard services.
#
ftp           stream tcp    nowait root   /usr/sbin/tcpd  in.ftpd -l -a
telnet       stream tcp    nowait root   /usr/sbin/tcpd  in.telnetd
#
# Shell, login, exec, comsat and talk are BSD protocols.
#
shell        stream tcp    nowait root   /usr/sbin/tcpd  in.rshd
login        stream tcp    nowait root   /usr/sbin/tcpd  in.rlogind
#exec        stream tcp    nowait root   /usr/sbin/tcpd  in.rexecd
#comsat      dgram  udp    wait   root   /usr/sbin/tcpd  in.comsat
talk         dgram  udp    wait   nobody.tty /usr/sbin/tcpd  in.talkd
ntalk        dgram  udp    wait   nobody.tty /usr/sbin/tcpd  in.ntalkd
#dtalk       stream tcp    wait   nobody.tty /usr/sbin/tcpd  in.dtalkd
#
# Pop and imap mail services et al
#
#pop-2       stream tcp    nowait root   /usr/sbin/tcpd  ipop2d
#pop-3       stream tcp    nowait root   /usr/sbin/tcpd  ipop3d
#imap        stream tcp    nowait root   /usr/sbin/tcpd  imapd
#
# The Internet UUCP service.
#
#uucp        stream tcp    nowait uucp   /usr/sbin/tcpd  /usr/lib/uucp/uucico
-1
#
# Tftp service is provided primarily for booting. Most sites
# run this only on machines acting as "boot servers." Do not uncomment
# this unless you *need* it.
#
#tftp        dgram  udp    wait   root   /usr/sbin/tcpd  in.tftpd
#bootps     dgram  udp    wait   root   /usr/sbin/tcpd  bootpd
#
# Finger, systat and netstat give out user information which may be
# valuable to potential "system crackers." Many sites choose to disable
# some or all of these services to improve security.
#
finger       stream tcp    nowait nobody /usr/sbin/tcpd  in.fingerd
#cfinger    stream tcp    nowait root   /usr/sbin/tcpd  in.cfingerd
#systat     stream tcp    nowait guest  /usr/sbin/tcpd  /bin/ps -auwx
#netstat    stream tcp    nowait guest  /usr/sbin/tcpd  /bin/netstat
-f inet
#
# Authentication
#

```

```
# identd is run standalone now
#
#auth stream tcp wait root /usr/sbin/in.identd in.identd -e -o
#
# End of inetd.conf
```

Each line in the `/etc/inetd.conf` file has the following syntax and is explained in Table 16-1:

Service socket_type protocol flags user server_path arguments

Lines that begin with “#” are comments. Comments are frequently used to stop `inetd` from managing a service without removing the configuration information, in case it is needed later.

Table 16-1
inetd Syntax

<i>Option</i>	<i>Explanation</i>
<i>Service</i>	The name of the service to be started by <code>inetd</code> . The name is checked against the <code>/etc/services</code> file to see which port number to monitor.
<i>Socket_Type</i>	TCP uses a socket type of stream while UDP uses datagram.
<i>Protocol</i>	A protocol as referenced in <code>/etc/protocols</code> , normally <code>dp</code> , <code>tcp</code> , <code>rpc/tcp</code> , or <code>rpc/udp</code> .
<i>Flags</i>	<code>wait</code> for UDP connections; all others are <code>nowait</code> .
<i>User</i>	The user account under which the started service should execute. This account should have only enough permissions to correctly run the daemon. Any extra permissions may introduce a security risk.
<i>Server_Path</i>	Path to the server executable or the name of the service if supplied by <code>inetd</code> .
<i>Arguments</i>	Any arguments to be supplied to the service.

Restarting the `inetd` process

The `inetd` daemon must be restarted any time a change is made to the `/etc/inetd.conf` file. If the system is Red Hat or a derivative, you can use the `inet` script:

```
[root@redhat ~]# /etc/rc.d/init.d/inet restart
Stopping INET services:  OK ]
Starting INET services:  OK ]
```

Debian and its derivatives use the `inetd` script:

```
debian:~# /etc/init.d/inetd restart
Restarting internet superserver: inetd.
```

In either case, `inetd` can be restarted by using the HUP signal:

```
debian:~# kill -HUP inetd
```

Or you can do it like this:

```
debian:~# ps aux | grep inetd
root      27947  0.0  0.2 1536  668 ?        S    22:53   0:00
/usr/sbin/inetd
debian:~# killall -HUP inetd
debian:~# kill -HUP 27947
```



Caution

The `inetd` daemon must be restarted whenever you make a change to `/etc/inetd.conf`.

Configuring Basic Network Services

Objective

1.12 Networking Fundamentals

- **Fundamentals of TCP/IP.** Demonstrate an understanding of network masks and what they mean (i.e. determine a network address for a host based on its subnet mask), understand basic TCP/IP protocols (TCP, UDP, ICMP) and also PPP, demonstrate an understanding of the purpose and use of the more common ports found in `/etc/services` (20, 21, 23, 25, 53, 80, 110, 119, 139, 143, 161), demonstrate a correct understanding of the function and application of a default route. Execute basic TCP/IP tasks: FTP, anonymous FTP, telnet, host, ping, dig, traceroute, whois.

Most Linux distributions install FTP and Telnet servers to provide remote file transfer and terminal access. These services are usually handled by `inetd`, and configuration is minimal.

Configuring an FTP server

FTP is one of the oldest and most used services on the Internet and private networks. While it can be set up to require login and can be secured, the most common setup is to share files anonymously. FTP is useful for distributing software, updates, or fixes. Unlike HTTP, FTP allows the user to easily browse a tree structure and see the size and date of files without any extra work.



The most popular FTP server is still wu-ftp, but ProFTPD is a popular new alternative. ProFTPD has a similar configuration structure to Apache.

The most popular FTP server is wu-ftp, created by Washington University. It is included with most Linux distributions. The `ftpd` daemon is normally run by `inetd` and enabled through the `/etc/inetd.conf` file. The following line shows the appropriate entry:

```
ftp      stream  tcp      nowait  root    /usr/sbin/tcpd  in.ftpd  -l  -a
```

The `ftpd` daemon takes the parameters shown in Table 16-2.

Table 16-2
ftpd Parameters

<i>Parameter</i>	<i>Function</i>
-V	Display version information.
-d	Write debug information to <code>syslog</code> .
-v	Write debug information to <code>syslog</code> .
-l	Log connections to <code>syslog</code> .
-t <value>	Set the amount of time a client may be idle before being disconnected.
-a	Use the <code>ftppass</code> configuration file.
-A	Do not use the <code>ftppass</code> configuration file (default).
-L	Log all user commands to <code>syslog</code> .
-I	Log received files to <code>xferlog</code> .
-i	Enable checking of the client name via <code>ident</code> .
-o	Log transferred files to <code>xferlog</code> .
-X	Log <code>-i</code> and <code>-o</code> information to <code>syslog</code> to facilitate central logging.
-W	Do not log logins to <code>wtmp</code> .
-r	Chroot (change root) the daemon as soon as it loads, not at login. This changes the perceived root directory for the daemon. For example, if you chroot the daemon to <code>/var/ftp</code> , the daemon will use that as its root. If someone breaks in to your FTP server they can not go below that directory.

Setting up anonymous FTP

Configuring anonymous FTP requires a few steps. Some Linux distributions automatically install and set up the server to allow anonymous access. The following steps walk you through setting it up manually and show you how it was done if your distribution did it for you.

1. Create a user named “ftp”.
2. Create the following subdirectories under the ftp user’s home directory: `bin`, `etc`, `lib`, and `pub`.
3. Copy any needed binaries to the `bin` directory. These may include `ls`, `more`, `gzip`, and `tar`.
4. Copy any needed library files for the binary files to the new `lib` directory.
5. Copy `/etc/passwd` and `/etc/group` to the new `etc` directory.
6. Edit the `/etc/passwd` file and change any encrypted passwords to asterisks.
7. Set the directory permissions as shown in Table 16-3.

When a user logs in via anonymous FTP, the server changes their perceived root, known as `chroot`, to the ftp user’s home directory. The newly created directories are needed to provide any files a user may need.

Table 16-3
Permissions

<i>File/Directory</i>	<i>Owner/Group</i>	<i>Permissions</i>
<code>~ftp</code>	<code>root/root</code>	<code>555 (r-xr-xr-x)</code>
<code>~ftp/bin</code>	<code>root/root</code>	<code>555 (r-xr-xr-x)</code>
<code>~ftp/bin/<file></code>	<code>root/root</code>	<code>111 (--x--x--x)</code>
<code>~ftp/etc</code>	<code>root/root</code>	<code>555 (r-xr-xr-x)</code>
<code>~ftp/etc/passwd</code>	<code>root/root</code>	<code>444 (r--r--r--)</code>
<code>~ftp/etc/group</code>	<code>root/root</code>	<code>444 (r--r--r--)</code>
<code>~ftp/pub</code>	<code>root/ftp</code>	<code>755 (rwxr-xr-x)</code>
<code>~ftp/lib</code>	<code>root/root</code>	<code>755 (rwxr--r--)</code>
<code>~ftp/lib/<file></code>	<code>root/root</code>	<code>755 (rwxr-xr-w)</code>

For the sake of security, it is a good idea to remove unneeded users from the `/etc/passwd` file when copying it to the `~ftp/etc` directory. Only the ftp, daemon, and root users need to be in the file. This way an anonymous user cannot get a list

of user accounts on the machine to aid in hacking. It is also recommended to set the ftp user's shell to `/bin/false`, so there is no way for the user to Telnet or log in remotely.



The anonymous FTP account should not have a valid shell statement in `/etc/passwd`. The `/bin/false` entry is normally used. Whatever fake statement is used, it must be present in the `/etc/shells` file.

Customizing configuration files

The `wu-ftpd` server provides several configuration files to control access to the server and other options. These are the following:

- ♦ `/etc/ftppaccess`
- ♦ `/etc/ftphosts`
- ♦ `/etc/ftpusers`
- ♦ `/etc/ftpgroups`
- ♦ `/etc/ftpconversions`
- ♦ `/var/log/xferlog`

Most of these files are already configured to work with an anonymous server during the Linux distribution install. Only when the service is expanded do you need to edit these.

The `ftppaccess` file

The `ftppaccess` file can be used to set restrictions, logging, information, and other options on the FTP server. An example `ftppaccess` file is shown here:

```
class    all    real,guest,anonymous    *
email   Jason@the-nashes.net
loginfails 5
readme  README*    login
readme  README*    cwd=*
message /welcome.msg          login
message .message          cwd=*
compress      yes          all
tar           yes          all
chmod        no           guest,anonymous
delete       no           guest,anonymous
overwrite    no           guest,anonymous
rename       no           guest,anonymous
log transfers anonymous,real inbound,outbound
```

```
shutdown /etc/shutmsg
passwd-check rfc822 warn
```

Table 16-4 shows the common `ftppaccess` entries.

Table 16-4 Common <code>ftppaccess</code> Entries	
Information Entries	Function
<code>banner <file></code>	The specified text file is displayed when a user connects, before their user name or password is entered. The full path from the volume root is required.
<code>message <file> {<when> {<class> ...}</code>	The file defined in the <code><file></code> is displayed to the client when specified. The <code><when></code> parameter can either be <code>LOGIN</code> , to display at login, or <code>CWD=<directory></code> , to display the file when a user changes into the defined directory. Variables known as “Magic Cookies” can be put in the text file to have the FTP server replace them with other information. See the man page for more information.
<code>email <email address></code>	Defines the administrator’s e-mail address that can be referenced with a “Magic Cookie.”
<code>readme <path> {<when> {<class>}}</code>	Displays a message telling the user when the file specified in <code><path></code> was modified. This is useful for a README file containing important information and for informing users of updates.
Access Control	Function
<code>autogroup <group name> <class> {<class...></code>	If the anonymous FTP user is a member of any defined <code><classes></code> , that user’s group is changed to <code><group name></code> .
<code>class <class> <typelist> <address glob> {<address glob>...}</code>	Defines a <code><class></code> of users based on their <code><address glob></code> . The <code><typelist></code> defines the user type as <code>anonymous</code> , <code>guest</code> , and <code>real</code> . The groups of users are then referenced elsewhere in the <code>ftppaccess</code> file to make administration easier.
<code>deny <address glob> <message file></code>	The <code><address glob></code> is either a domain name or a numeric address, such as <code>192.168.1.*</code> . Displays the <code><message file></code> to any address matching the <code><address glob></code> and then denies access.

Continued

Table 16-4 (continued)

Access Control	Function
limit <class> <n> <times> <message_file>	Limits the <class> to <n> number of users connected during <times>. If the limit is reached the <message_file> is displayed.
noretrieve [absolute relative] [class=<classname>] ... [-] <file name> <filename> ...	Blocks the retrieval of certain files. Paths can be set to either absolute or relative, and restricted by class. This is useful for blocking the downloading of the /etc/passwd file.
allow-retrieve [absolute relative] [class=<classname>]... [-] <filename> ...	Allows files to be retrieved that would normally be blocked by the noretrieve entry.
loginfails <number>	After <number> of attempts to enter the correct password, the FTP connection is disconnected.
private <yes no>	Allows the user to change to a different group using the SITE GROUP and SITE GPASS commands. The group must be specified in the /etc/ftpgroups file.
Logging	Function
log commands <typelist>	Logs commands for user types specified in <typelist>. These are real, anonymous, and guest.
log transfers <typelist> <directions>	Logs transfers by <typelist>. <directions> are either inbound or outbound.
log security <typelist>	Logs security violations for the user type specified in <typelist>. The violations are things such as trying to download files protected by noretrieve.
Miscellaneous Capabilities	Functions
alias <string> <dir>	Creates an alias that refers to a directory. Used only by the cd command.
cdpath <dir>	Defines a search path for directories. If a user should enter cd foo , the current directory and those listed with cdpath will be checked.
compress <yes no> <classglob> [<classglob> ...]	Enables the compress function so files can be shrunk before downloading.
tar <yes no> <classglob> [<classglob> ...]	Enables the tar function.

Miscellaneous Capabilities	Functions
<code>shutdown <path></code>	The server periodically checks the file specified in <code><path></code> to see if the server is going to be shut down. The file has the following format: <code><year> <month> <day> <hour> <minute> <deny_offset> <disc_offset> <text></code> . <code><deny_offset></code> and <code><disc_offset></code> specify the offset times before shutdown when connections will be denied and then disconnected.
<code>daemonaddress <address></code>	Defines the IP addresses that the server will listen on. By default it is all addresses.
<code>virtual <address> <root banner logfile> <path></code>	Enables virtual server capabilities.
Permission Capabilities	Function
<code>chmod <yes no> <typelist></code>	Sets the ability to change file permissions.
<code>delete <yes no> <typelist></code>	Sets the ability to delete files.
<code>overwrite <yes no> <typelist></code>	Sets the ability to overwrite files.
<code>rename <yes no> <typelist></code>	Sets the ability to rename files.
<code>umask <yes no> <typelist></code>	Sets the ability to set file creation permissions.
<code>passwd-check <none trivial rfc822> (<enforce warn>)</code>	<p>Sets the level of password checking done by the server for anonymous FTP. The following options are available:</p> <ul style="list-style-type: none"> <code>none</code> – No password checking is performed. <code>trivial</code> – The password must contain an @ character. <code>rfc822</code> – The password must be an rfc822 compliant address. <p>The following two parameters set what happens when the check fails:</p> <ul style="list-style-type: none"> <code>warn</code> – Warn the user, but allow the user to log in. <code>enforce</code> – Warn the user and then log the user out.
<code>path-filter <typelist> <msg> <allowed_charset> {<disallowed regexp> ...}</code>	Defines a regular expression that limits the possible characters used in filenames or paths.

Continued

Table 16-4 (continued)

<i>Permission Capabilities</i>	<i>Function</i>
<pre>upload [absolute relative] [class=<classname>]... [-] <root-dir> <dirglob> <yes no> <owner> <group> <mode> ["dirs" "nodirs"] [<d_mode>]</pre>	<p>Specifies which directories can receive uploads by using <i><dirglob></i>. The absolute and relative options specify if it is an absolute path name or a relative path name to the current directory. If the directory is set to allow uploads with <i><yes no></i>, then they will be owned by <i><owner></i> and <i><group></i> with the permissions of <i><mode></i>.</p>

The `ftphosts` file

The `/etc/ftphosts` file is used to allow or deny access to your ftp server from other systems. The format for this file is as follows:

```
allow <username> <address glob> [<address glob> ...]
```

or

```
deny <username> <address glob> [<address glob> ...]
```

The *<address glob>* can be defined as either a CIDR (Classless Inter-Domain Routing) address or *address:netmask*. CIDR addressing follows the format of *<network address>/<bits in subnet mask>*. For example:

```
10.0.0.0/8
```

The *address:netmask* syntax is as follows:

```
10.0.0.0:255.0.0.0
```

If an `allow` command is used, only users and hosts allowed by it can log in. All others are denied. On the other hand, if the `deny` command is used, any user or host can log in, except for those hosts that are explicitly denied.

The `ftpusers` and `ftpgroups` files

The `/etc/ftpusers` file lists accounts that cannot be accessed via FTP. The standard Red Hat `/etc/ftpusers` contains the following:

```
root
bin
daemon
adm
lp
sync
shutdown
```

```

halt
mail
news
uucp
operator
games
nobody

```

As you can see, these are important accounts such as `root`, or others that there is no reason to use with FTP. If you run only an anonymous FTP server, putting all accounts, except for the anonymous account, in this file would be a good idea.

The `/etc/ftpgroups` file lists groups with a password that FTP users can change into. Usually this provides them with more access to the site, if they know the password. The `private` entry must be set to “yes” in the `/etc/ftpaccess` file for this to work.

The `ftpconversions` file

The `/etc/ftpconversions` file tells the FTP server how to handle compression and archive operations. It lists the filename changes and the procedure to use. Each line in the file has the following format, and Table 16-5 lists the fields:

```

%s:%s:%s:%s:%s:%s:%s:%s
 1  2  3  4  5  6  7  8

```

Table 16-5
ftpconversions Fields

<i>Field Number</i>	<i>Function</i>
1	Strip Prefix (Currently not supported)
2	Strip Postfix: Removes the specified extension from the filename.
3	Addon Prefix (Currently not supported)
4	Addon Postfix: Adds an extension to the filename.
5	External Command: The command used to perform the action.
6	Types: The types of files that are supported by the operation.
7	Options: Options for compress, uncompress, and tar.
8	Descriptions: Describes what the operation does.

The following is an example of the `/etc/ftpconversions` file:

```

.:Z: : :/bin/compress -d -c %s:T_REG|T_ASCII:O_UNCOMPRESS:UNCOMPRESS
: : :.Z:/bin/compress -c %s:T_REG:O_COMPRESS:COMPRESS
.gz: : :/bin/gzip -cd %s:T_REG|T_ASCII:O_UNCOMPRESS:GUNZIP
: : :.gz:/bin/gzip -9 -c %s:T_REG:O_COMPRESS:GZIP
: : :.tar:/bin/tar -c -f - %s:T_REG|T_DIR:O_TAR:TAR
: : :.tar.Z:/bin/tar -c -Z -f - %s:T_REG|T_DIR:O_COMPRESS|O_TAR:TAR+COMPRESS
: : :.tar.gz:/bin/tar -c -z -f - %s:T_REG|T_DIR:O_COMPRESS|O_TAR:TAR+GZIP

```

For example, the first entry enables the uncompress function, by using `compress -d -c`. The `.Z` extension is removed, and no other extension is added.

The FTP log file

The `/var/log/xferlog` file, sometimes found in `/var/adm`, shows file transfers to and from the FTP server. Table 16-6 lists the fields that make up the log entries. The syntax for this log is as follows:

```

current-time  transfer-time  remote-host  file-
size  filename  transfer-type  special-action-
flag  direction  access-mode  username  ser_
vice-name  authentication-method  authenticated-
user-id  completion-status

```

The following is a sample `xferlog` file:

```

Thu Dec 7 20:07:04 2000 1 10.254.90.154 119795 /home/ftp/pub/dhcp-2.0-5.i386.rp
m b _ o a jason@the-nashes.net ftp 0 * c
Thu Dec 7 20:07:14 2000 1 10.254.90.154 340452 /home/ftp/pub/e2fsprogs-1.18-5.i
386.rpm b _ o a jason@the-nashes.net ftp 0 * c
Thu Dec 7 20:07:14 2000 1 10.254.90.154 89800 /home/ftp/pub/e2fsprogs-devel-1.1
8-5.i386.rpm b _ o a jason@the-nashes.net ftp 0 * c

```

Table 16-6
xferlog Fields

<i>Field</i>	<i>Information</i>
current-time	The current time in the format of "DDD MMM dd hh:mm:ss YYYY"
transfer-time	The time in seconds for the transfer to complete
remote-host	The remote hostname
file-size	The size of the file transferred
filename	The name of the file
transfer-type	The type of transfer: a designates ASCII; b designates binary

Field	Information
special-action-flag	Characters that indicate when special action was taken: C indicates the file was compressed; U indicates the file was uncompressed; T indicates the file was tar'd; - indicates that no action was taken.
direction	Direction of the transfer: o indicates an outgoing transfer; i indicates an incoming transfer.
access-mode	The method that the user logged in: a indicates an anonymous login; g indicates the guest account; r indicates a real user
username	The local user name.
service-name	The name of the service invoked, which is usually <i>ftp</i> .
authentication-method	The method of authentication: 0 indicates no authentication; 1 indicates RFC931 authentication.
authenticated-user-id	The user ID returned from the authentication method. A * is used if the user ID is not available.
completion-status	Character that specifies if the transfer was completed: c indicates a completed transfer; i indicates an incomplete transfer.

Using FTP server utilities

You can use several tools to manage the FTP server and monitor its function. These tools are as follows:

- ♦ ftpshut
- ♦ ftpwho
- ♦ ftpcount

The ftpshut command

The `ftpshut` command is used to gracefully shut down the FTP server at a given time while providing warning to users. The format is as follows, and Table 16-7 shows the parameters:

```
ftpshut [ -V ] [ -l min ] [ -d min ] time [ warning-message... ]
```


Table 16-7
ftpshtut Parameters

<i>Parameter</i>	<i>Function</i>
-V	Display version information.
-l <min>	The number of minutes before shutdown that connections are no long accepted. By default this value is 10.
-d <min>	The number of minutes before shutdown that users are disconnected. By default this value is 5.
<i>time</i>	The time at which the server will be shut down. Possible values are the following: <i>now</i> , which shuts the server down immediately; <i>+<number></i> , which specifies the number of minutes from now to shut down the server; and <i>HHMM</i> , which specifies the exact time to shut the server down.
<i>warning-message</i>	The warning message that is displayed to users. You can embed "Magic Cookie" variables that will automatically be replaced by dynamic information by the FTP server. See the man page for a list of the variables.

The ftpwho command

The `ftpwho` command shows the currently connected users to your FTP server. The following shows an example output:

```
[root@redhat ~]# ftpwho
Service class all:
8349 ?      S      0:00 ftpd: 10.254.90.154: anonymous/jason@the-nashes.net:
- 1 users (no maximum)
```

The output displays the following five fields:

- ♦ Process ID
- ♦ The tty connection number, which is always a ? since it isn't used.
- ♦ Status of the connection
- ♦ The amount of CPU time used so far
- ♦ Connection details

The status field is either *R* for running, *S* for sleeping, or *Z* for crashed. The details show who is connected, from where, and the e-mail address entered for anonymous logins.

The ftpcount command

The `ftpcount` command shows the number of users connected, broken down by the class of user dictated in the `ftppaccess` file. It also shows the maximum number of users allowed to connect. For example:

```
[root@redhat ~]# ftpcount
Service class all                - 1 users (no maximum)
```

Configuring Telnet

A Telnet server provides remote terminal access to a system and is normally run from `inetd` using the following entry from `inetd.conf`:

```
telnet stream tcp nowait root /usr/sbin/tcpd in.telnetd
```

Table 16-8 shows the available parameters for `in.telnetd`. If the `/etc/issue.net` file exists, it will be shown to users before they log in to the system.

Table 16-8
in.telnetd Parameters

Parameter	Function
<code>-a <authmode></code>	The type of authentication that the telnet daemon provides. By default, <code>telnetd</code> does not handle authentication, leaving that to <code>login</code> instead.
<code>-D <debugmode></code>	Sets the debug mode. Possible values are as follows: <code>options</code> , which shows information about the negotiation of Telnet options; <code>report</code> , which shows the options information, but also provides more information; <code>netdata</code> , which shows the data stream; and <code>ptydata</code> , which shows the data written to the <code>pty</code> (pseudo terminal).
<code>-edebug</code>	If encryption support was compiled into the Telnet server, this enables encryption debug information.
<code>-h</code>	Disables the display of host-specific information before the user logs in.
<code>-L <Login Program></code>	Specifies a nondefault login program. By default this is <code>/bin/login</code> .
<code>-n</code>	Disables TCP keep-alives.
<code>-s</code>	Used with SecurID authentication cards.
<code>-S <tos></code>	Sets the IP Type-of-Service option.
<code>-X <authtype></code>	Used to disable authentication, if that option was compiled into the server.

Using sendmail

Objective
1.13 Networking Services

- **Operate and perform basic configuration of sendmail.** Modify simple parameters in sendmail config files (modify the DS value for the “Smart Host” if necessary), create mail aliases, manage the mail queue, start and stop sendmail, configure mail forwarding (.forward), perform basic troubleshooting of sendmail. Does not include advanced custom configuration of sendmail. Includes commands mailq, sendmail, and newaliases. Includes aliases and mail/ config files.

A Linux server can be a very powerful and high performance mail server. Most mail servers on the Internet today run the sendmail server. Most Linux distributions include a recent version of sendmail, if only for local mail use. sendmail alone does not give you a complete mail system. Table 16-9 shows the basic components.



More information and documentation for sendmail is available at <http://www.sendmail.org>.

Table 16-9
Mail System Components

<i>Component</i>	<i>Use</i>	<i>Example</i>
MTA (message transfer agent))	Transports messages between servers.	sendmail, Postfix, eximl
MUA (Mail User Agent)	Interfaces the user to the e-mail system.	pine, elm, Outlook Express
MDA (Mail Delivery Agent)	Delivers messages from the server to the user's account.	Procmail

The three main configuration files are shown in Table 16-10.

Table 16-10
sendmail Configuration Files

<i>File</i>	<i>Function</i>
/etc/sendmail.cf	The main configuration file.
/etc/sendmail.cw	Lists the hosts that mail is accepted for.
/etc/aliases	Lists alias addresses.

Customizing the sendmail.cf

The main configuration file for sendmail is `/etc/sendmail.cf`. This file is very big and complex. It was not designed to be easy to customize, but was set up to be easily parsable. An example from the Red Hat `sendmail.cf` is as follows:

```
R< > $+          $: < $H > $1          try hub
R< > $+          $: < $R > $1          try relay
R< > $+          $: < > < $1 $&h >      nope, restore +detail
R< > < $+ + $* > $*    < > < $1 > + $2 $3    find the user part
R< > < $+ > + $*      $#/local $@ $2 $: @ $1      strip the extra +
R< > < $+ >          @$ $1              no +detail
R$+             $: $1 <> $&h          add +detail back in
R$+ <> + $*      $: $1 + $2          check whether +detail
R$+ <> $*        $: $1              else discard
R< local : $* > $*  $: $>95 < local : $1 > $2    no host extension
R< error : $* > $*  $: $>95 < error : $1 > $2      no host extension
R< $- : $+ > $+    $: $>95 < $1 : $2 > $3 < @ $2 >
R< $+ > $+        @$ $>95 < $1 > $2 < @ $1 >
```

This looks pretty scary, and the file is over 1,200 lines long on a default Red Hat system. Most of the options will never need to be changed, but even one typo by accident in a section like that can be disastrous. The only entry in the `sendmail.cf` file that is needed for the exam is the Smart Host entry. In some cases your server may not handle mail delivery itself, but instead may hand mail off to another server, usually at your ISP. The `DS` directive sets the server to send mail to for delivery. For example, to give mail to an ISP's server and have it handle delivery, you would set it as follows:

```
# "Smart" relay host (may be null)
DSisp.server.name.net
```



The Smart Host entry may need to be changed if your ISP actually handles the mail delivery for your system.

To help alleviate these configuration problems, a newer method of configuration has been developed. The new method uses the `m4` preprocessor. This preprocessor lets the administrator write a file that is much easier to understand and then have it converted to the `sendmail.cf` format.



The `m4` preprocessor is used to simplify the configuration of sendmail. A macro file is converted to the final `sendmail.cf`.

This section provides a quick overview of the process, but is not intended to be a complete guide, which would be well outside the scope of this book. A complete list of `m4` macros is given in the `README` file. An `m4` macro has the following syntax:

```
Name(argument1, argument2, argument3 . . .)
```

There are a number of built-in macros, and you can create your own. Table 16-11 shows some of the most common macros used.

Table 16-11
Common m4 Macros

<i>Macro</i>	<i>Function</i>
define	Defines a macro with an argument.
undefined	Undefines a previously set macro.
include	Includes the file specified in the argument.
dnl	Ignores all characters up to the next new line.
divert	Manages output.
feature	Used to enable other features.

The following is an example of the input file used to create the `sendmail.cf` at the beginning of this section.

```
divert(-1)
dnl This is the macro config file used to generate the /etc/sendmail.cf
dnl file. If you modify the file you will have to regenerate the
dnl /etc/sendmail.cf by running this macro config through the m4
dnl preprocessor:
dnl
dnl      m4 /etc/sendmail.mc > /etc/sendmail.cf
dnl
dnl You will need to have the sendmail-cf package installed for this to
dnl work.
include(`../m4/cf.m4')
define(`confDEF_USER_ID',`8:12'')
OSTYPE(`linux')
undefine(`UUCP_RELAY')
undefine(`BITNET_RELAY')
define(`confAUTO_REBUILD')
define(`confTO_CONNECT', `1m')
define(`confTRY_NULL_MX_LIST',true)
define(`confDONT_PROBE_INTERFACES',true)
define(`PROCMAIL_MAILER_PATH',`/usr/bin/procmail')
FEATURE(`smrsh',`/usr/sbin/smrsh')
FEATURE(`mailertable',`hash -o /etc/mail/mailertable')
FEATURE(`virtusertable',`hash -o /etc/mail/virtusertable')
FEATURE(redirect)
FEATURE(always_add_domain)
FEATURE(use_cw_file)
FEATURE(local_procmail)
MAILER(procmail)
MAILER(smtp)
FEATURE(`access_db')
FEATURE(`blacklist_recipients')
```

```

dn1 We strongly recommend to comment this one out if you want to protect
dn1 yourself from spam. However, the laptop and users on computers that do
dn1 not hav 24x7 DNS do need this.
FEATURE(`accept_unresolvable_domains')
dn1 FEATURE(`relay_based_on_MX')
```

This 36-line macro file was turned into the 1,216-line `sendmail.cf` file. The following is the procedure to convert the macro file `redhat.mc` to the final `sendmail.cf` file.

```
m4 redhat.mc > /etc/sendmail.cf
```

Be careful not to overwrite an existing configuration file. The following description goes through the main configuration options of this file to show how it works.

The following line removes extra information from the resulting configuration file:

```
divert(-1)
```

The following line includes the `cf.m4` file:

```
include(`../m4/cf.m4')
```

The next line defines the default user ID:

```
define(`confDEF_USER_ID',``8:12'')
```

The next line specifies the OS type to use. Many parameters, such as path files, are dictated by this option.

```
OSTYPE(`linux')
```

This line indicates that this host is not a UUCP mail relay:

```
undefine(`UUCP_RELAY')
```

This line indicates that this host does not accept BITNET mail:

```
undefine(`BITNET_RELAY')
```

This line indicates that the server should automatically rebuild aliases:

```
define(`confAUTO_REBUILD')
```

The following line sets the connection timeout to 1 minute:

```
define(`confTO_CONNECT', `1m')
```

The following line indicates that if this server is the best MX for a host and hasn't made other arrangements, try connecting to the host directly; normally this would be a config error.

```
define(`confTRY_NULL_MX_LIST',true)
```

The following line states that sendmail will not insert the names and addresses of any local interfaces into the `$=w` class (list of known “equivalent” addresses):

```
define(`confDONT_PROBE_INTERFACES',true)
```

The next line is the path to the procmail program:

```
define(`PROCMAIL_MAILER_PATH',`/usr/bin/procmail')
```

This line tells sendmail to use the sendmail Restricted Shell (smrsh) provided with the distribution instead of `/bin/sh` for mailing to programs:

```
FEATURE(`smrsh',`/usr/sbin/smrsh')
```

This following line defines a table that can be used for customized mail routing:

```
FEATURE(`mailertable',`hash -o /etc/mail/mailertable')
```

The next line defines a domain-specific form of aliasing, allowing multiple virtual domains to be hosted on one machine:

```
FEATURE(`virtusertable',`hash -o /etc/mail/virtusertable')
```

The following line causes sendmail to reject all mail addressed to `address`. REDIRECT with a 551 User not local; please try `<address>` message. If this is set, you can alias people who have left to their new address with `.REDIRECT` appended.

```
FEATURE(redirect)
```

The next line causes sendmail to always append the domain name to addresses, even with local mail.

```
FEATURE(always_add_domain)
```

This line enables the feature to use the local `sendmail.cw` file.

```
FEATURE(use_cw_file)
```

This line causes sendmail to use procmail as the local mail handler.

```
FEATURE(local_procmail)
```

This line is an interface to procmail to be used with a mailer table.

```
MAILER(procmail)
```

The next line enables the SMTP mail server functionality.

```
MAILER(smtp)
```

The following line turns on the access database feature. The access database gives you the ability to allow or to refuse to accept mail from specified domains for administrative reasons.

```
FEATURE(`access_db')
```

The next line enables the ability to block mail based on user names, hostnames, or addresses.

```
FEATURE(`blacklist_recipients')
```

This line causes sendmail to accept mail even if the sending domain name cannot be resolved with DNS.

```
FEATURE(`accept_unresolvable_domains')
```

The `dn1` macro in the following line is used to comment out the `FEATURE` command.

```
dn1 FEATURE(`relay_based_on_MX')
```

Aliasing and forwarding mail

Aliases are useful because they allow mailboxes to have multiple names, because they can forward mail to other addresses if needed, and because they let important mail (such as mail to root) go to an account that is checked more often. Aliases are configured with the `/etc/aliases` file, which has the following syntax:

```
name: name_1, name_2, name_3, . . .
```

Aliases can be recursive so that one alias points to another, different, alias. The following is an example of an `/etc/aliases` file.

```
# Basic system aliases -- these MUST be present.
MAILER-DAEMON: postmaster
postmaster: root

# General redirections for pseudo accounts.
bin: root
daemon: root
games: root
ingres: root
nobody: root
system: root
toor: root
uucp: root
```



```
# Well-known aliases.
manager:      root
dumper:       root
operator:     root

# trap decode to catch security attacks
decode:       root

# Person who should get root's mail
root:         jason, angie

# Custom Aliases
jason:        jason.nash
angie:        angie.nash
jason.nash    jnash
```

Any time that the `/etc/aliases` file is updated you need to run the `newaliases` command, which is the same as running `sendmail -bi`.

```
[root@redhat ~]# newaliases
/etc/aliases: 14 aliases, longest 10 bytes, 152 bytes total
```



You must run the `newaliases` command whenever a new entry is added to `/etc/aliases`.

Aliases must be handled by root. If users want to forward their own mail to another location they use a `.forward` file in their home directory. The format for the `.forward` file is just a list of e-mail addresses to forward the mail to. For example:

```
jason@the-nashes.net
jnash@site.org
"/home/jason/archive"
```

This file would forward any mail received to the first two addresses and then write it to the `/home/jason/archive` file.



`sendmail` checks to make sure that the `.forward` file, or the directory it is in, is writeable only by the user it belongs to. This is a security measure to make sure no one forwards your mail without your knowing.

Managing sendmail

In most cases `sendmail` is not run via `inetd` because it stays resident all of the time. It is started at boot by a script that depends on the distribution. To manage `sendmail` on a Red Hat system you would use the `/etc/rc.d/init.d/sendmail` script, as follows:

```
/etc/rc.d/init.d/sendmail start|restart|stop|status
```

Debian use the `/etc/init.d/sendmail` script, as follows:

```
/etc/init.d/sendmail start|stop|restart|reload|force-reload|debug
```

The `force-reload` command restarts `sendmail` even if it is hung. The `debug` option sends `sendmail` the `USR1` kill signal, which tells the daemon to dump its process information. To view the messages currently in the queue, you use the `mailq` command, or `sendmail -bp`. The first line for each message shows the internal identifier used on this host for the message (TAA08962), the size of the message in bytes (11), the date and time the message was accepted into the queue (Thu Dec 7 19:55), and the envelope sender of the message (`root`). The second line shows the error message that caused this message to be retained in the queue; it will not be present if the message is being processed for the first time. The following lines show message recipients, one per line (`jason@the-nashes.net`). An optional `-v` option increases the amount of information given for some errors. For example:

```
debian:/etc# mailq -v
Mail Queue (2 requests)
--Q-ID-- --Size-- --Priority- ---Q-Time--- -----Sender/Recipient-----
SAA02253      5      30026 Mar 23 18:45 root
              (host map: lookup (#doom.org): deferred)
              jason@#doom.org
SAA02248      5      30025 Mar 23 18:45 root
              (host map: lookup (blah.com): deferred)
              jason@blah.com
```



The `mailq` tool is used to show messages currently waiting in the queue.

Using Apache



1.13 Networking Services

- **Operate and perform basic configuration of apache.** Modify simple parameters in apache config files, start, stop, and restart `httpd`, arrange for automatic restarting of `httpd` upon boot. Does not include advanced custom configuration of apache. Includes managing `httpd` conf files.

The Apache Web server is currently the most popular on the Internet. Every Linux distribution uses it by default, and for good reason. It is free, fast, very configurable, widely supported, and very stable. Apache is based on the original NCSA (National Center for Supercomputing Applications) Web server. So many “patches” were applied to the original source code that it became known as “a patchy” server. Apache may have a corny name, but it is a great Web server!

Apache provides both standard HTTP and secure connections with support for SSL. Other features can be easily added with modules.



Online documentation is available for Apache at www.apache.org.

Starting and stopping httpd

On most systems, Apache is installed and set up to run standalone, not through `inetd`. This is because Apache can take several seconds to start, and this would cause a noticeable delay. Also, if the Web server is even moderately busy, the process of restarting would cause a lot of overhead to the server. If you know that the server will be used very little, you can put the following entry into the `inetd.conf` file to have `inetd` manage Apache.

```
http stream tcp nowait nobody /usr/sbin/httpd /usr/sbin/httpd
```

On most systems the Apache executable, `httpd`, is started from a script. Red Hat systems use the `/etc/rc.d/init.d/httpd` script, while Debian uses `/etc/init.d/apache`. To have the server start automatically, be sure to put a link from the script to the correct runlevel for your system.

Apache provides a control utility called `apachectl` for managing the Web server. Table 16-12 shows the possible controls for `apachectl`.

Table 16-12
apachectl Controls

<i>Command</i>	<i>Function</i>
<code>start</code>	Starts the <code>httpd</code> daemon.
<code>stop</code>	Stops the <code>httpd</code> daemon.
<code>restart</code>	Restarts the <code>httpd</code> daemon if it is already running. If not, it starts the daemon.
<code>fullstatus</code>	Reports the current status of the Web server. This requires the <code>lynx</code> text Web browser, and <code>mod_status</code> compiled into Apache.
<code>graceful</code>	Gracefully restarts the server if running. This lets current connections finish. The daemon is started, if it was not already running.
<code>configtest</code>	Does a configuration syntax test. Useful for sanity checking new configuration files. This makes sure the configuration syntax is correct, not necessarily that it is going to do what you expect.
<code>help</code>	Displays the possible commands.

The `httpd` daemon can be run directly if needed. On most systems it has the name of `httpd`, but on Debian it is `apache`. Table 16-13 shows the available parameters for `httpd`.

Table 16-13
httpd Parameters

<i>Parameter</i>	<i>Function</i>
-D <name>	Defines a name for use in <IfDefine name> directives.
-d <directory>	Specifies an alternative server root.
-f <file>	Specifies an alternative server configuration file.
-C "Directive"	Processes directive before reading the config files.
-c "Directive"	Processes directive after reading the config files.
-v	Shows version information.
-V	Shows compile configuration.
-h	Lists available command-line parameters.
-l	Lists compiled-in modules.
-L	Lists the available configuration directives.
-S	Shows parsed settings.
-t	Runs a syntax check on the configuration file, with a document root check.
-T	Runs a syntax check on the configuration file, without a document root check.

Configuring Apache

Apache was originally designed to use three different configuration files.

- ♦ `httpd.conf`
- ♦ `srm.conf`
- ♦ `access.conf`



The `srm.conf` and `access.conf` files are no longer used in most cases. All configuration should now be done in `httpd.conf`.

These three files were used to provide backward compatibility with the original NCSA server. The `httpd.conf` file was used to configure the server, `srm.conf` was for file types and document specifications, and `access.conf` was used to handle security settings. Current versions of Apache no longer follow that standard, and putting everything in the `httpd.conf` file is now recommended. Apache still installs the other two, but by default they are empty. The following is a part of a sample `httpd.conf`, minus comments.

```
ServerType standalone
ServerRoot /etc/apache
LockFile /var/lock/apache.lock
PidFile /var/run/apache.pid
ScoreBoardFile /var/run/apache.scoreboard
Timeout 300
KeepAlive On
MaxKeepAliveRequests 100
KeepAliveTimeout 15
MinSpareServers 5
...
```

Configuration is done by the use of *directives*, such as `KeepAlive` and `Timeout`. Directives dictate basic information, as shown in the preceding example, or are used to do more complex things such as virtual servers. Depending on the placement of the directives in the configuration file their scope is either server-wide or narrowed down to certain directories. The following sections cover many of the major directives.

Beyond these three configuration files, the configuration can also be fine-tuned by `.htaccess` files. These files are placed in the content directory tree and override settings in the main configuration file. The `.htaccess` file can include the same directives as used in the other files.

Server configuration

Some directives are used to set server-wide options. These should appear in most server configurations. The `ServerAdmin` directive sets the administrator's e-mail address.

```
ServerAdmin Jason@the-nashes.net
```

The `ServerName` directive sets the host name of the server.

```
ServerName www.the-nashes.net
```

The `DocumentRoot` directive sets the root directory for all content shared by the Web server. This is `/home/httpd/html` on Red Hat and `/var/www` on Debian.

```
DocumentRoot /home/httpd/html
```

The `ServerRoot` directive sets the root for the server files. Off this directory are others such as `conf`, `bin`, and `logs`. The Red Hat default is `/etc/httpd` and Debian's is `/etc/apache`.

```
ServerRoot /etc/httpd
```

The `ServerType` directive sets whether the `httpd` daemon is run in standalone mode or managed by `inetd`.

```
Servertype standalone
```

The `MinSpareServers` and `MaxSpareServers` directives manage the number of `httpd` processes that stay resident. Apache tries to maintain as many processes as needed to handle the current load, plus a few extras. If the number of spare servers falls below the minimum, more are started. If there are more maximum spares than defined, some are killed. Tune these settings for the expected load.

```
MinSpareServers 5  
MaxSpareServers 15
```

The `StartServers` directive dictates how many processes to start initially.

```
StartServers 5
```

The `MaxClients` directive defines the upper limit on the number of processes started. If more clients try to connect than the number set, they will be denied access. This is mainly a guard against a runaway server, since most sites do not want to accidentally lock out users. Set this to a level much higher than the number of users you expect.

```
MaxClients 150
```

The `Listen` directive binds the `httpd` daemon to specific IP addresses or ports, in addition to the default IP and port configured with `BindAddress` that is covered:

```
Listen 192.168.10.2:80
```

The `AccessFileName` directive sets the name of the override file, normally `.htaccess`.

```
AccessfileName .htaccess
```

Directory block directives

Apache lets you define some directive blocks so they apply only to certain directory trees. This way some parts of a site can be configured differently than others. The block begins with the `<Directory path>` directive and ends with a `</Directory>` directive. The following is an example.

```
<Directory /Games>
    Options SymLinksIfOwnerMatch
    AllowOverride None
</Directory>
```

Any directives contained in this block apply only to the directory defined. The `Options` directive allows special features to be enabled or configured. The directive in this example allows the Web server to follow symbolic directory links only if the owner matches, for security. The `AllowOverride` directive defines whether the `.htaccess` file can override the options specified in this block.

Access control

With the access control directives of `allow` and `deny`, you can tune which users can access the directories on your site. This example begins with a `<Location /doc>` directive, and ends with a `</Location>` directive. The `Location` directive specifies a certain directory and lets you assign specific directives. The following is an example of these directives:

```
<Location /doc>
    order deny,allow
    deny from all
    allow from 127.0.0.0/255.0.0.0
    Options Indexes FollowSymLinks
</Location>
```

The `order` statement says to deny first and then allow. This causes the server to deny everyone, except those explicitly allowed. The `deny` and `allow` directives show that only users sitting on the system can access the `/doc` directory, since it allows only the loopback network. The `Options` directive enables `Indexes`, which causes the server to send back a nicely formatted directory listing if no default Web page is found. Also enabled in `Options` is `FollowSymLinks`, which lets the server follow symbolic links in the directories.

These directives can be used in other types of blocks, such as `Directory` and `Files`, as well.

Aliases and redirects

The `Alias` directive can point a user to another directory, even if it is on another file system.

```
Alias /icons/ /usr/share/apache/icons/
```

This way the `icons` directory looks to be a subdirectory of the `DocumentRoot` directory, but is actually in `/usr/share/apache/icons`.

The `Redirect` directive lets you point users to a new location for documents. The syntax is as follows:

```
Redirect Old_URL New_URL
```

Default pages

Most users do not put in a complete URL such as `http://www.debian.org/index.html`, instead they only enter `http://www.debian.org`. So how does a site know which page to serve up? This is set with the `DirectoryIndex` directive.

```
DirectoryIndex index.html index.htm index.shtml index.cgi
```

By default the server will look for pages in the listed order and display the first one found.

User Web directories

Many sites provide their users with the ability to create Web sites. This is configured with the `UserDir` directive.

```
UserDir public_html
```

This tells the server to look in the `~/public_html` directory. If someone tried to open `http://www.homepages.com/~jnash`, the server would actually go to `http://www.homepages.com/~jnash/public_html`. This is a good way to let users set up their own pages without worrying about a central location to store them.

MIME types

The `TypesConfig` directive tells the server where the `mime.types` configuration file is stored. This file tells the server which MIME types correspond to which file extensions.

```
TypesConfig /etc/mime.types
```

The `DefaultType` directive tells the server how to treat files with an unknown MIME type.

```
DefaultType text/plain
```

CGI files

CGI (Common Gateway Interface) files are programs or scripts that are executed to provide dynamic content on a site. The `ScriptAlias` directive defines the directory where they are located.

```
ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
```

Special consideration must be paid to CGI files since incorrect permissions or bad code can cause severe security breaches.

Directory browsing

When a user goes to a Web page with no default page, Apache can display a list of files in the directory. To have Apache create a better-looking list with icons for files you can enable the `FancyIndexing` option.

```
IndexOptions FancyIndexing
```

Several other useful `IndexOptions` are available, such as `ScanHTMLTitles`, `IconsAreLinks`, and `FoldersFirst`. `ScanHTMLTitles` reads HTML files and uses the title instead of the filename. `IconsAreLinks` lets users click on the icon and not just the filename. `FoldersFirst` lists folders at the top and files below them.

```
IndexOptions ScanHTMLTitles IconsAreLinks FoldersFirst
```

The following directives let you fine tune the icons used for the listings.

```
DefaultIcon /icons/unknown.gif
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/binhex.gif .hqx
AddIcon /icons/tar.gif .tar
```

The `AddDescription` directive lets you put a description with a file or set of files that match a pattern.

```
AddDescription "GZIP compressed tar archive" .tgz
```

The `HeaderName` and `ReadmeName` directives specify the display of certain text files. The file defined by `HeaderName` is attached to the top of the directory listing, and `ReadmeName` is appended to the bottom.

```
ReadmeName README
HeaderName HEADER
```

Authentication

Apache provides authentication mechanisms that allow you to secure directories or entire sites. The most basic form uses a file that holds user accounts and encrypted passwords. Below is a block example that shows the needed directives.

```
<Directory /home/httpd/html/private>
AuthType Basic
AuthName Private
AuthUserFile /etc/web.users
AuthGroupFile /etc/web.groups
Require valid-user
</Directory>
```

This block causes the `/home/httpd/html/private` to be protected so that only valid users contained in the `web.users` file can get access. When users try to access this directory, they will be prompted for a user name and a password. The `htpasswd` command is used to create this file. Its options are shown in Table 16-14.

Table 16-14
htpasswd Options

<i>Parameter</i>	<i>Function</i>
-c	Creates a new password file.
-n	Displays the results to the console, not to the password file.
-m	Forces MD5 encryption.
-d	Forces Crypt encryption (Default).
-p	Does not encrypt the password.
-s	Forces SHA encryption.
-b	Puts the password on the command-line, does not prompt.

For example, to add a user and create a new password file, you would use the following:

```
europa:~# htpasswd -c /etc/web.users jason
New password:
Re-type new password:
Adding password for user Jason
```

To add a new user to an existing file, you would use the following:

```
europa:~# htpasswd /etc/web.users angie
New password:
Re-type new password:
Adding password for user angie
```

Virtual hosts

Apache provides the ability to support many different Web sites from one server. There are two different ways to set this up. The first method uses a separate IP address for each virtual host, while the second method requires only one address.



IP addresses are now in such short supply, that they are no longer being allocated for use with IP-based virtual hosts.

IP-based virtual hosts

IP-based virtual hosts can be set up to either use one daemon for all hosts or separate domains. To use separate domains you would create a new configuration file for each host and use the `BindTo` and `Listen` directives.

```
BindAddress 192.168.10.10
Listen 192.168.10.10:80
```

These directives instruct the server to listen only on a certain IP address and port number. To use one daemon for multiple hosts, you would use the `VirtualHost` directive.

```
<VirtualHost 192.168.10.10>
  ServerAdmin jason@the-nashes.net
  DocumentRoot /home/httpd/html/nash
  ServerName www.the-nashes.net
  ...
</VirtualHost>
```

You can also use a domain name in the directive.

```
<VirtualHost www.the-nashes.net>
```

Most directives, except those that configure basic Apache functions, can be used inside of the `VirtualHost` directive.

Name-based virtual hosts

Name-based virtual hosts do not require a separate IP address for each host. The client Web browser must support HTTP v1.1 for this to work, as it puts the requested site name in the header. If the client browser does not support this, the user will get the default site set up on the server, which is normally the first one configured.

The directives for named-based virtual hosts are similar to that of IP based, except for the addition of the `NameVirtualHost` directive.

```
NameVirtualHost 192.168.10.11

<VirtualHost 192.168.10.11>
  ServerName www.domain.com
  DocumentRoot /www/domain
</VirtualHost>

<VirtualHost 192.168.10.11>
  ServerName www.otherdomain.com
  DocumentRoot /www/otherdomain
  ServerAlias www site.otherdomain.com
</VirtualHost>
```

The `NameVirtualHost` directive sets the IP address that should be used for the name-based virtual hosts. The `ServerAlias` directive gives other aliases to the virtual server. Users can now reach the site using the names `www` or `site.otherdomain.com`.

Logging directives

Apache provides very customized logging controls. By default, Apache uses the Common Log Format (CLF). The CLF format is made up of the following entries, with each request on a separate line:

```
host ident authuser date request status bytes
```

Table 16-15 shows the meanings of each entry. An example of the log follows:

```
24.163.40.182 - - [04/Jan/2001:11:11:19 -0500] "GET /angie/angie2.thm HTTP/1.1"
404 297
24.163.40.182 - - [04/Jan/2001:11:11:25 -0500] "GET /angie/angie2.htm HTTP/1.1"
200 4089
24.163.40.182 - - [04/Jan/2001:11:11:25 -0500] "GET /cgi-bin/img_counter.pl?test
.txt HTTP/1.1" 200 171
24.163.40.182 - - [04/Jan/2001:11:12:14 -0500] "GET /angie/angie2.htm HTTP/1.1"
200 4090
24.163.40.182 - - [04/Jan/2001:11:12:14 -0500] "GET /angie/1blue511.jpg HTTP/1.1"
304 -
```

Table 16-15
CLF Entries

<i>Entry</i>	<i>Meaning</i>
<i>host</i>	The fully qualified domain name of the requesting host. If that cannot be resolved, the IP address is shown.
<i>ident</i>	If the <code>IdentifyCheck</code> directive is enabled and the client machine runs <code>identd</code> , then this is the identity information reported by the client.
<i>authuser</i>	The user ID of the authenticated user, if enabled.
<i>date</i>	The date of the request.
<i>request</i>	The request line of the client.
<i>status</i>	The three-digit status code returned by the client.
<i>bytes</i>	The number of bytes transferred, not including headers.

The default logging may not capture all of the data you need or may be too verbose. Also, by default all information is logged to one file. The following example shows several logging directives to enable custom logging.

```

LogFormat "%h %l %u %t \"%r\" %s %b"
TransferLog privatelog
CustomLog privatelog "%h %l %u %t \"%r\" %s %b"

```

The `LogFormat` directive changes the default format for logging. Table 16-16 shows the possible options. The `TransferLog` directive outputs the log to a new file, using the default format or the one specified for `LogFormat`. `CustomLog` combines the previous two directives and sets up a new log file and format.

Table 16-16
LogFormat Arguments

Argument	Function
<code>%b</code>	Bytes sent, not counting header information
<code>%f</code>	Filename
<code>%{VAR}e</code>	The value of variable <i>VAR</i> . For example, this can give you information on the client by using the <code>Referer</code> and <code>User-Agent</code> variables.
<code>%h</code>	Remote hostname
<code>%{foobar}i</code>	The contents of <i>foobar</i> : header line(s) in the request sent to the server
<code>%l</code>	Remote logname from <code>identd</code>
<code>%{foobar}n</code>	The value of note <i>foobar</i> from another module
<code>%{foobar}o</code>	The contents of <i>foobar</i> : header line(s) in the reply
<code>%p</code>	The port that the request was served to
<code>%P</code>	The process ID of the child process that handled the request
<code>%r</code>	The first line of the request
<code>%s</code>	Status
<code>%t</code>	The time of the request
<code>%{format}t</code>	The time given in the specified format
<code>%T</code>	The number of seconds taken to handle the request
<code>%u</code>	Remote user, taken from <code>ident</code>
<code>%U</code>	The URL of the path requested
<code>%v</code>	The name of the server that handled the request, used for virtual hosts

Using NFS

Objective
1.13 Networking Services

- **Properly manage the NFS, smb, and nmb daemons.** Mount remote filesystems using NFS, configure NFS for exporting local filesystems, start, stop, and restart the NFS server. Install and configure Samba using the included GUI tools or direct edit of the `/etc/smb.conf` file (Note: this deliberately excludes advanced NT domain issues but includes simple sharing of home directories and printers, as well as correctly setting the `nmbd` as a WINS client).

The Network File System, or NFS, allows you to share out file systems to other networked systems. It was originally developed by Sun Microsystems, but is now supported on all major UNIX systems. NFS is transparent to most users in that they do not even realize the file system is remote, unless a problem occurs. NFS is “stateless,” meaning that no information is lost when an NFS server fails. When the server returns, client services can continue accessing the data like nothing happened.

Linux supports two different NFS servers. One server runs entirely in user space, while the newer server runs as part of the kernel. The kernel server has better performance, but may not be as feature filled as the user-space version. Most Linux distributions currently ship the kernel version as the default. The NFS server requires several processes to be running. Table 16-17 shows these processes. The RPC portmapper service is required, since NFS uses RPC. When clients want to connect to the NFS server, they first query the portmapper service on port 111 to see which port NFS is on.

Table 16-17
NFS Processes

<i>Process</i>	<i>Function</i>
<code>rpc.nfsd</code>	NFS server process.
<code>rpc.mountd</code>	Handles mount requests.
<code>rpc.rquotad</code>	Used to display quota information to clients.
Portmap	Used to map port numbers to RPC processes.
<code>rpc.statd</code>	This is used by the NFS service, <code>rpc.lockd</code> , to manage lock recover when the NFS server crashes or is rebooted.
<code>rpc.lockd</code>	Client process for managing file locking.

Configuring exports

The `/etc/exports` file lists all directories to be shared out by the NFS server. Each line in the file lists a directory to be exported, along with options that apply to that export. A directory can only be exported once. The following is a sample `/etc/exports` file:

```
/pub          (r)   *.the-nashes.net(rw)
/home        192.168.0.0/24(rw)
/documents   norbert(ro,root_squash)
```

On the left is the directory to be exported; on the right are the systems allowed to connect and any permission options applied to them. Wildcard domain names can be used. When a client connects, their IP address is reversed to their hostname and checked against the `/etc/exports` file. Table 16-18 lists the permissions options available.

Table 16-18
Export Options

<i>Option</i>	<i>Function</i>
<code>ro</code>	Export the directory with read-only access.
<code>rw</code>	Export the directory with read and write access (depending on the user's permissions).
<code>rw=<list></code>	Export the directory with read-only access, except for those hosts listed.
<code>root_squash</code>	Remap the root user (UID 0 and GUID 0) to that of the user <code>nobody</code> or the user specified by <code>anonuid</code> and the group specified in <code>anongid</code> . This is the default.
<code>no_root_squash</code>	Do not remap root user access.
<code>all_squash</code>	Remap all users to user <code>nobody</code> .
<code>anonuid=<uid></code>	The user number that root is mapped to with <code>root_squash</code> .
<code>anonguid=<gid></code>	The group number that root is mapped to with <code>root_squash</code> .
<code>secure</code>	Require that the client's originating port be 1023 or less.
<code>insecure</code>	Allow any originating port on the client.
<code>sync</code>	Require that all write requests are committed to disk before being completed.

When a change is made to `/etc/exports`, the `exportfs` command needs to be used to update the server. Table 16-19 lists the options for `exportfs`. The `exportfs` tool can also be used to add and remove individual exports without

editing the `/etc/exports` file. But, these will be lost if the server is rebooted. An example is as follows:

```
exportfs -o no_root_squash norbert:/public
```

This command would share the `/public` directory to the host `norbert` with the `no_root_squash` option.



Any time a change is made to the `/etc/exports` file, the `exportfs -a` command should be run.

Table 16-19
exportfs Options

<i>Option</i>	<i>Function</i>
-a	Alone this option causes the NFS server to export all directories in the <code>/etc/exports</code> file. With the <code>-u</code> option it causes the NFS server to stop exporting all directories.
-o	Specify a list of export options, as done in the <code>/etc/exports</code> file.
-i	Ignore the <code>/etc/exports</code> file. Use only commands given on the command-line of <code>exportfs</code> .
-r	Reexport all directories contained in <code>/etc/exports</code> . If any exports were manually created using <code>exportfs</code> , they are no longer shared.
-u	Unexport one or more directories.
-v	Verbose.

Mounting exported directories

Mounting exported directories from a client is very similar to mounting local file systems. The `mount` command is used with the following syntax:

```
mount host:/directory /mount_point
```

An example would be the following:

```
mount norbert:/home /mnt/norbert
```

This would mount the `/home` directory shared out from the system named `norbert` to the local directory named `/mnt/norbert`. Remote file systems can be mounted at boot with an entry in the `/etc/fstab` file, such as the following:

```
# filesystem      mountpoint      fstype   flags   dump   fsck
norbert:/home     /mnt/norbert   nfs      rw      0      0
```


Some of the more common flags for `fstab` are shown in Table 16-20.

Table 16-20
fstab Flags

Flag	Function
<code>rw</code>	Mounts the file system with read and write operations, but only if it was exported <code>rw</code> .
<code>ro</code>	Mounts the file system read-only.
<code>bg</code>	If a <code>mount</code> command fails, keeps trying in the background.
<code>hard</code>	If an NFS file operation has a major timeout, then reports “server not responding” on the console and continues retrying indefinitely. This is the default.
<code>soft</code>	If an NFS file operation has a major time out, then reports an I/O error to the calling program. The default is to continue retrying NFS file operations indefinitely.
<code>intr</code>	Allows the user to interrupt out of a timeout operation.
<code>nointr</code>	Does not allow for user interrupts.
<code>retrans=<number></code>	The number of timeouts before returning a hard error on a soft mount.
<code>timeo=<number></code>	The number (in tenths of seconds) to wait to retransmit after a timeout.
<code>rsize=<number></code>	Reads buffer size.
<code>wsize=<number></code>	Writes buffer size.
<code>retry=<number></code>	The number of times to retry a mount either in the foreground or background before giving up.

Managing the NFS server

Like most other network services, the NFS server is started via a script. Red Hat uses the following:

```
/etc/rc.d/init.d/nfs start|stop|status|restart|reload
```

while Debian uses the following:

```
/etc/init.d/nfs-kernel-server start|stop|reload|force-reload|restart
```

If the system is only an NFS client, these do not need to be run. Red Hat starts all NFS client services with the following:

```
/etc/rc.d/init.d/netfs start|stop|status|restart|reload
```

and Debian uses the following:

```
/etc/init.d/nfs start|stop|restart
```

The `nfsstat` utility is used to show statistics for the NFS server. Table 16-21 shows the possible parameters. The following is sample output from `nfsstat`.

```
Server rpc stats:
calls      badcalls   badauth    badclnt    xdrcll
8          0          0          0          0
Server nfs v2:
null       getattr    setattr    root        lookup      readlink
0          0% 2      25% 0      0% 0      0% 5      62% 0      0%
read      wrccache   write      create      remove      rename
0          0% 0      0% 0      0% 0      0% 0      0% 0      0%
link      symlink    mkdir      rmdir      readdir     fsstat
0          0% 0      0% 0      0% 0      0% 1      12% 0      0%

Client rpc stats:
calls      retrans    authrefrsh
8          0          0
Client nfs v2:
null       getattr    setattr    root        lookup      readlink
0          0% 2      25% 0      0% 0      0% 5      62% 0      0%
read      wrccache   write      create      remove      rename
0          0% 0      0% 0      0% 0      0% 0      0% 0      0%
link      symlink    mkdir      rmdir      readdir     fsstat
0          0% 0      0% 0      0% 0      0% 1      12% 0      0%
```

Table 16-21
nfsstat Options

<i>Option</i>	<i>Function</i>
-s	Print only server-side statistics.
-c	Print only client-side statistics (currently not supported).
-n	Print only NFS statistics; the default is to also include RPC statistics.
-r	Print only RPC statistics.

Continued

Table 16-21 (continued)

Option	Function
-z	Zero the kernel statistics counters.
-o <facility>	Show only the requested facility. Available facilities are as follows: nfs – NFS protocol information rpc – RPC information net – Network layer statistics and information fh – File handler information rc – Request reply cache information

Security considerations

NFS was not designed with security in mind, and special attention must be paid to secure an NFS system. NFS uses the UID (user ID) and GIDs (group IDs) of the user on the client to grant permissions. The UID is the numeric number that represents a user in the `/etc/passwd` file, and the GID is the numeric number that represents a group in `/etc/group`. The client UID and GUID are matched to the local IDs in `/etc/passwd` and `/etc/group`. The only way to have a secure NFS setup is to make sure all IDs match on all systems. A user should always have the same number. Also consider that users have root access on their workstations, they can change to any ID they want and impersonate any users they want to.

Access by remote systems can be controlled by restrictions in the `/etc/exports` file. As another precaution, you should always block or filter the NFS and RPC ports for any servers exporting directories. Even though you can limit which clients connect, it is not out of the question for a DNS server to be compromised and hostnames or domain names changed to allow unwanted access. Table 16-22 shows the ports that should be blocked.

Table 16-22
NFS Ports

Port	Protocol(s)	Process
111	TCP and UDP	portmapper
745	UDP	mountd
747	TCP	mountd
2049	TCP and UDP	nfsd

Also keep in mind that all data transferred over NFS is sent in clear text. No encryption is used. For these reasons it is advised to use NFS only on a network that is trusted and where users do not have root control on their own workstations.

Using Samba

Objective

1.13 Networking Services

- **Properly manage the NFS, smb, and nmb daemons.** Mount remote filesystems using NFS, configure NFS for exporting local filesystems, start, stop, and restart the NFS server. Install and configure Samba using the included GUI tools or direct edit of the `/etc/smb.conf` file (Note: this deliberately excludes advanced NT domain issues but includes simple sharing of home directories and printers, as well as correctly setting the `nmbd` as a WINS client).

Samba is a very popular suite that allows Linux, and other UNIX computers to share network resources with Windows PCs. Samba can be configured to provide the following:

- ♦ File sharing
- ♦ Printer sharing
- ♦ User authentication
- ♦ Name resolution (WINS client and server)
- ♦ Service announcement (browsing)

Samba provides its services using two daemons, `smbd`, server message block daemon, and `nmbd`, name message block daemon. File and print services, as well as user authentication, are provided by `smbd`. The `nmbd` daemon handles name resolution and service announcement and browsing.



More information and documentation on Samba is available at www.samba.org.

Configuring Samba

Samba is configured with a well-documented file called `/etc/smb.conf`. This file is made up of many configuration entries. Some are optional and some are not. The following is an example `/etc/smb.conf` file.

```
#===== Global Settings =====
[global]

# workgroup = NT-Domain-Name or Workgroup-Name
# workgroup = NASH

# server string is the equivalent of the NT Description field
# server string = Linux Samba Server
```

```

# This option is important for security. It allows you to restrict
# connections to machines which are on your local network.
  hosts allow = 192.168.1. 127.

# if you want to automatically load your printer list rather
# than setting them up individually then you'll need this
  printcap name = /etc/printcap
  load printers = yes

# this tells Samba to use a separate log file for each machine
# that connects
  log file = /var/log/samba/log.%m

# Put a capping on the size of the log files (in Kb).
  max log size = 50

# Security mode. Most people will want user level security. See
# security_level.txt for details.
  security = user

# Most people will find that this option gives better performance.
# See speed.txt and the manual pages for details
  socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192

# Browser Control Options:
# set local master to no if you don't want Samba to become a master
# browser on your network. Otherwise the normal election rules apply
  local master = no

# WINS Server - Tells the NMBD components of Samba to be a WINS Client
# Note: Samba can be either a WINS Server, or a WINS Client, but NOT both
  wins server = 192.168.1.12

# DNS Proxy - tells Samba whether or not to try to resolve NetBIOS names
# via DNS nslookups. The built-in default for versions 1.9.17 is yes,
# this has been changed in version 1.9.18 to no.
  dns proxy = no

```

The next section sets up the sharing of directories. These start with the name of the share in brackets, [homes] for example. An optional comment line sets the displayed comment when the network is browsed. The browseable configuration option specifies whether this share will show up in Network Neighborhood on a Microsoft network. A path command sets the directory to be shared out.

```

#Share Definitions

[documents]
  comment = Our Documents
  browseable = yes
  writable = yes
  path = /var/docs

```

Next, the example shows how to share a printer. The `path` variable points to the spool directory. The `guest ok` command either enables or disables guest access. The directory cannot be written to because of the `writable` command, but it is used for printing because of the `printable` command.

```
# NOTE: If you have a BSD-style print system there is no need to
# specifically define each individual printer
[printers]
    comment = All Printers
    path = /var/spool/samba
    browseable = no
# Set public = yes to allow user 'guest account' to print
    guest ok = no
    writable = no
    printable = yes
```

Table 16-23 shows many of the configuration options in the `smb.conf` file.

Table 16-23
Common `smb.conf` Entries

Entry	Function
<code>workgroup = <name></code>	Workgroup or Windows NT domain name the system belongs to.
<code>server string = <description></code>	The description field that shows up when Windows clients browse.
<code>hosts allow = <hosts></code>	A list of hosts that are allowed to connect to Samba shares. You can either specify the hosts or use a wildcard such as <code>4.18.</code> , which specifies any host in the <code>4.18.0.0</code> network. The use of the <code>EXCEPT</code> keyword is also allowed to restrict certain hosts.
<code>guest account = <account></code>	The local guest account that permissions are applied to.
<code>security = <type></code>	The type of security the system uses. Can be set to <code>share</code> , <code>user</code> , <code>server</code> , and <code>domain</code> .
<code>password server = <server name></code>	If the server security type is selected, this server is checked for authentication.
<code>encrypt passwords = yes/no</code>	Some configurations require passwords to be encrypted. An incorrect setting will cause connections to fail.

Continued

Table 16-23 (continued)

<i>Entry</i>	<i>Function</i>
<code>username map = <file></code>	Remote user names can be mapped to local Linux users.
<code>interfaces = <ip addresses></code>	Which network interfaces to use.
<code>local master = yes/no</code>	Enables or disables the Linux system from becoming a Master Browser on the network.
<code>domain master = yes/no</code>	Allows the Linux system to be a Domain Master Browser for the network. Do not use this if you have a Windows NT Domain Controller.
<code>preferred master = yes/no</code>	Enables preferred Master Browser status in elections.
<code>domain controller = <DC name></code>	The name of a local Windows NT domain controller.
<code>wins support = yes/no</code>	Enables or disables WINS server functionality.
<code>wins server = <ip address></code>	Enables WINS client functionality.



The `wins server` command in `smb.conf` enables the name resolution client in `nmbd`.

The bottom of the `/etc/smb.conf` file lists the shared directories, along with options such as `browsable` and `comment`.

Managing Samba

Samba is started by a script, and is usually not run via `inetd`, though that is the default on Debian. Red Hat uses the following script:

```
/etc/rc.d/init.d/smb start/stop/restart/status
```

Debian uses the following:

```
/etc/init.d/samba start/stop/reload/restart/force-reload
```

The Samba suite includes a tool, `testparm`, to check your configuration file for syntax errors. The following is an example output from `testparm`.

```
[root@redhat ~]# testparm
Load smb config files from /etc/smb.conf
Processing section "[homes]"
Processing section "[printers]"
Loaded services file OK.
```

Press enter to see a dump of your service definitions

```
# Global parameters
[global]
    workgroup = MYGROUP
    netbios name =
    netbios aliases =
    server string = Samba Server
    interfaces =
...

```

Note that `testparm` checks only for syntax errors. Depending on your network setup and specific configuration, you may still have connectivity problems.

To monitor connections to a Samba server you can use the `smbstatus` command.

```
[root@redhat ~]# smbstatus

Samba version 2.0.6
Service      uid          gid          pid          machine
-----
jason        jason       jason       11436        compaq      (10.254.90.154)
Sat Dec  9 14:45:15 2000

No locked files

Share mode memory usage (bytes):
  1048464(99%) free + 56(0%) used + 56(0%) overhead =
 1048576(100%) total

```

Client connections

Samba provides a tool to let users connect to other Samba or Windows shares on the network. The `smbclient` tool uses an FTP-like command syntax to navigate the server. Table 16-24 lists the command-line options for `smbclient`.

Table 16-24
Command-Line `smbclient` Parameters

<i>Parameter</i>	<i>Function</i>
<code>-s <file></code>	Path name to the <code>smb.conf</code> file.
<code>-O <options></code>	Socket options to use.
<code>-R <services></code>	Use only the specified name resolution services.
<code>-M <host></code>	Send a WinPopUp message to the host specified.

Continued

Table 16-24 (continued)

Parameter	Function
-i <scope>	Use the specified NetBIOS scope.
-N	Do not ask for a password.
-n <name>	Use the specified NetBIOS name for this system.
-d <level>	Set the debugging level.
-P	Connect to the service requested as a printer. This may not be needed with Samba 2.0.
-p <port>	Manually define the port to connect to.
-l <log basename>	Specify the log basename.
-h	List the available command-line options.
-I <ip address>	Connect to the specified IP address.
-E	Output messages to <code>stderr</code> instead of <code>stdout</code> .
-U <username>	Define the network user name.
-L <host>	Retrieve a list of shares from the specified host.
-t <code>	Terminal ID code.
-m <max protocol>	Set the maximum protocol level.
-W <workgroup name>	Set the Windows workgroup name.
-T<option>	Command-line tar.
-D <directory>	Start the connection from the specified directory.
-c <command string>	Execute the specified commands. Multiple commands can be specified, separated by semicolons.
-b	Change the transmit or send buffer size.

The following is an example of a connection with `smbclient`. Table 16-25 lists the commands that can be used inside of `smbclient` during a connection.

```
[root@redhat ~]# smbclient //nash-pdc/c$ -U jason
added interface ip=10.254.90.155 bcast=10.254.90.159 nmask=255.255.255.224
Password:
Domain=[NASH-DOMAIN] OS=[Windows 5.0] Server=[Windows 2000 LAN Manager]
smb: \> ls
AUTOEXEC.BAT                H           0 Thu May  4 18:57:35 2000
boot.ini                    HS          265 Thu May  4 18:20:38 2000
CONFIG.SYS                  H           0 Thu May  4 18:57:35 2000
Documents and Settings      DA           0 Sun Sep 26 21:45:20 1999
drwtsn32.log                 A 1034425   Fri Sep 24 22:27:17 1999
Exchange Server Setup.log    A 2206836   Fri Sep 24 22:56:46 1999
```

```

EXCHSRVR                D            0 Mon Sep 27 00:38:20 1999
font                    D            0 Thu Aug 10 15:41:55 2000
InetPub                 D            0 Thu Nov 30 22:35:17 2000
IO.SYS                  AHSR         0 Thu Aug 19 16:18:18 1999
MSDOS.SYS               AHSR         0 Thu Aug 19 16:18:18 1999
Multimedia Files       D            0 Fri Aug 20 14:45:28 1999
NTDETECT.COM           AHSR    34468 Tue Dec  7 07:00:00 1999
ntldr                  AHSR   214416 Tue Dec  7 07:00:00 1999
pagefile.sys           AHS 402653184 Tue Oct 17 21:28:04 2000
Program Files          DAR            0 Mon Oct 30 09:27:29 2000
RECYCLER               DHS            0 Mon Jan 31 12:45:57 2000
SMSSETUP.LOG           A            240 Sun Sep 26 22:16:10 1999
System Volume Information DHS            0 Fri Oct  1 23:47:13 1999
TEMP                   DA            0 Fri Jul  7 17:47:29 2000
WINNT                  DA            0 Sun Oct  1 14:55:56 2000

```

51834 blocks of size 131072. 27845 blocks available

smb: \>

Table 16-25
smbclient Commands

Command	Function
? <command>	If a <command> is specified, help will be shown for that command; if not, general help is displayed.
! <shell command>	If a <shell command> is specified, that command is run; if not, the user is dropped to a shell.
cd <directory name>	Changes to the specified directory.
del <mask>	Deletes files or directories.
dir <mask>	Displays files or directories.
exit	Exits the smbclient tool.
get <remote file name> [local file name]	Retrieves the specified file.
help <command>	This is the same as ? <command>.
lcd <directory name>	Changes the local directory. If no <directory name> is specified, the current local directory is displayed.
ls <mask>	This is the same as dir <mask>.
mask <mask>	This defines a mask that will be used with the mget and mput commands.
md <directory name>	Creates a new directory.
mget <mask>	Retrieves the files that match the <mask>.

Continued

Table 16-25 (continued)

Command	Function
<code>mkdir <directory name></code>	This is the same as <code>md <directory name></code> .
<code>mput <mask></code>	Copies all files matching the <code><mask></code> to the remote system.
<code>print <file name></code>	Prints a local file to a print device on the server.
<code>printmode <graphics text></code>	Sets the print mode to graphics or text.
<code>prompt</code>	Toggles prompting for <code>mget</code> and <code>mput</code> .
<code>put <local file name> [remote file name]</code>	Copies a local file to the server.
<code>queue</code>	Displays the print queue.
<code>quit</code>	This is the same as <code>exit</code> .
<code>rd <directory name></code>	Removes the specified directory.
<code>recurse</code>	Toggles directory recursion for <code>mput</code> and <code>mget</code> .
<code>rm <mask></code>	Deletes all files matching <code><mask></code> .
<code>rmdir <directory name></code>	This is the same as <code>rd <directory name></code> .
<code>tar</code>	Performs <code>tar</code> operations.
<code>blocksize <number></code>	Sets the <code>tar</code> block size.
<code>tarmode <full inc reset noreset></code>	Changes how <code>tar</code> manages the archive bit.
<code>setmode <filename> <perm=+ -rsha></code>	Sets the file attributes on the remote server. Similar to the DOS attribute command.

Remote Windows and Samba shares can be mounted as a local file system, which is sometimes more convenient than using `smbclient`. The syntax for this is as follows:

```
mount -t smbfs -o username=<username> //server/share
/mount_point
```

For this to work, the `smb` file system support must be compiled into the kernel.

Using DNS

Objective**1.13 Networking Services**

- **Setup and configure basic DNS services.** Configure hostname lookups by maintaining the `/etc/hosts`, `/etc/resolv.conf`, `/etc/host.conf`, and `/etc/nsswitch.conf` files, troubleshoot problems with local caching-only name server. Requires an understanding of the domain registration and DNS translation process. Requires understanding key differences in config files for bind 4 and bind 8. Includes commands `nslookup`, `host`. Files: `named.boot` (v.4) or `named.conf` (v.8)

Most people know Internet sites by their easy-to-remember names, such as `http://www.hungryminds.com`. However, packets can only be directed to an IP address, not to a name. Several ways to convert the easy-to-remember names to the IP addresses that routers and systems use exist. The most common two ways are by using a hard-coded file such as `/etc/hosts` and by using *domain name system*, or DNS.

Overview of DNS

Before the advent of DNS, when the Internet was much smaller than today, hostnames and IP addresses were kept in a static file that an administrator had to update periodically. This caused problems with out-of-date files, and once hosts were created at a faster rate, the job of updating this file became unbearable. Obviously, a better solution was needed. DNS provides a hierarchical name space for hosts and IP addresses. This tree is made up of a distributed database of information. This way an administrator is responsible only for the systems under his or her control. No central repository of information exists in this structure.

DNS is a client/server system. Usually a few systems are designated to be DNS servers, while the rest are clients. The servers handle queries from the local network and from the Internet. A small office may not even have a DNS server; they may have their ISP provide it for them. Large companies may have many DNS servers spread throughout the enterprise to provide better service to their users.

The DNS namespace

The DNS namespace is made up of a tree of domain names, starting at the top with the root called “.”. Figure 16-1 shows an example of this tree structure. Beneath the root are the *top-level domains*, or TLDs. These are also known as root-level domains. Many countries use a two-letter country code as their top-level domain. This is available in the United States as well, but is not normally used. Some countries use a combination, such as `.com.tw` for a company in Taiwan, though the domains are not always categorized the same in other countries as they are in the U.S. For example, an educational institution in Japan is `.ac.jp` instead of `.edu.jp`.

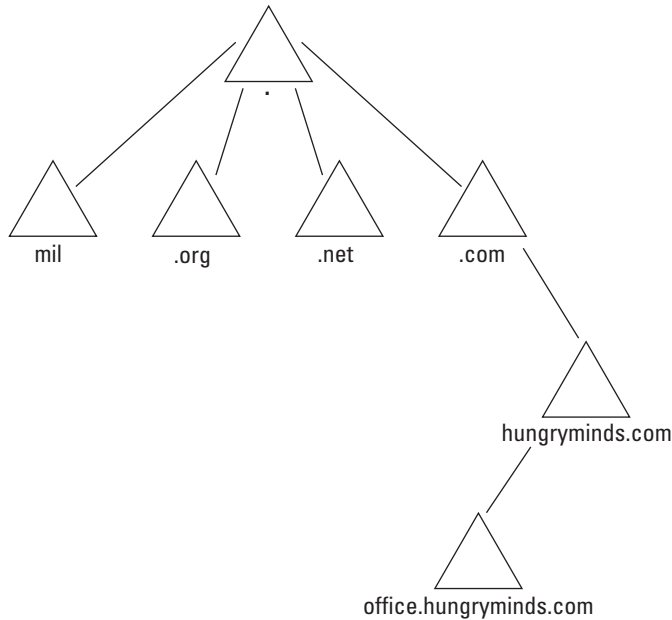


Figure 16-1: DNS namespace

Organizations, individuals, or companies register the second-level domain names. At one time, all domains had to be registered from one registrar known as InterNIC, but that is no longer the case. The DNS for the second-level domain is handled by the organization that registered it. This way the organization can make any changes to its own namespace without the need to update any other information. For example, Hungry Minds, Inc., can set up any subdomains or hostnames for systems in the hungryminds.com domain. It does not need to consult with anyone else to make these changes, because it has authority over its piece of the namespace.

A fully qualified domain name is made up of a system's hostname along with any subdomains and domains. For example, `norbert.office.the-nashes.net` is a fully qualified domain name. `Norbert` is a hostname, `office` is a subdomain, `the-nashes` is a second-level domain, and `.net` is the top-level domain. To specify a fully qualified domain name in most DNS configuration files, the name should also end in a period. For example:

```
www.hungryminds.com.
```

This final dot represents the root domain. It is not normally used in client applications because it is assumed and appended by the system. If a name does not end in a period inside of a configuration file, the name is assumed to be a relative name to the current domain. For example, if you only entered `www.hungryminds.com` in a configuration file, the DNS server would consider the name to be the following:

```
www.hungryminds.com.hungryminds.com.
```

If you get responses such as these, check for a missing dot.

So how does all of this work? When a client needs to resolve a name to an IP address, it consults the DNS server it is configured to use. If the requested name is part of the domain handled by that DNS server, the server returns the answer directly. If the DNS server does not handle the requested domain, the server then checks its cache information to see if it knows which server may have the answer. This would happen if another host at the requested domain had been looked up before. The local DNS server would already know the remote domain's DNS server and could just query it. If the cache does not help, it queries the root name servers, who may then forward the local DNS server to a server that handles the TLD of the requested domain. The TLD server can then forward the client's DNS server to the next closest server in its search. The client's DNS server keeps querying through the tree until the answer is found. Figure 16-2 shows this process. While this may sound like a lot of overhead, caching cuts down on the amount of traffic since information is stored for a set amount of time. If the information needed is already cached, the server will not need to go through all of the steps again. Also, this tree structure allows each administrator to control his or her own namespace.

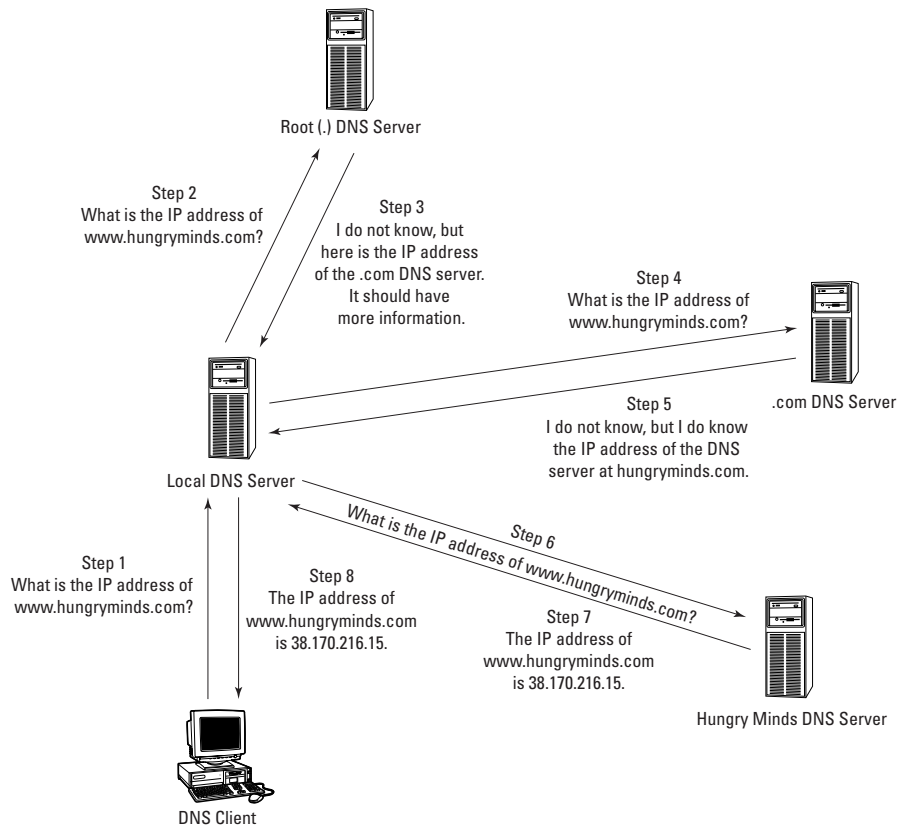


Figure 16-2: Client query

DNS and BIND

Many people use the terms DNS and BIND interchangeably, while this is not really correct. DNS is to BIND as HTTP is to Apache. BIND, or Berkeley Internet Name Domain, is the software package that implements DNS. It was originally developed at the University of California in Berkeley, but is now maintained by the Internet Software Consortium (ISC). Several versions of BIND are being used today. The newest is v9, though it is not widely used yet. The most popular is v8. The oldest you will encounter is v4. To see which version a DNS server is running you can use the `dig` command, which is used to query DNS servers for information. This may not work if the administrator has disabled giving out this information:

```
[root@redhat ~]# dig @192.168.10.10 version.bind txt chaos

; <<>> DiG 8.2 <<>> @192.168.10.10 version.bind txt chaos
; (1 server found)
;; res options: init recurs defnam dnsrch
;; got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 6
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0,
ADDITIONAL: 0
;; QUERY SECTION:
;;      version.bind, type = TXT, class = CHAOS

;; ANSWER SECTION:
VERSION.BIND.      0S CHAOS TXT      "8.2.2-P7"

;; Total query time: 20 msec
;; FROM: redhat.the-nashes.net to SERVER: 192.168.10.10
;; WHEN: Sat Dec  9 15:40:19 2000
;; MSG SIZE  sent: 30  rcvd: 63
```

This example shows a server running BIND v8.2.2-P7. The BIND daemon is called `named`. The older v4 used different configuration syntax than the newer versions use. In the BIND suite along with `named` are sample configuration files and resolver libraries that let client applications perform queries.

Configuring BIND v8

BIND v8 uses the `named.conf` configuration file, along with several other *zone files* that hold address-mapping information. The format of `named.conf` is made up of a series of statements that each end with a semicolon. Curly brackets are sometimes used to group lines together. While the syntax is easy to read and use, a misplaced semicolon can cause serious problems that are sometimes hard to track down. A zone file is a list of resource records that are the real information in DNS. A zone is the piece of the DNS namespace that a server is responsible for. This is different than a domain because a zone may contain more than a single domain or subdomain. Multiple DNS servers can be set up to act as a backup, or secondary, to a master server. The secondary servers pull information from the master, in what is known as a *zone transfer*. The following is an example `named.conf` file.



BIND v8 uses the `named.conf` configuration file.

```
options {
    directory "/";
    named-xfer "/named-xfer";

    //directory "/var/named";
    //named-xfer "/var/named/named-xfer";

    // Don't reveal BIND version
    version "";
};

logging {
    category lame-servers { null; };
    category cname { null; };
};

// ndc control socket
controls {
    unix "/var/run/ndc" perm 0600 owner 0 group 0;
};
// Standard zones
//

zone "the-nashes.net" IN {
    type master;
    file "db.the-nashes";
};

zone "localhost" IN {
    type master;
    file "named.localhost";
    allow-transfer { localhost; };
};

zone "0.0.127.in-addr.arpa" IN {
    type master;
    file "named.loopback";
    allow-transfer { localhost; };
};

zone "." IN {
    type hint;
    file "named.root";
    type master;
    file "named.loopback";
    allow-transfer { localhost; };
};

zone "." IN {
    type hint;
```



```

        file "named.root";
    };

    // Slave zones
    //
    zone "otherzone.net" IN {
        type slave;
        file "slave/otherzone.net";
        masters { 192.168.1.10; [...]; };
    };

```

This configuration file sets up the domain of the `the-nashes.net` and a slave domain of `otherzone.net`. A slave domain is one where the DNS server retrieves the zone file from another server. The slave server holds a read-only copy of the information.

named.conf statements

The next sections cover the statements that can be used in the `named.conf` file.

The include statement

The `include` statement brings in another configuration file to be added to the main `named.conf`. This is useful for large sites with departmental control. Each department can maintain their own DNS information in a file on a server, and this statement pulls that information into the DNS configuration. The syntax is as follows:

```
include "path"
```

The options statement

The `options` statement specifies global configuration settings. There are over 50 possible options settings in BIND v8. The syntax is as follows:

```
options {
    option statement;
    option statement;
    ...
}
```

Table 16-26 lists some of the common `options` statements. Many of the `options` statements use address lists for security. The following are some examples of these lists.

```
{192.168.10/24; 192.168.12/24; 10.10/16; 127.0.0.1}
{! 192.168.10.15; 192.168.10/24}
```

The first example shows several network addresses, and one single host (local-host). The second example has one denied host, designated with the `!`, and a network that is allowed. The order is very important in the second example. BIND stops searching after it finds the first match. If the order were reversed, the address of `192.168.10.15` would match `192.168.10/24`, and BIND would never get to the entry limiting the address.

Table 16-26
options Statements

Statement	Function
<code>version "string";</code>	Change the version number reported to clients. Some people consider this more secure.
<code>directory "path";</code>	The full path of the working directory of the server. Other paths in the configuration file may be relative paths to this directory.
<code>notify yes no;</code>	If this server is a master server for others, notify the slave servers when a change is made to a zone database.
<code>also-notify <server address> ...;</code>	Also notify other servers not listed in your configuration file. Useful for sites that have separate internal servers that are not advertised to the world.
<code>recursion yes no;</code>	By default the DNS server will check recursively through the namespace to find an answer to a client's query. This statement can disable this function.
<code>maintain-ixfr-base yes no;</code>	Send changes only to other servers, not the entire database, during zone transfers.
<code>transfers-in <number>;</code>	Limit the number of inbound zone transfers at one time.
<code>transfers-out <number>;</code>	Limit the number of outbound zone transfers at one time.
<code>forwarders {ip addresses};</code>	A list of servers that queries are forwarded to, instead of being resolved by this server.
<code>forward only first;</code>	If set to <code>only</code> , this server will only forward requests to other servers and not resolve them itself. If set to <code>first</code> , the server tries to forward the query, but if that fails, it will resolve the query itself.
<code>allow-query {address match list};</code>	Limit the clients that can query your DNS server.
<code>allow-transfer {address match list};</code>	Limit the systems that can request zone transfers from your server.
<code>blackhole {address match list};</code>	Systems in this list cannot communicate with your DNS server.

The `acl` statement

The *access control list* (ACL) statement must be at the top of the configuration file, but the `name_of_acl` created can be used anywhere that an address match list is used. The syntax is as follows:

```
acl name_of_acl {
    Address_match_list
};
```

The `acl` statement should be at the top of your configuration file. The ACL name defined in the block cannot be used before the `acl` block is encountered. Table 16-27 lists four predefined ACL lists.

Table 16-27
Predefined ACL Lists

<i>List</i>	<i>Members</i>
Any	Allow all hosts.
None	Deny all hosts.
Localnets	Allow local networks off any interface on the system.
Localhost	Allow the IP of any interface on the system.

The `server` statement

The `server` statement lets you define certain characteristics for remote servers. The possible characteristic settings are shown in Table 16-28. The syntax is as follows:

```
server ip_addr {
    [ bogus yes_or_no; ]
    [ support-ixfr yes_or_no; ]
    [ transfers number; ]
    [ transfer-format ( one-answer | many-answers ); ]
    [ keys { key_id [key_id ... ] }; ]
};
```

Table 16-28
server Statements

<i>Statement</i>	<i>Function</i>
<code>bogus yes no;</code>	If set to <code>yes</code> , no queries are sent to the server. This can be useful to temporarily stop queries going to a server that is giving out incorrect information.

Statement	Function
<code>support-ixfr yes no;</code>	Enable or disable incremental zone transfers.
<code>transfers <number>;</code>	Limit the number of inbound zone transfers from the server.
<code>transfer-format one-answer many-answers;</code>	Do zone transfers one record at a time or many. The <code>many-answers</code> option is supported only by BIND 8.1 and patched versions of 4.9.5.
<code>keys { key_id...};</code>	Used with secure transfers.

The logging statement

BIND has some of the most customizable logging options of any application. The syntax is as follows:

```
logging {
  [ channel channel_name {
    ( file path_name
      [ versions ( number | unlimited ) ]
      [ size size_spec ]
      | syslog ( kern | user | mail | daemon | auth | syslog | lpr |
                news | uucp | cron | authpriv | ftp |
                local0 | local1 | local2 | local3 |
                local4 | local5 | local6 | local7 )
      | null );
    [ severity ( critical | error | warning | notice |
                info | debug [ level ] | dynamic ); ]
    [ print-category yes_or_no; ]
    [ print-severity yes_or_no; ]
    [ print-time yes_or_no; ]
  }; ]

  [ category category_name {
    channel_name; [ channel_name; ... ]
  }; ]
  ...
};
```

The `channel` block defines where information is sent to, whether it be to a `syslog` facility or a separate file. The `category` block defines exactly which information is logged, so you see only what you want or need. The BIND documentation covers logging in detail.

The zone statement

The `zone` blocks are the real heart of the information in the `named.conf` file. These blocks tell the server which domains, or zones, the server is responsible for and what role it plays. The format of the `zone` block depends on the function of the server. Table 16-29 shows the different zone types.

Table 16-29
Zone Types

Zone Type	Function
master	This server contains the master zone information and provides authoritative answers.
slave	A replica of a master zone. The server holds a read-only copy of the zone information.
stub	Similar to a slave server, but replicates only name server (NS) records.
forward	Directs all queries for this zone to another server.
hint	Used to locate the root name servers by using a <code>hint</code> file.

Master zones keep the resource records that hold DNS information on a file on the disk, called a *zone file*. The following is the format for a master zone.

```
zone domain_name [ ( in | hs | hesiod | chaos ) ] {
  type master;
  file path_name;
  [ forward ( only | first ); ]
  [ forwarders { [ ip_addr ; [ ip_addr ; ... ] ] }; ]
  [ check-names ( warn | fail | ignore ); ]
  [ allow-update { address_match_list }; ]
  [ allow-query { address_match_list }; ]
  [ allow-transfer { address_match_list }; ]
  [ dialup yes_or_no; ]
  [ notify yes_or_no; ]
  [ also-notify { ip_addr; [ ip_addr; ... ] }; ]
  [ ixfr-base path_name; ]
  [ pubkey number number number string; ]
};
```

Below is the format for a slave or stub zone:

```
zone domain_name [ ( in | hs | hesiod | chaos ) ] {
  type ( slave | stub );
  [ file path_name; ]
  [ ixfr-base path_name; ]
  masters [ port ip_port ] { ip_addr; [ ip_addr; ... ] };
  [ forward ( only | first ); ]
  [ forwarders { [ ip_addr ; [ ip_addr ; ... ] ] }; ]
  [ check-names ( warn | fail | ignore ); ]
  [ allow-update { address_match_list }; ]
  [ allow-query { address_match_list }; ]
  [ allow-transfer { address_match_list }; ]
  [ transfer-source ip_addr; ]
  [ dialup yes_or_no; ]
```

```
[ max-transfer-time-in number; ]
[ notify yes_or_no; ]
[ also-notify { ip_addr; [ ip_addr; ... ] }; ]
[ pubkey number number number string; ]
};
```

Below is the format for a forward zone:

```
zone domain_name [ ( in | hs | hesiod | chaos ) ] {
    type forward;
    [ forward ( only | first ); ]
    [ forwarders { [ ip_addr ; [ ip_addr ; ... ] ] }; ]
    [ check-names ( warn | fail | ignore ); ]
};
```

A hint zone is used to bootstrap the DNS server. The only information the server needs to resolve client queries is where to go for its next step, which is the root name server. The server consults a hint file that lists the IP addresses of the root name server. The BIND package includes this file, but it should be periodically updated in case a server is moved.

```
zone "." [ ( in | hs | hesiod | chaos ) ] {
    type hint;
    file path_name;
    [ check-names ( warn | fail | ignore ); ]
};
```

Table 16-30 shows the options for a zone file.

Table 16-30
zone Options

Option	Function
check-names warn fail ignore;	Checks to see if the name to be resolved is a legitimate name.
Allow-query {address match list};	Controls which systems are allowed to submit queries.
Allow-transfer {address match list};	Controls which systems can request a zone transfer.
Transfer-source <ip address>'	Controls which local address will be used as the source for zone transfers.
Ixfr-base "path";	Incremental transfer transactional log name.

Continued

Table 16-30 (continued)

<i>Option</i>	<i>Function</i>
<code>max-transfer-time-in <number>;</code>	If an inbound zone transfer does not complete in <number> minutes, it will be disconnected.
<code>dialup yes no;</code>	Specifies whether zone transfers will be on a dial-up connection.
<code>notify yes no;</code>	Specifies whether or not to notify other servers when a change is made.
<code>also-notify {ip address...};</code>	Lists other servers to notify of changes.
<code>forward first only;</code>	Specifies whether to forward queries to another server first or always.
<code>forwarders {ip addresses...};</code>	List of servers to forward queries to, if enabled.
<code>pubkey</code>	Public encryption key for this zone. This is used for secure updates.

The controls statement

BIND provides a tool named `ndc` to help manage the `named` daemon. The `controls` statement configures how this tool controls the server. The syntax is as follows:

```
controls {
  [ inet ip_addr
    port ip_port
    allow { address_match_list; }; ]
  [ unix path_name
    perm number
    owner number
    group number; ]
};
```

Without proper configuration, anyone on the Internet may be able to do bad things to your DNS server such as change configuration or disable it. The `inet` line specifies an IP address and port number to listen to. If this is used, you must be careful to properly set up the `allow` statement so that only connections from trusted hosts are allowed. Anyone from an allowed host can Telnet to the opened port and stop the DNS server, no login required. If used, connections are usually allowed only from 127.0.0.1, and only if no other users use the server.

The `unix` statement allows the `/var/run/ndc` socket to be used for controlling the daemon. The `perm`, `owner`, and `group` statements set the permissions on the socket and, therefore, limit the users that can access it.


```

; Aliases
;
mail                               IN CNAME brain.the-nashes.net.

```

Table 16-31
Common Record Types

<i>Type</i>	<i>Name</i>	<i>Function</i>
SOA	Start of Authority	Defines the zone of authority. This dictates which part of the namespace this zone is responsible for.
NS	Name Server	Identifies name servers and delegates subdomains.
A	Address	Maps a name to an IP address.
PTR	Pointer	Maps an IP address to a hostname.
MX	Mail Exchanger	Lists servers responsible for receiving e-mail.
CNAME	Canonical Name	Specifies an alias for a host.
LOC	Location	Specifies geographic location.
RP	Responsible Person	Lists host contact information.
SRV	Service	Gives the location of well known services.
TXT	Text	Comments or other untyped information.

The SOA record

The SOA record marks the beginning of a zone. Each zone has one SOA record. The record includes the name of the zone, a technical contact (with a period instead of the @ sign), and some timeout values. The following is an example.

```

the-nashes.net. IN SOA ns1.the-nashes.net. jason.the-nashes.net. (
    00120401 ; Serial
    10800   ; Refresh after 3 hours
    3600    ; Retry after 1 hour
    604800  ; Expire after 1 week
    86400   ) ; Minimum TTL of 1 day

```

The name of this zone is `the-nashes.net.` (notice the trailing period). A shortcut is to use the @ symbol, which means to use the current zone. The serial number is used to track changes. Any time the file is modified, the serial number should be manually incremented. This way remote servers know when to update their zone information.

The first timeout specifies the refresh timeout in seconds. This specifies how often slave servers should check with the master to see if the serial number has changed. If the master server is down, the slave servers wait for the retry timeout to expire before trying again. The expire timeout specifies how long the slave servers will continue serving information they retrieved from the master server, if the master server has failed. The minimum time to live value tells the server how long to cache negative answers, such as when a name cannot be resolved. This is new in BIND v8.2.

The NS record

The NS record specifies the name server for a zone or subdomain. The syntax is as follows:

```
Zone      [ttl]      IN      NS      hostname
```

For example:

```
the-nashes.net      IN      NS      ns1.the-nashes.net.
office.the-nashes.net  IN      NS      dns.the-nashes.net.
```

If the zone name matches the zone in the SOA record, it can be left out.

The A record

Most of your records in the zone files will probably be A records. They provide the name-to-IP address mapping that is the heart of DNS. The syntax is as follows:

```
hostname      [ttl]      IN      A      ip_address
```

For example:

```
Norbert      IN      A      216.254.90.133
```

The MX record

Internet mail servers use MX records to route mail correctly. When you send an e-mail to someone at another domain, the mail server checks the remote domain's DNS server to see which system handles mail delivery. The syntax is as follows:

```
name      [ttl]      IN      MX      priority      host
```

The *priority* lets you specify more than one mail server, possibly as a backup to the main server. The lower the priority number, the more likely the server is to be used. For example:

```
the-nashes.net.      IN      MX      10      mail
                    IN      MX      20      mail.myisp.net.
```

This would set up `mail` as the primary mail server, with `mail.myisp.net` as the backup. Since the backup is not part of the zone being configured, the full domain name must be included.

The CNAME record

The CNAME record is used to alias a host to another name. The syntax is as follows:

```
alias [ttl] IN CNAME hostname
```

For example:

```
www IN CNAME brain
```

This example would give the alias of `www` to the host named `brain`. CNAMEs can point to other CNAMEs, and can be nested.

The LOC record

The LOC is a rarely used record that shows the physical location of a service. The syntax is as follows:

```
Name [ttl] IN LOC latitude longitude [alt [size [hp [up]]]]
```

The SRV record

The SRV record shows the locations of services in a domain. Using these records a user could look to see which hosts provide a service, such as FTP. The syntax is as follows:

```
Service.protocol_name [ttl] IN SRV priority weight port target_host
```

For example:

```
ftp.tcp IN SRV 0 0 21 ftp.mydomain.org.
```

This specifies that the FTP service is available on port 21/TCP of `ftp.mydomain.org`. The priority works like MX records, and the weight is used for load balancing.



Microsoft Windows 2000 makes extensive use of the SRV record.

The TXT record

The TXT record adds any text information you want to the DNS information. The syntax is as follows:

```
name [ttl] IN TXT "info"
```

For example:

```
Info IN TXT "Put the information in quotes"
```

The PTR record

The PTR record maps IP addresses back to hostnames. This is done with a special zone called `in-addr.arpa`. The syntax for the PTR record is as follows:

```
address [ttl] IN PTR hostname
```

For example:

```
133 IN PTR norbert.the-nashes.net.
```

IP addresses have the most significant information on the left, whereas domain names such as `www.hungryminds.com` have them on the right. To resolve names to IP addresses the IP must be turned around. The special `in-addr.arpa` domain does this.

The IP address for `norbert.the-nashes.net` is `216.254.90.133`. For this example to function, the zone name in the SOA record would be `90.254.216.in-addr.arpa`. This works because when a server tries to reverse an address to a name, it queries the root name servers for the `in-addr.arpa` TLD. That root server then instructs the querying server to check the owner of the 216 network, and then down to the 254 network, and then the 90 network.

To be able to handle your own reverse DNS, the entire block of IP addresses must be under your control. If your ISP assigned you only a small part of a larger block, it controls the reverse DNS. The TLD servers will not know to direct a querying server to you. However, a “hack” with the CNAME record and zone transfers that lets you manage your reverse entries even if you do not have the entire block exists. Check with your ISP to see if they support it.

Configuring a caching-only name server

A caching-only name server can speed up responses to queries, and you do not need to set up an entire server to use it. The caching-only server remembers answers to queries, and keeps them in a cache for later if needed. The `named.conf` for a caching-only server would look like the following:

```
// generated by named-bootconf.pl

options {
    directory "/var/named";
};

//
// a caching only nameserver config
//
zone "." {
    type hint;
    file "named.ca";
};
```

```
};

zone "0.0.127.in-addr.arpa" {
    type master;
    file "named.local";
};
```

The only other file needed is a `hint` file, which normally comes with a distribution of BIND.



A caching-only name server increases performance for clients by caching the answers to queries for later use. They are easy to set up since they are not responsible for any zone information.

Using BIND v4



BIND v4 uses the `named.boot` configuration file.

The earlier version of BIND was v4. The old v4 is not widely used today, and the only work being done on it is security fixes, when they arise. It is not recommended to use v4 in most cases. The LPI exam requires you to know the differences between v4 and v8, but not how to configure v4. Unlike v8, v4 does not use a `named.conf` file but instead uses a `named.boot` file. The following is an example of a `named.boot` file:

```
; tell what subdir has the lookup database files
directory      /namedb

; type      domain          source host/file
backup file
cache
root.cache
primary 0.0.127.IN-ADDR.ARPA localhost.rev

; example primary server config:
primary mydomain.com mydomain
primary 1.168.192.IN-ADDR.ARPA mydomain.rev
```

The zone files are very similar to those in v8. Table 16-32 lists the configuration entries in `named.boot`.

Table 16-32
named.boot Entries

Entry	Function
domain <domain name>	Sets the default domain name.
directory <directory path>	Sets the working directory for the server.
primary <domain> <file>	Specifies that this server is a primary server for the defined domain. The <file> is where the records for that zone are stored.
secondary <domain> <server list> <file>	Specifies that this server is a secondary for <domain>. The server list shows other servers, at least one a master, for that domain. The <file> stores information transferred from the master.
stub <domain> <server list> <file>	Similar to secondary, but sets this server to be a stub.
cache . root.cache	Used to specify the root.cache file, which is the same as the hint file in v8.
forwarders <server list>	Lists servers to forward queries to.
options forward-only	Defines that this server is not to resolve queries, but instead to forward them.
options no-recursion	Disables recursions, so that if this server cannot answer the query, a negative reply is sent to the client.
options query-log	Logs every query to syslog that is received. This can quickly fill up a file system.
limit <name> <value>	Allows you to set limits, such as the number of inbound or outbound transfers at once.
xfrnets <network>	Limits zone transfers to certain network ranges.

Configuring client DNS

Clients must be configured to use a DNS server. The three main files to configure are the following:

- ♦ /etc/hosts
- ♦ /etc/nsswitch.conf
- ♦ /etc/resolv.conf

Some of this information is covered in more detail in Chapter 15.

The hosts file

The `/etc/hosts` file lists static name-to-IP address mappings. It does not interact with DNS, but provides another mechanism for resolving names. The following is an example.

```
127.0.0.1          localhost.localdomain localhost
10.254.90.155     redhat.the-nashes.net  redhat
```

The nsswitch.conf file

The `/etc/nsswitch.conf` file configures system databases and name services. It specifies how some client libraries resolve names and addresses. If this is set incorrectly, the client may not ever know to check a DNS server. The following is an example entry from this file.

```
passwd:    files nisplus nis
shadow:    files nisplus nis
group:     files nisplus nis

#hosts:    db files nisplus nis dns
hosts:     files nisplus nis dns
```



An incorrectly configured `nsswitch.conf` file can cause name resolution to fail.

The relevant entry for host resolution is `hosts`. Table 16-33 lists the possible settings. The system goes through the fields until the name is resolved. In the previous example it would first check the `/etc/hosts` file, then NIS+, then NIS, and finally DNS. NIS and NIS+ are ways of centralizing information in a UNIX network so that user accounts need to be created in only one place. If you do not use NIS+ or NIS you can speed up the process by removing those fields.

Table 16-33
nsswitch.conf Entries

Entry	Function
<code>nisplus</code> or <code>nis+</code>	Consult NIS+.
<code>nis</code> or <code>yp</code>	Consult NIS.
<code>dns</code>	Use a DNS server.
<code>files</code>	Use local files, normally <code>/etc/hosts</code> , for name resolution.
<code>db</code>	Use local database files, <code>.db</code> files.
<code>compat</code>	Use NIS in compat mode.
<code>[NOTFOUND=return]</code>	Stop searching.

The resolv.conf file

The `/etc/resolv.conf` file configures how the system uses DNS. An example is as follows:

```
search the-nashes.net jasonnash.com
nameserver 10.254.90.131
nameserver 10.254.90.130
nameserver 4.2.2.1
```

The `search` line specifies the domain names to search, if none is specified. For example, if a user typed `www` into a Web browser, the system would search `www.the-nashes.net` and `www.jasonnash.com`.

The `nameserver` entries specify up to three DNS servers to query. Put the most stable server at the top, since the servers are checked in order. The timeout between servers is noticeably long, so you and your users will notice an unstable server at the top of the list.

Using DNS tools

Several command-line tools can be used to troubleshoot and query the DNS database. These are the following:

- ♦ `nslookup`
- ♦ `dig`
- ♦ `host`

Some functionality overlaps between them, so the choice of which to use sometimes comes down to preference.



Debian users may need to install the `dnsutils` package to get these tools.

The nslookup tool

The `nslookup` tool is a powerful query tool that can be used as a noninteractive command-line tool, or from an interactive shell. When you run `nslookup` with no options, it drops you to a prompt that should look like the following:

```
Default Server: ns1.the-nashes.net
Address: 10.254.90.131

>
```

The commands available to you from the prompt are shown in Table 16-34. The syntax to use noninteractive mode is as follows:

```
nslookup [-option...] host-to-find -[server]
```


The same options that are used in interactive mode can be specified on the command-line.

Table 16-34
nslookup Command-Line Options

Command	Function
<code><name></code>	Prints information for the requested name.
<code>help</code> or <code>?</code>	Prints command help.
<code>exit</code>	Exits <code>nslookup</code> .
<code>server <host></code>	Sets the default server to the specified host, using the current server.
<code>lserver <host></code>	Similar to <code>host</code> , but uses the originally set server instead of the current server.
<code>set querytype=<type></code>	Specifies the record type to query for.
<code>set debug</code>	Enables debugging.
<code>set d2</code>	Enables even more debugging.
<code>Set domain=<domain></code>	Changes the default domain.
<code>Set retry=<number></code>	Sets the number of times to retry.
<code>ls [option] <domain></code>	Lists all hosts and address mapping for the requested domain. The following options can be added to list only certain host types: <ul style="list-style-type: none"> -a – Lists canonical (alias) names. -h – Lists host information. -s – Lists well known services. These are services listed with the WKS resource record in DNS. -d – Lists all records. -t <type> – Lists the type of record specified.
<code>finger <user></code>	Fingers the optional user at the default host.
<code>root</code>	Sets the current default server to a root server.

The dig tool

The `dig` (domain information groper) tool roughly provides the same functionality as `nslookup`, but only from a command line. The syntax is as follows:

```
dig [@server] domain [<query-type>] [<query-class>] +<query-option> [-<dig-option>] [%comment]
```

Tables 16-35 and 16-36 show the query types and query classes supported. Query and dig options are usually not used, unless dig is run in batch mode. Information on running batch mode can be found in the man page. An example use of dig is shown here:

```
[root@redhat ~]# dig @ns1.the-nashes.net MX the-nashes.net

; <<>> DiG 8.2 <<>> @ns1.the-nashes.net MX the-nashes.net
; (1 server found)
;; res options: init recurs defnam dnsrch
;; got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 6
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0,
ADDITIONAL: 1
;; QUERY SECTION:
;;      the-nashes.net, type = MX, class = IN

;; ANSWER SECTION:
the-nashes.net.      1H IN MX      10 mail.the-nashes.net.

;; ADDITIONAL SECTION:
mail.the-nashes.net. 1H IN A      216.254.90.131

;; Total query time: 87 msec
;; FROM: redhat.the-nashes.net to SERVER: ns1.the-nashes.net
216.254.90.131
;; WHEN: Sat Dec 9 22:40:26 2000
;; MSG SIZE sent: 32 rcvd: 69
```

As you can see, dig provides a lot of information about the query and how it was resolved.

Table 16-35
dig Query Types Supported

Query Type	Function
a	Host address (default).
any	Any information about the specified domain.
mx	Mail exchanger information for the domain.
ns	Name server information about the domain.
soa	Start of Authority records.
hinfo	Host information records.
axfr	Request a zone transfer. For this to work you must query an authoritative server.
txt	Text string records.

Table 16-36
dig Query Classes Supported

<i>Query Class</i>	<i>Function</i>
in	Internet addresses (default)
any	Any class

The host tool

The `host` tool is smaller than the other two tools and usually provides less information; however, it is useful in many situations when a quick query is needed. The syntax for `host` is as follows:

```
host [-adlrwv] [-t querytype] [-c class] host [server]
```

An example of `host` is as follows:

```
[root@redhat ~]# host -v norbert.the-nashes.net
Trying null domain
rcode = 0 (Success), ancourt=1
The following answer is not verified as authentic by the
server:
norbert.the-nashes.net 3600 IN A          216.254.90.133
```

Table 16-37 shows the available `host` options.

Table 16-37
host Options

<i>Option</i>	<i>Function</i>
-a	Equivalent to <code>-v -t *</code> .
-c	Class to look for, if not Internet.
-d	Enable debugging information.
-l	Turn on list mode.
-r	Disable recursion. The queried server must already know the data requested.
-t	Record type to query for. An <code>*</code> can be used to list all types.
-v	Enable verbose output for more information.
-w	Do not timeout; instead wait forever on a reply.

Managing the DNS server

BIND should never be run from `inetd`. Like the other services, it is usually started and stopped with an `rc` script. On Red Hat this script is the following:

```
/etc/rc.d/init.d/named start/stop/status/restart
```

Debian uses the following:

```
/etc/init.d/bind start/stop/restart/reload/force-reload
```

BIND also provides a tool named `ndc` to manage the `named` daemon. Table 16-38 shows the options for `ndc`. The syntax is as follows:

```
ndc [-c channel] [-l localsock] [-p pidfile] [-d] [-q] [-s]
[-t] [command]
```

Table 16-38
ndc Options

<i>Option</i>	<i>Function</i>
-c	Specify the channel to use to communicate with the DNS server daemon. The default is <code>/var/run/ndc</code> . An address and port can be specified with <code>ipaddr/port</code> , such as <code>127.0.0.1/54</code> .
-l < <i>localsock</i> >	Bind the source <code>ndc</code> connection to a certain interface.
-p < <i>pidfile</i> >	Path to a <code>pidfile</code> file, for older versions of BIND.
-d	Enable debugging output.
-q	Quiet mode.
-s	Do not display nonfatal errors.
-t	Enable protocol and system tracing.

Key Point Summary

The reason that most people run Linux is to provide network services for their users. This chapter covered the major network services used today. The information provided in this chapter should be more than enough for the exam and enough to get the services up and running and working for you right away.

- ♦ `inetd` can be used to start many services that do not need to continuously run.
- ♦ `inetd` is configured via `/etc/inetd.conf`.

- ♦ Each line in `inetd.conf` specifies a service to run, the user to run as, the protocol to use, plus other options.
- ♦ `inetd` is managed by the `/etc/rc.d/init.d/inet` Red Hat script and the `/etc/init.d/inetd` Debian script.
- ♦ The `ftppass` file is used to set restrictions, logging options, information settings, and other options for the FTP server.
- ♦ The `ftphosts` file is used to allow or deny access to an FTP server based on user names and hostnames.
- ♦ The `ftpconversions` file tells the FTP server how to handle compression and archive operations.
- ♦ The `/var/log/xferlog` file logs transfers to and from the FTP server.
- ♦ The `ftpsht` command gracefully shuts down the FTP server and warns users.
- ♦ The `ftpwho` command shows the currently connected users to the FTP server.
- ♦ The `ftpcount` tool shows the number of users connected to the FTP server, broken down by class.
- ♦ The `/etc/issue.net` file is shown to Telnet users before they log in to the system.
- ♦ The MTA, message transfer agent, is used to transport messages between servers using SMTP.
- ♦ The MUA, Mail User Agent, is the application that interfaces users to the mail system.
- ♦ The MDA, Mail Delivery Agent, is used to deliver messages from the server to the user's mailbox.
- ♦ The `sendmail.cf` is the main `sendmail` configuration file and is very complex.
- ♦ The `sendmail.cw` file lists the hosts that mail is accepted for by that server.
- ♦ The `/etc/aliases` file lists mail aliases.
- ♦ When a new mail alias is created, the `newaliases` command should be run.
- ♦ The Smart Host entry may need to be changed in the `sendmail.cf` if you want to forward all mail to another server for delivery.
- ♦ The `m4` preprocessor is used to create a new `sendmail.cf` file from a macro file.
- ♦ A user can create a `.forward` file to have their mail forwarded to another address or file.
- ♦ `sendmail` is managed by the `/etc/rc.d/init.d/sendmail` Red Hat script and the `/etc/init.d/sendmail` Debian script.
- ♦ The `mailq` tool is used to list messages in the mail queue.
- ♦ Apache can be set up to run either standalone or from `inetd`.

- ♦ The `srm.conf` file can be used to control Apache's file types and document specifications.
- ♦ The `access.conf` file can be used to control Apache's access configuration.
- ♦ The `httpd.conf` is the main Apache configuration file and has superseded the other two.
- ♦ Apache is configured using directive commands in the configuration files.
- ♦ The `.htaccess` file can be used to override settings in the main Apache configuration files.
- ♦ The `htpasswd` tool is used to create user and group password files for Apache.
- ♦ Virtual hosts can either use a single IP address or multiple IP addresses.
- ♦ To use name-based virtual hosts the client's Web browser needs to support HTTP/1.1.
- ♦ NFS is used to share out directories to other network users.
- ♦ The NFS exports are configured using the `/etc/exports` file.
- ♦ The `exportfs` tool is used to update the exported out directories and to manually add new directories.
- ♦ Remote NFS directories are mounted with the command `mount host:/directory /mount_point`.
- ♦ The NFS server is managed using the `/etc/rc.d/init.d/nfs` Red Hat script and the `/etc/init.d/nfs-kernel-server` Debian script.
- ♦ The `nfsstat` tool is used to display statistics on the NFS server.
- ♦ Samba is used to share files and printers with Windows clients.
- ♦ Samba is configured via the `smb.conf` file.
- ♦ Samba is managed using the `/etc/rc.d/init.d/smb` Red Hat script and the `/etc/init.d/samba` Debian script.
- ♦ The `testparm` tool is used to check the `smb.conf` for syntax errors.
- ♦ The `smbstatus` tool is used to show users currently connected to the Samba server.
- ♦ The `smbclient` utility is used to connect to a Samba server or Windows share.
- ♦ DNS is used to map names to IP addresses, and vice versa.
- ♦ The root name servers are responsible for directing querying servers to TLD servers.
- ♦ The DNS namespace is a hierarchical tree structure, with the root name servers at the "top" and individually administered domains at the "bottom."
- ♦ BIND is a software suite that provides an implementation of DNS.

- ♦ BIND v8 uses the `named.conf` file for server configuration.
- ♦ Zone files contain DNS resource records, which hold the mapping information.
- ♦ A BIND server is a master server if it holds the master copy of zone information for that domain or subdomain.
- ♦ A BIND server is a slave server if it gets its zone information from a master server.
- ♦ A BIND forward server forwards all queries to other DNS servers.
- ♦ A BIND stub server retrieves only NS records from the master server.
- ♦ When zone information is copied to another server, it is known as a zone transfer.
- ♦ A caching-only name server does not have authority over a domain. It only queries other servers for information and caches that information for later use to increase performance.
- ♦ BIND v4 uses the `named.boot` configuration file.
- ♦ The `/etc/hosts` file stores static mapping information.
- ♦ The `/etc/nsswitch.conf` file controls how a system resolves names.
- ♦ The `/etc/resolv.conf` file lists the search domains and the DNS servers to query.
- ♦ The `nslookup` tool provides an interactive and noninteractive interface to perform DNS queries.
- ♦ The `dig` tool has similar functionality as `nslookup`, but can provide more information.
- ♦ `host` is a small tool that can do queries, but usually provides less information than `dig`.
- ♦ The `ndc` tool is used to manage the BIND `named` daemon.
- ♦ The BIND daemon is managed with the `/etc/rc.d/init.d/named` Red Hat script and `/etc/init.d/bind` Debian script.



STUDY GUIDE

The following questions and exercises will allow you to review the information covered in this chapter. Take your time when completing this section of the chapter, carefully reviewing any questions that you may have. Answering the question correctly is not as important as understanding the answer, so review any material that you might still be unsure of. Being comfortable with the questions and answers presented here will help you be more prepared for the certification exam questions.

Assessment Questions

1. Which socket type is used in `inetd.conf` when the service uses TCP?
 - A. `dgram`
 - B. `stream`
 - C. `raw`
 - D. `rdm`
2. The `inetd.conf` file is read every 30 seconds by `inetd` to look for configuration changes.
 - A. True
 - B. False
3. The account used for anonymous FTP should have which shell?
 - A. `/bin/bash`
 - B. `/bin/csh`
 - C. `/bin/false`
 - D. `/bin/ksh`
4. Which entry in `ftpdaccess` limits the number of failed password attempts?
 - A. `loginfails`
 - B. `loginlimit`
 - C. `badpw`
 - D. `pwattempts`

5. Which tool is used to see who is logged in to your FTP server?
- A. ftpstats
 - B. whoftp
 - C. ftpd -w
 - D. ftpwho
6. What needs to be done to create a new mail alias for a user? (Choose all that apply.)
- A. Add the alias to `/etc/aliases`.
 - B. Add the alias to `/etc/mailboxes`.
 - C. Run the `newaliases` command.
 - D. Run the `addalias` command.
7. The _____ command is used to list messages currently in the mail queue.
8. To make your Web server perform the best, it should be run via `inetd`.
- A. True
 - B. False
9. What needs to be done to run `httpd` through `inetd`? (Choose all that apply.)
- A. Change the port number specified in `httpd.conf`.
 - B. Change the `ServerType` directive in `httpd.conf`.
 - C. Add an entry to `inetd.conf` for `httpd`.
 - D. Run `inetd -d /usr/sbin/httpd`.
10. By default Apache uses the `/etc/passwd` file for user authentication.
- A. True
 - B. False
11. Which entry or entries in `/etc/exports` will share the `/home` directory only to users on the `192.168.10.0/24` network, which is made up of the domain `mydomain.org`?
- A. `/home 192.168.10.0 - 254(rw)`
 - B. `/home 192.168.10.0/24(ro)`
 - C. `/home *.mydomain.org(rw)`
 - D. `/home ANY.mydomain.org(rw)`

12. After editing the `/etc/exports` file, the _____ command should be run (no parameters).
13. Which `smb.conf` entry sets the Windows NT domain for Samba?
- A. `workgroup`
 - B. `domain`
 - C. `nt_domain`
 - D. `MS_DOMAIN`
14. Which command(s) shows all the files on a Windows share using `smbclient`?
- A. `dir`
 - B. `list`
 - C. `cp`
 - D. `ls`
15. BIND v4 uses the _____ configuration file.
- A. `named.boot`
 - B. `named.conf`
 - C. `named.cf`
 - D. `conf.named`
16. Which DNS resource record is used to reverse map IP addresses to names?
- A. A
 - B. MX
 - C. SRV
 - D. PTR
17. Which DNS resource record shows the hosts that provide services?
- A. SRV
 - B. LOC
 - C. A
 - D. CNAME

18. Which field should be put at the front of the hosts entry in the `nsswitch.conf` file to have the system check `/etc/hosts` first?
- A. files
 - B. NIS
 - C. dns
 - D. yp
19. Which mail server has the highest priority?
- A. IN MX 10 mail
 - B. IN MX 20 mail
 - C. IN MX 100 mail
 - D. IN MX 0 mail
20. Which file is displayed to users before they log in to a system via Telnet?
- A. `/etc/issue`
 - B. `/etc/login.net`
 - C. `/etc/motd.net`
 - D. `/etc/issue.net`

Scenarios

1. You're a network administrator at a medium-sized company. To help maintain the Linux workstations on your network, you want to share a common base of files. What is the best way to accomplish this?
2. While doing routine network monitoring, you notice that a lot of traffic across your WAN is from DNS queries. What could you do to help reduce the amount of traffic and increase performance for your users?
3. Your company has several salesmen who travel constantly. They need to access several documents and files from remote locations while on the road. How could you set up a service to let them retrieve these files?

Lab Exercises

Lab 16-1 Local DNS server

Unless you have registered your own domain name, you can only set up a local DNS server. The zone hosted by this server will not be accessible from the outside

world. Red Hat and Debian both provide packages that automatically install BIND as a caching name server. All steps in this lab require the user to be logged in as root.

To install BIND on a Debian system, use the following steps:

1. Run an `apt-get update` to make sure your package list is current.

```
debian:~# apt-get update
Hit http://http.us.debian.org stable/main Packages
Hit http://http.us.debian.org stable/main Release
Hit http://http.us.debian.org stable/contrib Packages
Hit http://http.us.debian.org stable/contrib Release
Hit http://http.us.debian.org stable/non-free Packages
Hit http://http.us.debian.org stable/non-free Release
Hit http://non-us.debian.org stable/non-US/main Packages
Hit http://non-us.debian.org stable/non-US/main Release
Hit http://non-us.debian.org stable/non-US/contrib Packages
Hit http://non-us.debian.org stable/non-US/contrib Release
Hit http://non-us.debian.org stable/non-US/non-free Packages
Hit http://non-us.debian.org stable/non-US/non-free Release
Hit http://security.debian.org stable/updates/main Packages
Get:1 http://security.debian.org stable/updates/main Release
[113B]
Hit http://security.debian.org stable/updates/contrib
Packages
Get:2 http://security.debian.org stable/updates/contrib
Release [116B]
Hit http://security.debian.org stable/updates/non-free
Packages
Get:3 http://security.debian.org stable/updates/non-free
Release [117B]
Fetched 346B in 3s (100B/s)
Reading Package Lists... Done
Building Dependency Tree... Done
```

2. Install the `bind` package using `apt-get install bind`. If prompted to add the local DNS server to the `/etc/resolv.conf` file, answer yes.

```
debian:~# apt-get install bind
Reading Package Lists... Done
Building Dependency Tree... Done
The following NEW packages will be installed:
  bind
0 packages upgraded, 1 newly installed, 0 to remove and 1 not
upgraded.
Need to get 571kB of archives. After unpacking 1323kB will be
used.
Get:1 http://security.debian.org stable/updates/main bind
1:8.2.3-0.potato.1 [571kB]
Fetched 571kB in 10s (56.1kB/s)
Selecting previously deselected package bind.
(Reading database ... 9900 files and directories currently
installed.)
```

```
Unpacking bind (from ../bind_1%3a8.2.3-0.potato.1_i386.deb)
...
Setting up bind (8.2.3-0.potato.1) ...
```

To be most effective, /etc/resolv.conf should list the IP address of your local machine (127.0.0.1) as a nameserver. It currently does not.

```
Would you like this to be added? [Y]
Starting domain name service: named.
```

3. To use the DNS tools you will need to install them using apt-get install dnsutils.

```
debian:~# apt-get install dnsutils
Reading Package Lists... Done
Building Dependency Tree... Done
The following NEW packages will be installed:
  dnsutils
0 packages upgraded, 1 newly installed, 0 to remove and 1 not
upgraded.
Need to get 340kB of archives. After unpacking 778kB will be
used.
Get:1 http://security.debian.org stable/updates/main dnsutils
1:8.2.3-0.potato.1 [340kB]
Fetched 340kB in 8s (38.0kB/s)
Selecting previously deselected package dnsutils.
(Reading database ... 9901 files and directories currently
installed.)
Unpacking dnsutils (from
../dnsutils_1%3a8.2.3-0.potato.1_i386.deb) ...
Setting up dnsutils (8.2.3-0.potato.1) ...
```

To install BIND on a Red Hat system follow these steps.

1. Download the bind-8.2.2_P5-25.i386.rpm, caching-nameserver-7.0-6.noarch.rpm, and bind-utils-8.2.2_P5-25.i386.rpm files from Red Hat's FTP site.

```
ftp> get bind-8.2.2_P5-25.i386.rpm
local: bind-8.2.2_P5-25.i386.rpm remote: bind-
8.2.2_P5-25.i386.rpm
227 Entering Passive Mode (216,148,218,192,144,191)
150 Opening BINARY mode data connection for bind-
8.2.2_P5-25.i386.rpm (1748960 bytes).
226 Transfer complete.
1748960 bytes received in 29.9 secs (57 Kbytes/sec)
ftp> get bind-utils-8.2.2_P5-25.i386.rpm
local: bind-utils-8.2.2_P5-25.i386.rpm remote: bind-utils-
8.2.2_P5-25.i386.rpm
227 Entering Passive Mode (216,148,218,192,229,109)
```

```

150 Opening BINARY mode data connection for bind-utils-
8.2.2_P5-25.i386.rpm (738549 bytes).
226 Transfer complete.
738549 bytes received in 12.1 secs (59 Kbytes/sec)
ftp> get caching-nameserver-7.0-6.noarch.rpm
local: caching-nameserver-7.0-6.noarch.rpm remote: caching-
nameserver-7.0-6.noarch.rpm
227 Entering Passive Mode (216,148,218,192,117,151)
150 Opening BINARY mode data connection for caching-
nameserver-7.0-6.noarch.rpm (5810 bytes).
226 Transfer complete.
5810 bytes received in 0.0976 secs (58 Kbytes/sec)

```

2. Install the newly downloaded RPM files. The `bind-utils` package may already be installed. This can be checked with `rpm` as well.

```

[root@rh7 bind]# rpm -q bind-utils
bind-utils-8.2.2_P5-25
[root@rh7 bind]# rpm -ivh *.rpm
bind
#####
caching-nameserver
#####

```

3. Change the `nameserver` entry in `/etc/resolv.conf` to `127.0.0.1` so that it uses the newly installed local nameserver.

```

search the-nashes.net
nameserver 127.0.0.1

```

4. Start `named` by running the `named` script in `init.d`.

```

[root@rh7 /root]# /etc/init.d/named start
Starting named: [
OK ]

```

Debian stores its BIND configuration in `/etc/bind` while Red Hat stores its `named.conf` in `/etc` and other files in `/var/named`. The final step is to test your newly installed name server by resolving names. For example:

```

debian:~# host www.hungryminds.com
www.hungryminds.com A 38.170.216.15
debian:~# host www.linux.com
www.linux.com CNAME linux.com
linux.com A 216.136.171.205

```

Lab 16-2 NFS server

This lab configures an NFS server to export out a directory. This lab works with either one or two workstations, as you can connect to a locally shared NFS export.

To install the NFS server on a Debian system, log in as root and follow these steps:

1. Make sure that your package list is updated by running `apt-get update`.

```
debian:~# apt-get update
Hit http://http.us.debian.org stable/main Packages
Hit http://http.us.debian.org stable/main Release
Hit http://http.us.debian.org stable/contrib Packages
Hit http://http.us.debian.org stable/contrib Release
Hit http://http.us.debian.org stable/non-free Packages
Hit http://http.us.debian.org stable/non-free Release
Hit http://non-us.debian.org stable/non-US/main Packages
Hit http://non-us.debian.org stable/non-US/main Release
Hit http://non-us.debian.org stable/non-US/contrib Packages
Hit http://non-us.debian.org stable/non-US/contrib Release
Hit http://non-us.debian.org stable/non-US/non-free Packages
Hit http://non-us.debian.org stable/non-US/non-free Release
Hit http://security.debian.org stable/updates/main Packages
Get:1 http://security.debian.org stable/updates/main Release
[113B]
Hit http://security.debian.org stable/updates/contrib
Packages
Get:2 http://security.debian.org stable/updates/contrib
Release [116B]
Hit http://security.debian.org stable/updates/non-free
Packages
Get:3 http://security.debian.org stable/updates/non-free
Release [117B]
Fetched 346B in 3s (100B/s)
Reading Package Lists... Done
Building Dependency Tree... Done
```

2. Install the NFS kernel server by running `apt-get install nfs-kernel-server`.

```
debian:~# apt-get install nfs-kernel-server
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
  nfs-common
The following NEW packages will be installed:
  nfs-common nfs-kernel-server
0 packages upgraded, 2 newly installed, 0 to remove and 1 not
upgraded.
Need to get 75.2kB of archives. After unpacking 344kB will be
used.
Do you want to continue? [Y/n]
Get:1 http://http.us.debian.org stable/main nfs-common
1:0.1.9.1-1 [23.1kB]
Get:2 http://http.us.debian.org stable/main nfs-kernel-server
1:0.1.9.1-1 [52.1kB]
Fetched 75.2kB in 0s (127kB/s)
Selecting previously deselected package nfs-common.
```

```
(Reading database ... 9899 files and directories currently
installed.)
Unpacking nfs-common (from .../nfs-
common_1%3a0.1.9.1-1_i386.deb) ...
Selecting previously deselected package nfs-kernel-server.
Unpacking nfs-kernel-server (from .../nfs-kernel-
server_1%3a0.1.9.1-1_i386.deb) ...
Setting up nfs-common (0.1.9.1-1) ...
Starting NFS common utilities: statd lockd.

Setting up nfs-kernel-server (0.1.9.1-1) ...
Not starting NFS kernel daemon: No exports.
```

3. Edit the /etc/exports file and share out the /home directory. For example:

```
# /etc/exports: the access control list for filesystems which
may be exported
#          to NFS clients.  See exports(5).

/home          (rw)
```

4. Start the NFS server by running the /etc/init.d/nfs-kernel-server script.

```
debian:~# /etc/init.d/nfs-kernel-server start
Starting NFS kernel daemon: nfsd mountd.
Exporting directories for NFS kernel daemon...done.
```

5. Make sure the directory was shared out by running exportfs.

```
debian:~# exportfs

/home          <world>
```

To install the NFS server on a Red Hat system, log in as root and follow these steps:

1. Check to see if the nfs-utils package is already installed on the system by running rpm -q nfs-utils. If it is, skip the next two steps.

```
[root@rh7 nfs]# rpm -q nfs-utils
nfs-utils-0.1.9.1-7
```

2. Download the nfs-utils-0.1.9.1-7.i386.rpm file from the Red Hat FTP site, or retrieve it from the installation CD-ROM.

```
ftp> get nfs-utils-0.1.9.1-7.i386.rpm
local: nfs-utils-0.1.9.1-7.i386.rpm remote: nfs-utils-
0.1.9.1-7.i386.rpm
227 Entering Passive Mode (216,148,218,192,4,27)
150 Opening BINARY mode data connection for nfs-utils-
0.1.9.1-7.i386.rpm (182828 bytes).
```



```
226 Transfer complete.
182828 bytes received in 1.59 secs (1.1e+02 Kbytes/sec)
```

- 3. Install the nfs-utils RPM package by running** `rpm -ivh nfs-utils*`.

```
[root@rh7 nfs]# rpm -ivh nfs-utils*
nfs-utils
#####
```

- 4. Edit the /etc/exports file and share out the /home directory. For example:**

```
# /etc/exports: the access control list for filesystems which
may be exported
#
to NFS clients. See exports(5).

/home                (rw)
```

- 5. Start the NFS server by running** `/etc/init.d/nfs start`.

```
[root@rh7 /root]# /etc/init.d/nfs start
Starting NFS services: [
OK ]
Starting NFS quotas: [
OK ]
Starting NFS mountd: [
OK ]
Starting NFS daemon: [
OK ]
```

- 6. Check to make sure the directory is being exported correctly by running** `exportfs`.

```
[root@rh7 /root]# exportfs
/home                <world>
```

The next steps guide you through mounting an NFS share. If you have two systems you should mount the share across the network for practice. Be sure that the user and group IDs match between the two systems so that the permissions are correct. If you have to change the IDs on one system, be sure to reset the permissions to their home directory and other files using the `chown` command.

- 1. Create a directory to mount the export to.**

```
[root@rh7 /mnt]# mkdir /mnt/nfs
[root@rh7 /mnt]# ls
cdrom floppy nfs
```

- 2. Mount the directory using the command** `mount ip.address:/home /mnt/nfs`.

```
[root@rh7 /mnt]# mount 10.254.90.145:/home /mnt/nfs
```

3. As a normal user, change to the `/mnt/nfs` directory and make sure the mapping worked as expected.

```
[jason@rh7 jason]$ cd /mnt/nfs
[jason@rh7 nfs]$ ls
angie jason
```

Lab 16-3 Apache server

For the exam you should know how to install and manage the Apache Web server. This lab can be performed with one or more systems. All of these steps require the user to be logged in as root.

To install Apache on a Debian system follow these steps:

1. Make sure that your package list is updated by running `apt-get update`.

```
debian:~# apt-get update
Hit http://http.us.debian.org stable/main Packages
Hit http://http.us.debian.org stable/main Release
Hit http://http.us.debian.org stable/contrib Packages
Hit http://http.us.debian.org stable/contrib Release
Hit http://http.us.debian.org stable/non-free Packages
Hit http://http.us.debian.org stable/non-free Release
Hit http://non-us.debian.org stable/non-US/main Packages
Hit http://non-us.debian.org stable/non-US/main Release
Hit http://non-us.debian.org stable/non-US/contrib Packages
Hit http://non-us.debian.org stable/non-US/contrib Release
Hit http://non-us.debian.org stable/non-US/non-free Packages
Hit http://non-us.debian.org stable/non-US/non-free Release
Hit http://security.debian.org stable/updates/main Packages
Get:1 http://security.debian.org stable/updates/main Release
[113B]
Hit http://security.debian.org stable/updates/contrib
Packages
Get:2 http://security.debian.org stable/updates/contrib
Release [116B]
Hit http://security.debian.org stable/updates/non-free
Packages
Get:3 http://security.debian.org stable/updates/non-free
Release [117B]
Fetched 346B in 3s (100B/s)
Reading Package Lists... Done
Building Dependency Tree... Done
```

2. To install the Apache server, you type `apt-get install apache`.

```
debian:~# apt-get install apache
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
```

```

    apache-common
The following NEW packages will be installed:
    apache apache-common
0 packages upgraded, 2 newly installed, 0 to remove and 1 not
upgraded.
Need to get 1077kB of archives. After unpacking 2171kB will
be used.
Do you want to continue? [Y/n]
Get:1 http://security.debian.org stable/updates/main apache
1.3.9-13.2 [359kB]
Get:2 http://security.debian.org stable/updates/main apache-
common 1.3.9-13.2 [719kB]
Fetched 1077kB in 26s (40.8kB/s)
Selecting previously deselected package apache-common.
(Reading database ... 9652 files and directories currently
installed.)
Unpacking apache-common (from ../apache-
common_1.3.9-13.2_i386.deb) ...
Selecting previously deselected package apache.
Unpacking apache (from ../apache_1.3.9-13.2_i386.deb) ...
Setting up apache-common (1.3.9-13.2) ...

Setting up apache (1.3.9-13.2) ...

Initializing apache config for immediate operation.
Installing new configuration file /etc/apache/httpd.conf ...
Installing new configuration file /etc/apache/access.conf ...
Installing new configuration file /etc/apache/srm.conf ...
Installing new configuration file /etc/apache/cron.conf ...
The ServerAdmin is set to webmaster@debian.
The DocumentRoot is set to /var/www.
Leaving existing site /var/www/index.html untouched.
Finding DSO
mods.....found.

# LoadModule vhost_alias_module
/usr/lib/apache/1.3/mod_vhost_alias.so
# LoadModule env_module /usr/lib/apache/1.3/mod_env.so
LoadModule config_log_module
/usr/lib/apache/1.3/mod_log_config.so
# LoadModule mime_magic_module
/usr/lib/apache/1.3/mod_mime_magic.so
LoadModule mime_module /usr/lib/apache/1.3/mod_mime.so
LoadModule negotiation_module
/usr/lib/apache/1.3/mod_negotiation.so
LoadModule status_module /usr/lib/apache/1.3/mod_status.so
# LoadModule info_module /usr/lib/apache/1.3/mod_info.so
# LoadModule includes_module
/usr/lib/apache/1.3/mod_include.so
LoadModule autoindex_module
/usr/lib/apache/1.3/mod_autoindex.so
LoadModule dir_module /usr/lib/apache/1.3/mod_dir.so

```

```

LoadModule cgi_module /usr/lib/apache/1.3/mod_cgi.so
# LoadModule asis_module /usr/lib/apache/1.3/mod_asis.so
# LoadModule imap_module /usr/lib/apache/1.3/mod_imap.so
# LoadModule action_module /usr/lib/apache/1.3/mod_actions.so
# LoadModule speling_module
/usr/lib/apache/1.3/mod_speling.so
LoadModule userdir_module /usr/lib/apache/1.3/mod_userdir.so
LoadModule alias_module /usr/lib/apache/1.3/mod_alias.so
LoadModule rewrite_module /usr/lib/apache/1.3/mod_rewrite.so
LoadModule access_module /usr/lib/apache/1.3/mod_access.so
LoadModule auth_module /usr/lib/apache/1.3/mod_auth.so
# LoadModule anon_auth_module
/usr/lib/apache/1.3/mod_auth_anon.so
# LoadModule dbm_auth_module
/usr/lib/apache/1.3/mod_auth_dbm.so
# LoadModule db_auth_module
/usr/lib/apache/1.3/mod_auth_db.so
# LoadModule proxy_module /usr/lib/apache/1.3/libproxy.so
# LoadModule digest_module /usr/lib/apache/1.3/mod_digest.so
# LoadModule cern_meta_module
/usr/lib/apache/1.3/mod_cern_meta.so
LoadModule expires_module /usr/lib/apache/1.3/mod_expires.so
# LoadModule headers_module
/usr/lib/apache/1.3/mod_headers.so
# LoadModule usertrack_module
/usr/lib/apache/1.3/mod_usertrack.so
LoadModule unique_id_module
/usr/lib/apache/1.3/mod_unique_id.so
LoadModule setenvif_module
/usr/lib/apache/1.3/mod_setenvif.so
# LoadModule sys_auth_module
/usr/lib/apache/1.3/mod_auth_sys.so
# LoadModule put_module /usr/lib/apache/1.3/mod_put.so
# LoadModule throttle_module
/usr/lib/apache/1.3/mod_throttle.so
# LoadModule allowdev_module
/usr/lib/apache/1.3/mod_allowdev.so
# LoadModule auth_mysql_module
/usr/lib/apache/1.3/mod_auth_mysql.so
# LoadModule pgsq1_auth_module
/usr/lib/apache/1.3/mod_auth_pgsq1.so
# LoadModule eaccess_module
/usr/lib/apache/1.3/mod_eaccess.so
# LoadModule roaming_module
/usr/lib/apache/1.3/mod_roaming.so

```

Pondering..... done.

```
/usr/sbin/apachectl start: httpd started
```

3. Test the Web server by using lynx or netscape. A home page is installed with the package for you to view.

To install Apache on a Red Hat system follow these steps:

1. Check to see if Apache is already installed on the system by running `rpm -q apache`. If it is, you can skip this lab or remove it with `rpm -e apache` and do the lab.

```
[root@rh7 apache]# rpm -q apache
apache-1.3.12-25
```

2. Download the `apache-1.3.12-25.i386.rpm` package from the Red Hat FTP site or retrieve it from the installation CD-ROM.

```
ftp> get apache-1.3.12-25.i386.rpm
local: apache-1.3.12-25.i386.rpm remote: apache-
1.3.12-25.i386.rpm
227 Entering Passive Mode (216,148,218,201,16,179)
150 Opening BINARY mode data connection for apache-
1.3.12-25.i386.rpm (482970 bytes).
226 Transfer complete.
482970 bytes received in 6.23 secs (76 Kbytes/sec)
```

3. Install the Apache package by running `rpm -ivh apache-1.3.12-25.i386.rpm`.

```
[root@rh7 apache]# rpm -ivh apache-1.3.12-25.i386.rpm
apache
#####
```

4. Test the Web server by using `lynx` or `netscape`. A home page is installed with the package for you to view.

Answers to Chapter Questions

Chapter Pre-Test

1. The Apache package provides the `apachectl` tool to manage the service.
2. The MX record tells mail servers where to forward mail to for a domain.
3. Directories to be exported are listed in the `/etc/exportfs` file.
4. The `ftppass` file controls logging for the `wu-ftpd` server.
5. The `ftpwho` tool shows who is connected to the FTP server, along with what they are doing.
6. Red Hat includes the `/etc/rc.d/init.d/sendmail` script to control the `sendmail` daemon.

7. The `LogFormat` and `CustomLog` directives control logging in Apache. `LogFormat` changes the standard logging, while `CustomLog` does that and lets you output logs to a new file.
8. The `nfsstat` tool displays statistics about the NFS server that can help troubleshooting and performance tweaking.
9. The `smbclient` tool is used to connect to remote Windows and Samba shares.
10. BIND v4 uses the `named.boot` file, while BIND v8 uses the `named.conf` file.

Assessment Questions

1. **B.** The `stream` type is always used with TCP, and `dgram` is used with UDP. For more information see the “Configuring `inetd`” section.
2. **B.** Any time the `inetd.conf` file is changed, `inetd` needs to be restarted for the changes to take effect. For more information see the “Restarting the `inetd` process” section.
3. **C.** Anonymous FTP accounts should not have a valid shell statement in `/etc/passwd`. This limits login to FTP only. For more information see the “Setting up anonymous FTP” section.
4. **A.** If a user exceeds the limit set by `loginfails`, they are disconnected. For more information see the “The `ftppass` file” section.
5. **D.** The `ftpwho` command shows who is connected to the FTP server. For more information see the “The `ftpwho` command” section.
6. **A and C.** Mail aliases are stored in `/etc/aliases`. Any time an entry is added the `newaliases` command needs to be run. For more information see the “Aliasing and forwarding mail” section.
7. **mailq.** The `mailq` command shows any messages waiting in the queue as well as any errors that may keep them from being delivered. For more information see the “Managing `sendmail`” section.
8. **B.** Only run Apache through `inetd` for a Web server that expects very few hits. The overhead of starting the Apache daemon every time a client connects severely impacts performance. For more information see the “Starting and stopping `httpd`” section.
9. **B and C.** An entry needs to be added to the `inetd.conf` file so that `inetd` knows to start the service when a user connects. The `ServerType` entry needs to be set correctly in `httpd.conf` so that Apache knows not to keep running when a connection completes. For more information see the “Server configuration” section for Apache.
10. **B.** Apache uses a file created by `htpasswd`. For more information see the “Authentication” section for Apache.
11. **B and C.** Restrictions can be set using CIDR notation or using wildcards with domains. For more information see the “Configuring exports” section.

12. **exportfs**. The `exportfs` command is used to display and change the currently exported directories. For more information see the “Configuring exports” section.
13. **A**. The `workgroup` entry sets both the workgroup name and the Windows NT domain name. For more information see the “Configuring Samba” section.
14. **A and D**. Both `dir` and `ls` can be used to list files. The other answers are invalid. For more information see the “Client connections” section.
15. **A**. BIND v4 uses `named.boot` while BIND v8 uses `named.conf`. For more information see the “Using BIND v4” section.
16. **D**. The PTR record is used for reverse queries. For more information see the “Zone files” section.
17. **A**. The SRV record shows which hosts provide certain services. The LOC record shows the physical location of a host. For more information see the “Zone files” section.
18. **A**. The `files` entry tells the system to check `/etc/hosts` first. The `dns` entry specifies when to check with a DNS server. For more information see “The `nsswitch.conf` file” section.
19. **D**. The lower the number set in the MX record, the higher the priority for the mail server. For more information see the “Zone files” section.
20. **D**. The `/etc/issue.net` file is shown to local users before they log in. For more information see the “Configuring Telnet” section.

Scenarios

1. The suggested solution would be to share out the common files via NFS and put the mount options in `/etc/fstab` so the file system is loaded at boot. This is a commonly used practice to ease administration of a large number of systems. The NFS server should be reliable and stable.
2. A caching-only DNS server could be placed at the remote sites. This way when other users need to DNS the same name, it can be retrieved from cache instead of sending another query across the WAN.
3. There are two ways to handle this. The first way is to configure an FTP server that holds the needed files. The second way is to use Apache. Either way, the information can be secured using either standard login accounts for FTP or secured directories on Apache.

Managing Security

17

C H A P T E R

EXAM OBJECTIVES

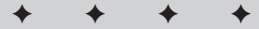
Exam 102 ♦ General Linux, Part 2

1.13 Networking Services

- **Configure and manage inetd and related services.** Configure which services are available through inetd, use tcp-wrappers to allow or deny services on a host-by-host basis, manually start, stop, and restart internet services, configure basic network services including telnet and ftp. Includes managing inetd.conf, hosts.allow, and hosts.deny.

1.14 Security

- **Perform security admin tasks.** Configure and use TCP wrappers to lock down the system, list all files with SUID bit set, determine if any package (.rpm or .deb) has been corrupted, verify new packages prior to install, use setgid on dirs to keep group ownership consistent, change a user's password, set expiration dates on user's passwords, obtain, install and configure ssh
- **Setup host security.** Implement shadowed passwords, turn off unnecessary network services in inetd, set the proper mailing alias for root and setup syslogd, monitor CERT and BUGTRAQ, update binaries immediately when security problems are found
- **Setup user level security.** Set limits on user logins, processes, and memory usage.



CHAPTER PRE-TEST

1. What is a secure replacement for Telnet and the Berkeley `r` commands?
2. Which files control access with TCP wrappers?
3. How can you limit the number of processes a user runs?
4. How can you create a directory for users to share files?
5. What is the main configuration file for `sshd`?
6. Which `r` command is used to execute a command on a remote system?
7. Which services cannot be used with TCP wrappers?
8. Where are the PAM configuration files stored?
9. Which package is used to set up a firewall on Linux?
10. What is used to get a network on the Internet with only one IP address?

Security is a process, not a procedure. There are a number of things you can do to initially secure a system, but you must also be prepared to continually update and monitor the system. New security exploits are found all the time, so a system that is secure today may not be secure in a month. This chapter covers the basics of securing a Linux system and its services. It starts with securing `inetd` and other services and continues with administration tasks such as TCP wrappers and file system settings. Also included in this chapter are pointers to security resources on the Internet that you should monitor so that you are aware of current exploits.

Performing Security Administration Tasks



1.13 Networking Services

- **Configure and manage `inetd` and related services.** Configure which services are available through `inetd`, use `tcpwrappers` to allow or deny services on a host-by-host basis, manually start, stop, and restart internet services, configure basic network services including `telnet` and `ftp`. Includes managing `inetd.conf`, `hosts.allow`, and `hosts.deny`.

1.14 Security

- **Perform security admin tasks.** Configure and use TCP wrappers to lock down the system, list all files with SUID bit set, determine if any package (`.rpm` or `.deb`) has been corrupted, verify new packages prior to install, use `setgid` on dirs to keep group ownership consistent, change a user's password, set expiration dates on user's passwords, obtain, install and configure `ssh`

An administrator can do several things to secure a newly installed system. These involve locking down some services with TCP wrappers and managing users and groups.



Assigning and using passwords are topics covered in more detail in Chapter 10.

Configuring TCP wrappers

The `tcpd` daemon, also known as the *TCP wrappers package*, can be used to limit what connections can be made to some network services. This way you can limit access to the services to only authorized users. It is easy to configure and integrate into a working system. No changes need to be made to the network daemons, only minor changes to the `inetd.conf` file. For example, if your `inetd.conf` file currently has the following entry for the FTP server:

```
ftp      stream  tcp      nowait  root    /usr/sbin/in.ftpd
in.ftpd -l -a
```

you would change it to the following to use TCP wrappers:

```
ftp      stream tcp      nowait  root    /usr/sbin/tcpd  in.ftpd
-l -a
```

The normal daemon server path is replaced with the path for `tcpd`, with the argument being the actual server to run. TCP wrappers uses two files to configure allowed connections:

- ♦ `/etc/hosts.allow`
- ♦ `/etc/hosts.deny`

Both configuration files use the same format. If either file does not exist, it is considered to be empty. If there are entries in `hosts.allow`, but not `hosts.deny`, anything not explicitly granted is denied. Conversely, if there are entries in `hosts.deny`, but not in `hosts.allow`, connections are allowed in unless explicitly denied. If entries exist in both, the `hosts.allow` is checked first, and then `hosts.deny`. So even if a connection is denied in `hosts.deny`, it can be overridden with `hosts.allow`. The syntax for these files is as follows:

```
service : host_list [: shell command]
```

The service name is exactly the name listed in `/etc/services` and `/etc/inetd.conf`. The host list is a list of hosts, domains, wildcards, or CIDR addresses. For example, the following entry in `/etc/hosts.allow` would allow any host in the `the-nashes.net` domain to connect to the FTP server:

```
in.ftpd : .the-nashes.net
```

Table 17-1 shows the possible host and service match methods.

Table 17-1
Match Methods

<i>Method</i>	<i>Function</i>
<code>.domain.org</code>	A domain that begins with a dot means "any host in that domain."
<code>192.168.</code>	Any numeric address that ends in a dot means any host that has that IP prefix. This would match <code>192.168.*.*</code> .
<code>Address/Mask</code>	A combination of a network address and subnet mask, such as <code>192.168.0.0/16</code> .
<code>ALL</code>	Match any client or service name.
<code>LOCAL</code>	Match any client hostname that does not contain a dot.
<code>EXCEPT</code>	Allows exceptions to <code>ALL</code> .



Be careful with the `LOCAL` command. If your `/etc/hosts` file contains an entry with just a host name and no domain, it may be allowed access whether it is actually local or not.

The optional shell command can be used to perform a function when a client connects that matches the specified rule. A number of patterns can be used in the shell command that are expanded out when run. Table 17-2 lists these patterns. Another useful option is the `twist` command. This causes the output of the shell command to be shown to the connecting client, along with any information from the expansion patterns. For example:

```
in.telnetd : .evil.org : twist /bin/echo "Sorry %c, but we
don't want your kind around here."
```

This would cause the message to be echoed out to any user from the `evil.org` domain connecting to the Telnet server.

Table 17-2
tcpd Patterns

<i>Pattern</i>	<i>Function</i>
<code>%a</code>	The connecting client's host address.
<code>%A</code>	The server's host address.
<code>%c</code>	Client information such as <code>user@host</code> , <code>user@address</code> , <code>hostname</code> , or address depending on what is available.
<code>%d</code>	The daemon process name.
<code>%h</code>	The connecting client's hostname, or address if the host cannot be retrieved.
<code>%H</code>	The server's hostname, or address if the host cannot be retrieved.
<code>%n</code>	The client hostname.
<code>%N</code>	The server hostname.
<code>%p</code>	The daemon's process ID.
<code>%s</code>	Server information such as <code>daemon@host</code> , <code>daemon@address</code> , or just the daemon name depending on what is available.
<code>%u</code>	The client user name, or unknown.

TCP wrappers provide a couple of utilities to check your configuration. The first is `tcpdchk`. It is used to check your syntax and make sure the rules do what you are expecting. Table 17-3 lists the `tcpdchk` command-line options. The following is sample output from `tcpdchk`.

Using network configuration file: /etc/inetd.conf

```
>>> Rule /etc/hosts.allow line 7:
daemons:  in.ftpd
clients:   .the-nashes.net
access:    granted
```

```
>>> Rule /etc/hosts.deny line 9:
daemons:  ALL
clients:  ALL
access:    denied
```

Table 17-3
tcpdchk Options

Option	Function
-a	Report rules that allow access without an explicit ALLOW statement.
-d	Check the <code>hosts.allow</code> and <code>hosts.deny</code> files in the current directory instead of those in <code>/etc</code> .
-i <i><inetd_conf></i>	Path to your <code>inetd.conf</code> file, if not using the current one in <code>/etc</code> .
-v	Display all access control rules. By default no output is displayed if no problems are found.

Another tool provided is `tcpdmatch`. The syntax is as follows:

```
tcpdmatch [-d] [-i inet_conf] daemon client
```

The `-d` and `-i` options are the same as with `tcpdchk`. The `tcpdmatch` tool allows you to provide theoretical connections to your system and watch your results. For example, using the rules:

```
hosts.deny
ALL      :    ALL

hosts.allow
in.ftpd  :    .the-nashes.net
```

testing a connection from `norbert.the-nashes.net` to the FTP and Telnet servers would produce the following:

```
[root@redhat /etc]# /usr/sbin/tcpdmatch in.ftpd
jason@norbert.the-nashes.net
client:  hostname norbert.the-nashes.net
client:  address 216.254.90.133
client:  username jason
```

```

server:   process  in.ftpd
matched: /etc/hosts.allow line 7
access:  granted

[root@redhat /etc]# /usr/sbin/tcpdmatch in.telnetd
jason@norbert.the-nashes.net
client:  hostname norbert.the-nashes.net
client:  address  216.254.90.133
client:  username jason
server:  process  in.telnetd
matched: /etc/hosts.deny line 9
access:  denied

```



TCP wrappers cannot be used with applications that use the UDP.

SUID security issues

Applications that are set as SUID root can be used to gain root access to the system. Only applications that require SUID should be set so. If the application does not have to run as root, you can create a separate account for just that application and assign rights only where needed. This way if someone uses the application to gain access to the system, they have only the account's permissions and cannot do harm to the rest of the system.

It is useful to occasionally check to see which files are SUID on the system. It is an even better idea to set up a cron job to do this periodically and have it mail you the results. The following command would accomplish this:

```
/usr/bin/find / -user root -perm -4000 -print | /bin/mail -s
"Files that are SUID root" <mail address>
```

If you know that a file system has no applications that need to be SUID, consider mounting it with the `-o nosuid` option. This way an application cannot be accidentally installed SUID without your knowledge.



Some systems automatically remove the SUID status of a shell script when it is run to reduce the security risk.

Managing packages

Many administrators and users now download precompiled binary packages for software. Downloading these packages is much more convenient than downloading source code and having to compile it. The problem is that unlike source code, which can be manually verified for backdoors or other security problems, binary files must be taken as is. To minimize this risk, you should get binary packages only from the creator's Web site and check them for validity.

RPM includes functions to let you check the integrity of a package to make sure it was downloaded correctly and not tampered with. The `-K` or `-checksig` option validates the integrity of a package using MD5 or GnuPG.

```
rpm -K package_file.rpm
```

For this to work the following steps must be taken:

1. Install the GnuPG application, available from <http://www.gnupg.org>.
2. Add the public key for the appropriate package maintainers to your key ring using the `gpg -import` command.

If the package validates correctly, `rpm` will output a message similar to the following:

```
[root@redhat /root]# rpm -K wget-1.5.3-6.src.rpm
wget-1.5.3-6.src.rpm: md5 gpg OK
```

Debian does not offer such an integrated approach. The usual method is to use the `md5sum` tool and compare its output to a published value. If the value matches, the software is most likely not compromised.

Using setgid

Changing a file to be `setgid` causes it to run with the group's permissions, but this setting has a different meaning with directories. Normally, when a user creates a file in a directory, the group owner of the file is set to that of the user. When `setgid`, or `SGID`, is set on the directory, the file has the group owner of the directory, not the user.



The `SGID` bit is used to create a shared directory.

Red Hat uses this plus a `umask` of `002` to allow you to create directories to share files. Normally, the `umask` is set to `022` on most Linux distributions. With a `umask` of `022`, the permissions for a new file are `755`, so the group does not have write access. If a new file is created in a directory that is `setgid`, the group owner is that of the directory, but members of the group still cannot write to the file. Red Hat gets around this with its `umask` of `002`. Newly created files have the group owner of the directory, and `775` permissions.

This is useful when making a directory to hold shared files. Just add all of the users who need to access the files to a group that also owns the shared directory and set the directory `setgid`.

How and why not to use the r commands

The Berkeley r commands are tools that allow you to log in, run commands, or copy files remotely across a network. They have been in use for many years. These commands are very insecure and should not be used in most circumstances. So why cover them in the security chapter? Because these commands are in wide use and can be replaced by the much more secure SSH. Many of the files and ideas that SSH uses are similar to those in the Berkeley r commands. This makes for a very easy migration.

**Caution**

The Berkeley r commands are very insecure and should be replaced with SSH.

The r commands were developed to be used when the world was a nicer place. Now that there are people on the Internet who cannot be trusted, these commands are a liability. They depend on hostnames, which can be faked with compromised DNS, and pass information in clear text, which can be picked up using a packet sniffer.

Configuring access to the r commands

Two files handle access from remote systems to local accounts with the r commands: `.rhosts` and `hosts.equiv`. You must be extremely careful when configuring these files, or you can open up the entire machine to remote users.

The `.rhosts` file

The `.rhosts` file is used to allow or deny access to remote users of the r commands. It is stored in a user's home directory and has the following syntax:

```
hostname [username]
```

For example, assume the user `peggy` had the following `.rhosts` file in her home directory:

```
Norbert    jason
deedee    angie
dexter
```

This `.rhosts` file would allow the user `jason` from `norbert` and the user `angie` from `deedee` access to her account without a password. It would also allow user `peggy` from the host `dexter`, since no other user name was specified. The `.rhosts` file can be owned only by the user `peggy` or `root`. No one else can have write permission, or the file will be ignored. Some security can be gained by using fully qualified domain names instead of just hostnames.

The `hosts.equiv` file

The `/etc/hosts.equiv` file is used to allow or deny access to the `r` commands for other systems. The syntax is as follows:

```
[+ | -] host [username]
```

By specifying a hostname, any user from the host can access an account of the same name on the local host without using a password. The optional user name gives a user on the remote host access to any account except for root on the local system. A preceding `-` sign makes all users enter a password for access, even for accounts with the same name.



If only a `+` sign is used on a line, any user from any host can connect to the system. Watch out for typos!

The `r` commands

The `r` commands provide remote access to another system. They are useful when you need to copy files, log in, and execute commands on the remote server or when you need to check to see if other users are logged in.

Using `rlogin`

The `rlogin` utility is used to remotely log in to a remote system. The syntax is as follows:

```
rlogin [-l username] host
```

By default the local user name is used to log in remotely. A password is required unless the other side is set up to trust the client system through standard `rhost` security.

Using `rcp`

The `rcp` utility is used to copy files between systems, usually without the need to log in or provide a password. The syntax is as follows:

```
rcp [-r] remote-host:source-file new-file
```

The `-r` option specifies a recursive copy of a directory. The command-line options can be reversed to copy a local file to a remote system.

Using `rsh`

The `rsh` utility is used to execute a single command on a remote system. The syntax is as follows:

```
rsh [-l username] host command
```

Using `rwho`

The `rwho` utility is used to show logged in users on the local network. No command-line options are used, except for `-a`, which shows all users, even if they have been idle.

Using `ruptime`

The `ruptime` utility shows the uptime of computers on the local network, along with their load average.

Using SSH

Now that you have learned how to use the `r` commands, you can replace them with SSH, which provides secure equivalents to most of them. Secure Shell, or SSH, is a secure remote communications suite that provides tools for copying, remote execution, and terminal services. SSH provides a “drop in” replacement for the Berkeley `r` commands, and most of syntax is similar or the same. SSH protects against IP spoofing, source routing attacks, DNS spoofing, and network sniffers. The way it does this is through encryption and public/private keys.

SSH has a complicated history, made clear by a new project. SSH was originally an open project that was made in to a commercial product. The commercial product also used RSA encryption, which was patented in the United States. A version of SSH without RSA was available for noncommercial use, but this sometimes caused problems with servers that required RSA. On top of this, two versions of the SSH protocol, normally called v1 and v2, now exist, which caused other compatibility problems.



OpenSSH is available at <http://www.openssh.com>. It is free and released under the BSD license.

The solution to all of this is the OpenSSH suite. OpenSSH is an open source, non-patent encumbered distribution of SSH that is derived from the original open SSH. The RSA patent has now expired in the United States, which should make distribution easier.

Version 1 authentication methods

SSH v1 has several different authentication methods. These are as follows:

1. The name of the client host is checked against the `hosts.equiv` and `shosts.equiv` files. If the host has an entry, and the user name is the same on the client and server, the user is logged in without a password. The `RhostsAuthentication` option must be enabled. This is the most insecure method, since it depends on the `hosts.equiv` and `shosts.equiv` files.

2. The primary method for authentication uses `.rhosts`, `.shosts`, `hosts.equiv`, and `shosts.equiv`, along with RSA-based host authentication. The host files are checked and if login would be allowed, the server checks the client's host key from `/etc/ssh_known_hosts` and `~/.ssh/known_hosts`. This stops IP, DNS, and route spoofing. Spoofing is when another host pretends to be a trusted system. This method still depends on the somewhat insecure host files. The `RhostsRSAAuthentication` option must be enabled for this method.
3. The third authentication method uses RSA-based authentication with public key cryptography. A user's public key is stored in a `~/.ssh/authorized_keys` file on a remote system. When the user connects, the client SSH tells the server which key pair it wants to use for authentication. The server then checks to see if the key is permitted, and if so, it sends the client SSH process a challenge, which is a random number encrypted by the user's public key. This challenge can be decrypted only by using the correct private key. This method is the most secure since it uses a well established mechanism, and the private key is never divulged.

Version 2 authentication methods

The standard SSH v2 authentication method is similar to the third v1 method. Instead of using just RSA, it uses DSA, Digital Signature Algorithm, or RSA. The client uses the `~/.ssh/id_dsa` file instead of `~/.ssh/identity`. The server uses `~/.ssh/authorized_keys2` instead of `~/.ssh/authorized_keys`.

If the standard method fails, an encrypted password can be sent to the remote system to prove the user's identity.

Using the SSH files

SSH uses several files to store keys and information. These are shown in Table 17-4. By default you are prompted for a password when connecting to a remote system with SSH, but to increase security you can use public key cryptography.

Briefly, public key cryptography provides a solution to the most common problem with conventional encryption. In conventional encryption, usually called symmetric cryptography, the sender and receiver of a message both use the same encryption key. A key is a piece of data used to encrypt and decrypt a message. If a third party finds out this shared key, they can decrypt secret messages, as well as encrypt their own messages to pass off as one of the trusted participants. Public key cryptography solves this by giving each person two keys. The first key is known as the public key. It is meant to be published to the world and used to encrypt any messages sent to you. The second key assigned is a private key. This key is kept secret and should be known only by the person it was issued to. It can decrypt messages that were encrypted with the matching public key. This way you can give out your public key and have people encrypt messages to you without giving away the key that also decrypts those messages.

Table 17-4
SSH Files

File	Function
<code>~/.ssh/known_hosts</code>	When a user logs in to a remote server they are sent a key. These keys are stored in this file. Should the user later receive a different key from a server than the one that is recorded here, he or she will be shown a warning message stating that someone may be trying to act as the remote server.
<code>~/.ssh/identity</code> or <code>~/.ssh/id_dsa</code>	Contains the user's private RSA and DSA information. This information should be readable only by the user. If someone else gets the information from these files, he or she can masquerade as the user.
<code>~/.ssh/identity.pub</code> or <code>~/.ssh/id_dsa.pub</code>	Contains the public key information for the user.
<code>~/.ssh/config</code>	The configuration file for the <code>ssh</code> client that can override some settings from the system-wide configuration file.
<code>~/.ssh/authorized_keys</code>	Lists the RSA keys that can be used for logging in as this user.
<code>~/.ssh/authorized_keys2</code>	Lists the public keys (DSA/RSA) that can be used for logging in as this user.
<code>/etc/ssh_known_hosts</code> or <code>/etc/ssh_known_hosts2</code>	Similar to the user's list of known hosts. This file can be set up by the administrator so that a central repository of known keys is used as a precaution against spoofing. <code>/etc/ssh_known_hosts</code> contains RSA keys <code>/etc/ssh_known_hosts2</code> contains DSA or RSA keys for protocol version 2.
<code>/etc/ssh_config</code>	The system-wide <code>ssh</code> client configuration file.
<code>/etc/sshd_config</code>	The system-wide <code>ssh</code> server configuration file.
<code>~/.rhosts</code>	Used in <code>.rhosts</code> authentication to list the host/user pairs that are permitted to log in. This file is a carry-over from the Berkeley <code>r</code> commands, which SSH replaces. If your system does not use the old <code>r</code> commands, you should use the <code>shosts</code> file.
<code>~/.shosts</code>	Used exactly the same way as <code>.rhosts</code> . The purpose for having this file is to be able to use <code>rhosts</code> authentication with SSH without permitting login with <code>rlogin</code> or <code>rsh</code> .

Continued

Table 17-4 (continued)

<i>File</i>	<i>Function</i>
<code>/etc/hosts.equiv</code>	Used during <code>.rhosts</code> authentication. It contains canonical hosts names, one per line. This is another carry-over from the Berkeley <code>r</code> commands. If your system does not use them you should use the <code>shosts.equiv</code> file.
<code>/etc/shosts.equiv</code>	Processed exactly as <code>/etc/hosts.equiv</code> . This file may be useful to permit logins using SSH but not using <code>rsh</code> or <code>rlogin</code> .
<code>/etc/sshr</code>	Commands in this file are executed by SSH when the user logs in just before the user's shell (or command) is started. This is a system-wide configuration file.
<code>~/.ssh/rc</code>	Commands in this file are executed by SSH when the user logs in just before the user's shell (or command) is started.
<code>~/.ssh/environment</code>	Contains additional definitions for environment variables.

Configuring the SSH server

The SSH server is made up of the `sshd` daemon. It uses the `sshd_config` file for configuration, normally stored in `/etc/ssh`. Here is a piece of an `sshd_config` file:

```
Port 22
#Protocol 2,1
ListenAddress 0.0.0.0
#ListenAddress ::
HostKey /etc/ssh/ssh_host_key
ServerKeyBits 768
LoginGraceTime 600
KeyRegenerationInterval 3600
PermitRootLogin yes
#
# Don't read ~/.rhosts and ~/.shosts files
IgnoreRhosts yes
# Uncomment if you don't trust ~/.ssh/known_hosts for
RhostsRSAAuthentication
#IgnoreUserKnownHosts yes
StrictModes yes
X11Forwarding no
X11DisplayOffset 10
PrintMotd yes
KeepAlive yes
```

The format is very simple: each line has an entry and a value. The possible entries are listed in Table 17-5.

Table 17-5
sshd_config Entries

Entry	Function	Default
AFSTokenPassing <yes no>	Specifies whether an AFS token may be forwarded to the server. AFS is a file service sometimes used with the Kerberos security system.	Yes
AllowGroups <group names>	Allows login only by groups listed in this entry.	Login is allowed for any group.
AllowTcpForwarding <yes no>	Enables or disables TCP traffic forwarding.	Yes
AllowUsers <user list>	Limits login to certain user names.	Login is allowed for any user.
Ciphers <cipher list>	Specifies the ciphers, or encryption algorithms, that are used for v2 of the SSH protocol.	3des-cbc, blowfish-cbc, arcfour, cast128-cbc
CheckMail <yes no>	Enables or disables the mail check when a user logs in.	No
DenyGroups <group list>	Lists groups not allowed to log in via SSH.	Any group is allowed to log in.
DenyUsers <user list>	Lists users not allowed to log in via SSH.	Any user is allowed to log in.
PubkeyAuthentication <yes no>	Specifies whether public key authentication is allowed.	Yes
GatewayPorts <yes no>	Specifies whether remote hosts can connect to ports forwarded for the client.	No
HostKey <file name>	Specifies the file containing the private host keys.	ssh_host_key
IgnoreRhosts <yes no>	Enables or disables the use of .rhosts and .shosts. This does not affect hosts.equiv and shosts.equiv.	Yes
IgnoreUserKnownHosts <yes no>	Specifies whether to ignore users' known_hosts files.	No

Continued

Table 17-5 (continued)

Entry	Function	Default
KeepAlive <yes no>	Sends keep-alive messages to the client.	Yes
KerberosAuthentication <yes no>	Enables or disables Kerberos authentication. Kerberos is a very secure system designed for single login across many different hosts and services.	Yes
KerberosOrLocalPasswd <yes no>	If Kerberos authentication fails, uses the local password file.	Yes
KerberosTgtPassing <yes no>	Specifies whether a Kerberos TGT may be forwarded to the server.	No
KerberosTicketCleanup <yes no>	Specifies whether to automatically destroy the user's ticket cache file on logout.	Yes
KeyRegenerationInterval < <i>n</i> seconds>	Specifies how often to regenerate the SSH server key.	3600 seconds.
ListenAddress < <i>ip address</i> >	Specifies which local addresses to listen on.	All addresses.
LoginGraceTime < <i>n</i> seconds>	If the user does not log in within the specified time, he or she is disconnected.	600 Seconds.
LogLevel < QUIET, FATAL, ERROR, INFO, VERBOSE, DEBUG>	Sets the logging level.	INFO
MaxStartups < <i>n</i> >	The number of unauthenticated connections allowed at one time.	10
PasswordAuthentication <yes no>	Enables or disables password authentication.	Yes
PermitEmptyPasswords <yes no>	Specifies if the server will allow blank passwords.	No
PermitRootLogin <yes no without-password>	Enables or disables remote login of the root account.	Yes
PidFile < <i>file name</i> >	Specifies the file that holds the process ID information.	/var/run/sshd.pid
Port < <i>n</i> >	Specifies the port number that SSH listens on.	22
PrintMotd <yes no>	Specifies whether or not to show the /etc/motd file when a user logs in.	Yes

Entry	Function	Default
Protocol <1, 2>	Specifies the protocol(s) to support. Both protocols can be supported by the daemon at the same time.	1
RhostsAuthentication <yes no>	Specifies whether to allow authentication via <code>.rhosts</code> and <code>hosts.equiv</code> .	No
RhostsRSAAuthentication <yes no>	Specifies whether <code>rhost</code> authentication is allowed when used with RSA host authentication.	No
ServerKeyBits <n>	The number of bits to use in the server key.	768
SkeyAuthentication <yes no>	Enable or disable <code>skey</code> authentication.	Yes
StrictModes <yes no>	Specifies whether <code>sshd</code> should check file ownership and permissions of user files before trusting.	Yes
Subsystem	Configures an external subsystem, such as <code>sftp</code> .	None
SysLogFacility <DAEMON, USER, AUTH, LOCAL0, LOCAL1, LOCAL2, LOCAL3, LOCAL4, LOCAL5, LOCAL6, LOCAL7>	The <code>syslog</code> facility to log messages to.	AUTH
UseLogin <yes no>	Specifies whether or not to use the <code>login</code> command for interactive sessions.	No
X11DisplayOffset <n>	Specifies the first display number used for X11 forwarding.	10
X11Forwarding <yes no>	Enables or disables X11 forwarding.	No
XAuthLocation <path>	Specifies the location of the <code>Xauth</code> program, used for X11 forwarding.	<code>/usr/ X11R6/ bin/xauth</code>

SSH is normally run via a script, but can be used with `inetd`. Table 17-6 shows the possible options for `sshd`. When `sshd` executes, it generates a key for the server, which can take several seconds. If SSH is used with `inetd`, that key must be generated every time a connection is received, since `sshd` does not stay resident. For this reason it is not recommended to use `inetd`. The only advantages to using `inetd` are the use of less memory and support for TCP wrappers.

Table 17-6
sshd Options

Option	Function
-b <n>	Specifies the number of bits for the server key.
-d	Enables debug mode.
-f <file>	Specifies the path for an alternate configuration file, instead of <code>sshd_config</code> .
-g <n>	Specifies the login grace time.
-h <file>	Specifies the host key filename.
-i	Specifies that <code>sshd</code> is being run from <code>inetd</code> . This causes <code>sshd</code> to not stay resident when a client disconnects.
-k <time in seconds>	Specifies how often to regenerate the server key.
-p <n>	Specifies the port number to use.
-q	Enables quiet mode, which tells the daemon not to send any information to the system log.
-u <n>	Specifies the length of the hostname in <code>utmp</code> .
-Q	Specifies that <code>sshd</code> does not display an error if support for RSA is missing.
-V <client protocol ID>	Enables SSH v2 compatibility mode.

Using the SSH client tools

The OpenSSH distribution provides two client tools for connecting to remote systems interactively and for copying files. Unlike the Berkeley `r` command tools, these are secure and pass no information in clear text.

Using `ssh`

The `ssh` tool replaces `telnet`, `rlogin`, and `rsh`. The syntax for its use is as follows:

```
ssh [-afgknqtvxACNPTX246] [-c cipher_spec] [-e escape_char]
[-i identity_file] [-l login_name] [-o option] [-p port]
[-L port:host:hostport] [-R port:host:hostport] [hostname
|user@hostname] [command]
```

The `ssh` client does more than just provide remote interactive connections. It can also be used to forward network ports securely between systems. This allows you to run insecure applications over a secure channel. Displaying back X11 applications is a popular use for this and has integrated support already built in. Table 17-7 lists the command-line options for `ssh`.

Table 17-7
ssh Options

Option	Function
-a	Disables forwarding of the authentication agent connection. The authentication agent is covered in the next section.
-A	Enables forwarding of the authentication agent connection.
-c <blowfish 3des>	Specifies the v1 cipher to use for this connection. Blowfish and 3DES, known as triple DES, are two popular and well tested encryption methods.
-c <3des-cbc,blowfish-cbc,arcfour,cast128-cbc>	Specifies the v2 cipher to use for this connection. Several ciphers are available, but the most common are 3DES and Blowfish.
-e <character>	Sets the escape character for the session so that you can exit out of <code>ssh</code> quickly.
-f	Tells <code>ssh</code> to go to the background right before command execution. This is useful if you are instructing <code>ssh</code> to execute a command at connect and want <code>ssh</code> itself to go to the background.
-g	Allows remote hosts to connect to local forwarded ports.
-i <file>	Specifies the file from which the private key is used for RSA or DSA authentication.
-k	Disables forwarding of Kerberos tickets and AFS tokens.
-l <login name>	Specifies the user name to use on the remote system. By default it is the current user name.
-n	Redirects <code>stdin</code> from <code>/dev/null</code> . This is required when <code>ssh</code> is directed to the background.
-N	Does not execute a remote command (version 2 only). This is useful if you are setting up <code>ssh</code> to only forward port connections.
-o <option>	Specifies options using the configuration file syntax for which there is no command-line option.
-p <n>	Specifies the port number to connect to.
-P	Uses a nonprivileged port for outgoing connections. This disables <code>RhostsAuthentication</code> and <code>RhostsRSAAuthentication</code> .
-q	Enables quiet mode.

Continued

Table 17-7 (continued)

Option	Function
-v	Enables verbose mode.
-x	Disables X11 forwarding.
-X	Enables X11 forwarding.
-C	Enables data compression to speed up slow network connections.
-L <i>port:host:hostport</i>	Specifies that the given port on the local (client) host is to be forwarded to the given host and port on the remote side. This works by allocating a socket to listen to <i>port</i> on the local side. Whenever a connection is made to this port, the connection is forwarded over the secure channel, and a connection is made to <i>host port hostport</i> from the remote machine.
-R <i>port:host:hostport</i>	Specifies that the given port on the remote (server) host is to be forwarded to the given host and port on the local side. This works by allocating a socket to listen to <i>port</i> on the remote side. Whenever a connection is made to this port, the connection is forwarded over the secure channel, and a connection is made to <i>host port hostport</i> from the local machine.
-2	Forces SSH v2.

For example, to log into the remote system named `europa` as user `jason`, you would use the following:

```
[root@redhat ~]# ssh -l jason europa
The authenticity of host 'europa' can't be established.
RSA key fingerprint is
1d:ee:ee:39:2e:4a:af:5d:8f:37:b2:8e:00:03:35:b7.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'europa' (RSA) to the list of known
hosts.
jason@europa's password:
Last login: Wed Dec  6 10:48:19 2000 from :0 on pts/3
Linux europa 2.4.0-test10 #1 Wed Nov  8 16:57:08 EST 2000 i686
unknown
```

Using scp

The `scp` tool replaces `rcp`, but uses `ssh` so it is secure. The syntax for `scp` is as follows:

```
scp [-pqrvc46] [-S program] [-P port] [-c cipher] [-i
identity_file] [-o option] [[user@]host1:]file1 [...]
[user@]host2:]file2
```

Table 17-8 lists the command-line options for `scp`. For example, to copy a file to the computer `europa` as user `jason`, you would use:

```
[root@redhat ~]# scp -r MyDir jason@europa:/home/jason
jason@europa's password:
Document          100% |*****|          97 KB    00:00
README            100% |*****|       99986    00:00
```

Table 17-8
scp Options

Option	Function
<code>-c</code>	Selects the cipher to use for encryption.
<code>-i <file></code>	Specifies the identify file holding the user's private key for RSA authentication.
<code>-p</code>	Preserves modification times, access times, and modes from the original file.
<code>-r</code>	Performs a recursive copy of a directory.
<code>-v</code>	Enables verbose mode.
<code>-B</code>	Enables batch mode.
<code>-q</code>	Disables the progress meter usually shown when copying files.
<code>-C</code>	Enables data compression.
<code>-p <n></code>	Specifies the remote port to connect to.
<code>-o <option></code>	Passes an option directly to <code>ssh</code> .

User key management

To use the public key method of authentication, a user must create a *key pair*. This is done with the `ssh-keygen` tool. The syntax for `ssh-keygen` is as follows:

```
ssh-keygen [-q] [-b bits] [-t type] [-N new_passphrase]
[-C comment] [-f output_keyfile]
ssh-keygen -p [-P old_passphrase] [-N new_passphrase]
[-f keyfile]
ssh-keygen -x [-f input_keyfile]
ssh-keygen -X [-f input_keyfile]
ssh-keygen -y [-f input_keyfile]
ssh-keygen -c [-P passphrase] [-C comment] [-f keyfile]
ssh-keygen -l [-f input_keyfile]
ssh-keygen -R
```

Table 17-9 shows the command-line options for `ssh-keygen`. If run by itself, it generates a new key for the current user and prompts for the storage location and passphrase. For example:

```
[root@redhat ~]# ssh-keygen
Generating RSA keys: Key generation complete.
Enter file in which to save the key (/root/.ssh/identity):
/root/.ssh/identity already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/identity.
Your public key has been saved in /root/.ssh/identity.pub.
The key fingerprint is:
82:9d:58:71:2c:f9:56:51:d4:51:50:84:77:9d:f7:36
root@redhat.the-nashes.net
```

Table 17-9
ssh-keygen Options

Option	Function
-b <n>	Specifies the key bit size. The default is 1024.
-c	Changes the comment in the public and private key files. This comment information is kept for the user's benefit and is not used by the system.
-f	Specifies the filename for the key.
-l	Shows the fingerprint of the specified key file.
-p	Changes the passphrase associated with a key file.
-q	Enables quiet mode.
-t <type>	Specifies the type of key to create. Version 1 supports RSA1, and version 2 supports RSA and DSA.
-C <comment>	Specifies the new comment.
-N <passphrase>	Specifies the new passphrase.
-P <passphrase>	Specifies the old passphrase.
-R	RSA function check.
-X	Reads in an unencrypted key and outputs a new encrypted version.
-y	Reads in a private key and outputs a public key.

Some users find it annoying to have to constantly enter their passphrase when connecting to a remote host. To help alleviate this problem, they can use the `ssh-agent` and `ssh-add` tools. The `ssh-agent` tool is used to hold private keys for public key authentication. The normal procedure is that `ssh-agent` is started at the beginning of a login or X11 session, and all other applications or windows are clients. From any of these clients, an SSH session can be made to a remote system, and the passphrase must be entered only once. The syntax for `ssh-agent` is as follows:

```
ssh-agent command args ...
ssh-agent [-c | -s]
ssh-agent -k
```

Table 17-10 lists the `ssh-agent` command-line options.

Table 17-10 ssh-agent Options	
<i>Option</i>	<i>Function</i>
-c	Generates C Shell commands. This is the default if the shell looks to be a C-like shell.
-s	Generates Bourne commands. This is the default if the shell does not look to be a C-like shell.
-k	Kills the current <code>ssh-agent</code> process.

The agent does not open any keys by itself; they are loaded with the `ssh-add` tool. To open a new shell with the `ssh-agent` you can use the following:

```
ssh-agent $SHELL
```

This causes all connections made from the shell to be handled by the agent. To start X11 with agent support you can use the following:

```
xsh-agent startx
```

This way any SSH connections made from within X11 will use the agent and not require the passphrase. To add your private keys to `ssh-agent`, use `ssh-add`. The syntax is as follows:

```
ssh-add [-lLdD] [file...]
```

Table 17-11 lists the command-line options for `ssh-add`. If no options are specified, `ssh-add` loads all private keys for the current user from `~/.ssh/identity` and prompts him or her for his or her passphrase if he or she has one.

Table 17-11
ssh-add Options

<i>Option</i>	<i>Function</i>
-l	Lists the fingerprints of all identities currently represented by the agent.
-L	Lists public key parameters for all identities represented by the agent.
-d	Removes an identity from the agent.
-D	Removes all identities from the agent.

To bring this all together, you take the following steps to set up public key authentication.

1. Have the user run `ssh-keygen` and create a new identity.
2. Copy the user's public key (from `identity.pub` or `id_dsa.pub`) to the remote host's `authorized_keys` file (`authorized_keys` and `authorized_keys2`). These filenames depend on whether you are using SSH v1 or v2.
3. Run `ssh-agent` with a new shell or X11.
4. Run `ssh-add` to load the user's identities.
5. Connect to the remote host, without needing to enter the passphrase again.

Providing Host Security

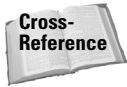
Objective

1.14 Security

- **Setup host security.** Implement shadowed passwords, turn off unnecessary network services in `inetd`, set the proper mailing alias for root and setup `syslogd`, monitor CERT and BUGTRAQ, update binaries immediately when security problems are found

You can do several things to increase the security on a system. These include the following:

- ♦ Using shadow passwords
- ♦ Removing unused services
- ♦ Blocking unwanted connections
- ♦ Customizing PAM
- ♦ Adding security updates



Setting up and managing syslogd is covered in Chapter 11. Using mail aliases is discussed in Chapter 16.

Using shadow passwords

A system that does not use shadow passwords stores all passwords in the `/etc/passwd` file. Look at the standard permissions on this file:

```
-rw-r--r-- 1 root root 760 Nov 4 17:21 /etc/passwd
```

See a problem? All users must have read access to this file so that they can log in. If a user can see the encrypted passwords, they can copy the file and use a password cracker to decrypt the password back to plain text.

Shadow passwords fix this problem by actually keeping the encrypted passwords in the `/etc/shadow` file. The permissions on this file are as follows:

```
-r----- 1 root root 703 Nov 2 22:49 /etc/shadow
```

Only root can read this file, so other users cannot snoop around and get the encrypted passwords to work on later. The problem is that some older utilities do not function with shadow passwords. If you use a current distribution and services, you should definitely enable shadow passwords.



Most Linux distributions ship with shadow passwords enabled by default.

The `/etc/shadow` file has nine fields that contain the following information:

- ♦ Login name
- ♦ Encrypted password
- ♦ The number of days since January 1, 1970, that the password was last changed.
- ♦ The number of days until the password can be changed.
- ♦ The number of days after which the password must be changed (password aging).
- ♦ The number of days before the password expires that the user is warned.
- ♦ The number of days after the password expires that the account is disabled.
- ♦ The number of days since January 1, 1970, that the account was disabled.
- ♦ A field reserved for later use.

The following is an example of a shadow password file:

```
root:$1$Vgk34Ae1dk38XyMuzPot5Jiod9Rzm0WY.:11376:0:99999:7:::  
jason:$1$nm2dkL9Z$298EbtINXUPMKwXxrahFA.:11406:0:99999:7:::  
angie:$1$1639jnrB$mr9z0eE3hzQzuN7ehKvoT1:11406:0:99999:7:::
```

The `pwconv` tool is used to convert normal passwords to shadow passwords. Since some information needed is not in `/etc/passwd`, `pwconv` uses the `PASS_MIN_DAYS`, `PASS_MAX_DAYS`, and `PASS_WARN_AGE` from `/etc/login.defs` to populate these needed fields. To undo the conversion you can use `pwunconv`.

The `/etc/group` file can also be converted to use shadow passwords using `grpconv`. This tool creates a `gshadow` file that stores the encrypted passwords. To undo the change use the `grpunconv` tool.



Shadow passwords are also covered in Chapter 10.

Removing unused services

By default, most Linux distributions come with far more services enabled than the user needs. Distribution makers enable servers to reduce support calls from users that expect a service to be there that isn't. The problem with this practice is that most users end up running services they do not need, and they don't know it. When a security problem is found with a service, the users do not know they need to update their system, which leads to a lot of vulnerable systems.

The lesson here is to install and run only services that you need. This begins during installation. It's definitely easier to choose one of the automated install paths that most Linux distributions offer, but you get much more granular control when doing a customized setup and picking and choosing the installed packages. The time spent here saves you from having to remove and disable services later.

The first step is removing unneeded services in the scripts directory where some services are started. These are `/etc/rc.d/rc<runlevel>.d` for Red Hat and `/etc/rc<runlevel>.d` for Debian. Remove any unneeded scripts for the runlevel that your system normally operates in.

The next place to look is in the `inetd.conf` file. It is easy to overlook this and end up with a number of services running that you did not anticipate. Any unneeded entries should be commented out. If you do not require any services listed in `inetd.conf`, there is no reason to even run `inetd`. Remove its script from your runlevel directory.

The final step is to do a complete process list, such as `ps aux`, and look at all running daemons and processes. Make sure you know what each process does and if it is required. It is common for a normal workstation to need almost no server

processes running, and this provides the best security. If a server service is required, make sure it is secured properly. If the service is inherently insecure, such as Telnet or the Berkeley r tools, replace them with secure alternative such as SSH.

Blocking unwanted connections with IP chains

Linux has a package called *IP chains* that allows you to set up a firewall on a Linux system. This capability is useful for protecting an entire network at its entry point or blocking certain connections on an individual workstation. When a network interface receives a packet, it puts the packet through a chain of rules. These rules are made up of evaluations that decide if a packet is denied, if it is allowed through, or if the default action is taken.

These chains and rules are configured using the `ipchains` tool. The syntax for `ipchains` is as follows:

```
ipchains -[ADC] chain rule-specification [options]
ipchains -[RI] chain rulenum rule-specification [options]
ipchains -D chain rulenum [options]
ipchains -[LFZNX] [chain] [options]
ipchains -P chain target [options]
ipchains -M [ -L | -S ] [options]
```

The `ipchains` tool can take many command-line options and parameters. They are broken up into Tables 17-12 and 17-13 for ease of reading and to separate the different functionalities.

Table 17-12
ipchains Commands

Command	Function
<code>-A, --append</code>	Appends a rule to the end of the specified chain.
<code>-D, --delete <rule number> <matching rule></code>	Deletes one or more rules from the specified chain.
<code>-R, --replace <rule number></code>	Replaces the specified rule with another.
<code>-I, --insert <number of rule to insert before></code>	Inserts a new rule before the specified rule.
<code>-L, --list [chain name]</code>	Lists the rules in the specified chain, or all chains.
<code>-F, --flush <chain name></code>	Clears all rules in the specified chain.
<code>-Z, --zero <chain name></code>	Zeroes all counters in the specified chain.
<code>-N, --new-chain <chain name></code>	Creates a new user-defined chain with the specified name.

Continued

Table 17-12 (continued)

Command	Function
-P, --policy	Sets the default target for a chain.
-M, --masquerading	Used with the -L function, lists the current masqueraded connections. Used with the -S option, sets the kernel masquerading parameters.
-S, --set <i>tcp tcpfin udp</i>	Changes the timeout values used for masquerading. The timeouts are as follows: <i>tcp</i> indicates TCP session timeout; <i>tcpfin</i> indicates TCP session timeout after receiving a FIN packet; and <i>udp</i> indicates UDP packet timeout. This option is only used along with the -M flag.
-C, --check	Checks the given packet against the selected chain. Useful for testing and debugging.
-h	Help.

Table 17-13
ipchains Parameters

Option	Function
-p, --protocol [!]	The protocol of the rule or packet to check. Possible values are <i>tcp</i> , <i>udp</i> , <i>icmp</i> , or <i>all</i> . Other values listed in <i>/etc/protocols</i> may be used. A ! preceding the protocol name inverts the test. For example, a rule with <i>!tcp</i> would apply to any protocol that is not <i>tcp</i> .
-s, --src, --source [!] <i>address[/mask] [!]</i> [<i>port[:port]</i>]	Specifies the source address and optional port of a connection.
--sport, --source-port [!] [<i>port[:port]</i>]	Used to separately define a source port or range of ports.
-d, --dst, --destination [!] <i>address[/mask] [!]</i> [<i>port[:port]</i>]	Specifies the destination address and optional port of a connection.
--dport, --destination-port [!] [<i>port[:port]</i>]	Separately specifies the destination port.
--icmp-type [!] < <i>typename</i> >.	Checks for ICMP type. Uses <i>-h icmp</i> to see the full list

Option	Function
<code>-j, --jump <target></code>	Specifies the target of a rule or what to do if a packet matches this rule. If this option is left out of a rule, the rule has no effect but can be used for logging purposes.
<code>-i, --interface [!] <interface name></code>	Specifies the interface through which a packet must enter for the Input chain or exit for the Output chain.
<code>[!] -f, --fragment</code>	Specifies that the rule is used only for the second and subsequent fragments of a packet.
<code>-b, --bidirectional</code>	Matches the packet in both directions. It is the same as adding two rules with the source and destinations reversed.
<code>-v, --verbose</code>	Enables verbose output for the <code>-L</code> command.
<code>-n, --numeric</code>	Shows addresses in numeric form, instead of resolving to hostnames.
<code>-l, --log</code>	Enables logging for packets that match this rule.

Chains and rules

When packets pass through the selected interface, they are evaluated against *rules*. A group of rules makes up a *chain*.

Chains

There are three main chains in IP chains: Input, Output, and Forward. A packet is processed through the Input chain when it is received by a network interface. The Output chain is used when a packet is sent out a network interface. The Forward chain is used when a packet is being forwarded from one interface to another, inside of a system.

In most cases you put your rules on the Input chain, since this provides the most protection. The chains are processed sequentially: When a packet is received, it goes through the Input chain; if it passes through that, it goes to the Forward chain; and if it survives that, it finally goes to the Output chain. If a packet is destined for the local system it will pass through only the Input chain.

The chains have a default policy that you set up. Normally, the default is `DENY`, and you must explicitly allow things through. The possible default policies are `DENY`, `REJECT`, and `ACCEPT`. A subtle but very important difference exists between `DENY` and `REJECT`. When a packet is met with `DENY`, it is simply dropped. No word is sent back to the source informing it of this action. The `REJECT` policy sends back an ICMP message to the source telling it that the packet was denied. The choice of which to use depends on the situation. Most people want their firewall to silently drop the packet and not report back any information to someone probing their site. The problem with this practice, however, is that it can sometimes cause confusion when troubleshooting a network problem.

Rules

Rules control what happens to a packet as it passes through a chain. Rules are made up of a condition that must be met and an optional target. When a packet is processed through a chain, it is matched against the rules in order until the first match is found. Once the match is found, the target is invoked, and the packet may then be forwarded along or rejected.

In some cases creating a rule that has no target is useful. This causes the packet to match, but no action is taken, and the packet continues through the chain. The idea is to use these phantom rules for logging and statistical purposes so you can see what type of traffic is being sent through the firewall.

The six basic targets that can be used in rules are as follows:

- ♦ ACCEPT
- ♦ DENY
- ♦ REJECT
- ♦ MASQ
- ♦ REDIRECT
- ♦ RETURN

ACCEPT is used to move the packet to the next chain or out of the network interface if it succeeds through the Output chain. DENY and REJECT were already discussed and cause a packet to be stopped, with varying results. MASQ is used only in the Forward chain and is discussed later in this chapter.

REDIRECT is used to redirect packets to a local socket, instead of the remote socket in their header. It is used only in the Input chain and finds use in applications such as transparent Web caches and proxies. This way, a user's Web request is sent through the cache without the user's even knowing.

RETURN is really used only in user-defined chains. This causes the packet to return to the calling chain. A user-defined chain is a chain just like Input, Output, and Forward, but has no default policy and is created by the user. It is called just like any other target, but by its name, which is set by the user. User-defined chains are used to increase modularity of complex chain structures. They allow the user to break out certain rule check sequences into other parts for ease of administration.

Rule order

The order in which rules are applied is extremely important. The target of the first matching rule is used on a packet. If the rules are not in the correct order, you may not get the expected result. Consider the following metarules:

```
Allow Anyone to WebServer at Port 80
Deny hacker.org to WebServer at Port 80
```

Since the processing of the chain stops at the first matching rule, everyone can get to the WebServer, even the guys from `hacker.org` you are trying to stop. The rules should be reversed so that specific rules should come before general rules.

Putting it all together

So now you understand rules, chains, conditions, targets, and rule order. Most likely you currently do not have any rules, but to be safe you should clear out the chains by using the `-F` option. These commands can be entered at the command line manually, but are usually put at the beginning of a configuration script:

```
ipchains -F input
ipchains -F output
ipchains -F forward
```

These commands should usually be at the beginning of any `ipchains` script, to be safe. The last thing you need are quirky results because of a forgotten rule hanging around. The next step is to set the default policy for all three of the standard chains by using the `-P` option. For example:

```
ipchains -P input DENY
ipchains -P output ACCEPT
ipchains -P forward ACCEPT
```

Most rules happen when a packet enters the system, so by default you deny everything and open the gate for certain packets with rules. To keep things simple, set the other two chains to `ACCEPT`. This may not be the best choice for a complicated setup, so use your judgment in the situation at hand. To make sure your changes worked, list the current state:

```
debian:/home/jason# ipchains -L
Chain input (policy DENY):
Chain forward (policy ACCEPT):
Chain output (policy ACCEPT):
```

Getting in the habit of also using the `-n` option when viewing the rules is a good idea. This option causes the system to display only IP addresses, and not to resolve them to hostnames. With an open ruleset resolving names may be fine, but when you start restricting access, you may no longer be able to access DNS names, and resolving names causes a very long timeout period.

Now you are ready to create your rules. Right now the system is like Fort Knox—nothing is getting in since the default Input chain's policy is set to `DENY`, and no rules are defined to get around that. First, allow the internal interfaces to receive packets. Assume this system has two network interfaces: the internal is `eth0` with a network range of `192.168.1.0/24` and the external is `eth1`. To allow the internal interfaces to receive packets, you would use the following:

```
ipchains -A input -i lo -j ACCEPT
ipchains -A input -i eth0 -j ACCEPT
```

Don't forget about the loopback interface named `lo`. A lot of services use this and may not function if it is denied. Certain ranges of IP addresses you should never see coming in, and it is usually a good idea to block those explicitly. For example:

```
ipchains -A input -i eth1 -s 10.0.0.0/8 -j DENY
ipchains -A input -i eth1 -s 172.16.0.0/12 -j DENY
```

These are private unroutable addresses that should never send packets to your incoming interface. To make sure no one tries to spoof you, you can block them. If you are using public addresses, IP addresses assigned to you by your ISP, you should also block those from coming in on the outside interface to prevent spoofing.

The next step is to open up any ports for services that you host, such as mail and SSH. For example:

```
ipchains -A input -i eth1 -p tcp -d 192.168.1.2 25 -j ACCEPT
ipchains -A input -i eth1 -dport 22 -j ACCEPT
```

These rules would allow outside hosts to connect to the internal mail server. It also allows SSH connections to the firewall itself from the outside world, in case of maintenance or a needed change when you're not at the workstation. The next examples are two rules that are used to allow replies to DNS queries. In the following examples, change the IP address of 1.2.3.4 to that of your ISP's DNS server.

```
ipchains -A input -i eth1 -p udp -s 1.2.3.4 53 -j ACCEPT
ipchains -A input -i eth1 -p tcp -s 1.2.3.4 53 -j ACCEPT
```

Finally, logging packets that are denied by your firewall is usually a good idea. This way you can get a look at who is trying to access blocked services. Sometimes this shows a configuration problem; other times it shows the people you are trying to keep out working to get in. To log all denied packets you would use the following:

```
ipchains -A input -j DENY -l
```

This reinforces the default policy of the chain, but adds the logging option.

IP masquerading

Many people are now getting broadband Internet connections to their home or office. They want to be able to share this connection with multiple PCs, but most ISPs give out only one IP address per connection. The solution to this is IP masquerading, or IP Masq. IP Masq lets all of your systems share one IP address. It's not really a part of firewalling or filtering, but is handled by IP chains.



The MASQ target is only used with the Forward chain.

Configuring IP Masq is very easy. Use the sample network and `ipchains` you set up in the previous section. Since the 192.168.1.0 network used is listed as private use only, packets from this network will not be routed on the Internet. To get these

systems on the Internet, you need to use IP Masq and have them hide behind a single real IP address. The command to enable IP Masq in this situation is as follows:

```
ipchains -F forward
ipchains -A forward -s 192.168.1.0/24 -d ! 192.168.1.0/24 -j
MASQ
```

That's it! This flushes any old rules out of the Forward chain and causes any packets from 192.168.1.0/24 that are not local (destination isn't on the same network) to be masqueraded as the real IP address assigned to the system's network interface.

Enabling forwarding

IP masquerading only functions if packet forwarding is enabled. Most Linux distributions ship with this feature disabled, for security. You can enable it in two ways. The first is to edit the `/etc/sysconfig/network` file in Red Hat and to add the following setting:

```
FORWARD_IPV4=true
```

The other method uses the `/proc` file system by echoing a 1 into `/proc/sys/net/ipv4/ip_forward` by using the following command:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

If you use the second method, be sure to add this command to your firewall startup script or another script run at boot.



IP forwarding can be enabled two ways. Make sure to know both.

Saving the configuration

The rules you create with `ipchains` are volatile; they go away when the system is rebooted. To keep them around, you should create a script that runs the commands again and have it started at boot. The following is an example script.

```
#!/bin/bash
#
# Firewall Script
#
# Enable IP Forwarding
#
echo 1 > /proc/sys/net/ipv4/ip_forward
#
# Flush old rulesets.
#
ipchains -F input
ipchains -F output
ipchains -F forward
#
```



```
# Set default policies.
#
ipchains -P input DENY
ipchains -P output ACCEPT
ipchains -P forward ACCEPT
#
# Allow local interfaces
#
ipchains -A input -i lo -j ACCEPT
ipchains -A input -i eth0 -j ACCEPT
#
# Deny address blocks we should never see.
# Also include our local address block since we should not
# see these on this interface.
ipchains -A input -i eth1 -s 10.0.0.0/8 -j DENY
ipchains -A input -i eth1 -s 172.16.0.0/12 -j DENY
ipchains -A input -i eth1 -s 192.168.1.0/24 -j DENY
#
# Allow inbound services. Mail and SSH.
#
ipchains -A input -i eth1 -p tcp -d 192.168.1.2 25 -j ACCEPT
ipchains -A input -i eth1 -dport 22 -j ACCEPT
#
# Allow DNS query responses.
#
ipchains -A input -i eth1 -p udp -s 1.2.3.4 53 -j ACCEPT
ipchains -A input -i eth1 -p tcp -s 1.2.3.4 53 -j ACCEPT
#
# Deny all other packets and log them.
#
ipchains -A input -j DENY -l
#
# Enable IP Masq for the internal network.
#
ipchains -A forward -s 192.168.1.0/24 -d ! 192.168.1.0/24 -j
MASQ
```

Controlling authentication with Pluggable Authentication Modules

Pluggable Authentication Modules, or PAM, is a suite of libraries that lets the system administrator tailor how applications authenticate users. This capability lets you change the way users authenticate without having to recompile an application. Modules can be added or removed to change the authentication type. Table 17-14 shows some of the most common modules to give you an idea of how they work.

Table 17-14
PAM Modules

<i>Module</i>	<i>Function</i>
pam_access.so	Restricts which hosts a session can originate from.
pam_cracklib.so	Checks the strength of a password.
pam_deny.so	Checks to see if the account being accessed is expired.
pam_listfile.so	Limits access to a service based on a file.
pam_nologin.so	Checks the /etc/nologin file to see if users are allowed to log in at this time.
pam_securetty.so	Blocks root login from any device not listed in /etc/securetty.
pam_time.so	Restricts the time in which users can access a service.

Since PAM functionality is enabled through called libraries, you can check to see if an application supports PAM with the `ldd` command.

```
[root@redhat /usr/bin]# ldd passwd
libdl.so.2 => /lib/libdl.so.2 (0x40019000)
libpam.so.0 => /lib/libpam.so.0 (0x4001d000)
libpam_misc.so.0 => /lib/libpam_misc.so.0 (0x40025000)
libpwdb.so.0 => /lib/libpwdb.so.0 (0x40028000)
libpopt.so.0 => /usr/lib/libpopt.so.0 (0x40072000)
libc.so.6 => /lib/libc.so.6 (0x40078000)
/lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
libcrypt.so.1 => /lib/libcrypt.so.1 (0x4016d000)
libnsl.so.1 => /lib/libnsl.so.1 (0x4019a000)
```

Notice the two `libpam` libraries. These libraries show that the `passwd` tool supports PAM. The PAM configuration files are now kept in `/etc/pam.d`. Each application has its own configuration file. The old method uses a single `/etc/pam.conf` file, which has the following format:

```
service-name module-type control-flag module-path
arguments
```

The new files stored in `/etc/pam.d` have the following format:

```
module-type control-flag module-path arguments
```

Below is an example of `/etc/pam.d/login`:

```
##PAM-1.0
auth required /lib/security/pam_securetty.so
auth required /lib/security/pam_stack.so service=system-auth
auth required /lib/security/pam_nologin.so
```

account	required	/lib/security/pam_stack.so	service=system-auth
password	required	/lib/security/pam_stack.so	service=system-auth
session	required	/lib/security/pam_stack.so	service=system-auth
session	optional	/lib/security/pam_console.so	

The `service-name` is no longer needed since each service now has its own file. Table 17-15 shows the use of each field. The `module-type` field supports four different types, as shown in Table 17-16. The `control-flag` defines what happens when a module returns a result. The available settings are `optional`, `required`, `requisite`, and `sufficient`. If this field is set to `required` or `requisite`, the module test must return a pass or access is denied.

Table 17-15
PAM File Entries

<i>Field</i>	<i>Function</i>
<i>service-name</i>	The name of the service used for this entry.
<i>module-type</i>	One of the four supported module types: <code>auth</code> , <code>account</code> , <code>session</code> , or <code>password</code> .
<i>control-flag</i>	Defines how the PAM library reacts to a success or failure.
<i>module-path</i>	The path to the pluggable module.
<i>arguments</i>	A list of tokens passed to the module when it is run.

Table 17-16
Module Types

<i>Type</i>	<i>Description</i>
<code>auth</code>	Handles user authentication and makes sure users are who they claim to be.
<code>account</code>	Used to restrict access to a service based on time, maximum user limit, or other criteria set by the module.
<code>session</code>	Handles tasks to be done before or after a user accesses a service, such as logging.
<code>password</code>	Responsible for updating authentication tokens.

Monitoring security lists and sites

New security exploits are discovered all of the time. At the beginning of the chapter, we said that security is a process. When a new exploit is found, you must apply a fix

or update to the service affected. There are a number of good mailing lists and sites on the Internet to monitor for information. Table 17-17 lists several Web sites to monitor.



When a new security advisory is released, be sure to update your software as soon as possible to reduce the time of risk.

Table 17-17
Security Web Sites

<i>Site</i>	<i>Description</i>
http://www.securityfocus.com	Excellent site for new exploits and security news.
http://www.cert.org	CERT site for security advisories. This site is run by a reporting center for security problems at Carnegie Mellon University.
http://www.sans.org	Professional organization that sponsors security-related conferences and training programs.

The most popular mailing list to monitor current security information is BUGTRAQ. To subscribe send a message to listserv@securityfocus.com with the following message body:

```
SUBSCRIBE BUGTRAQ Lastname, Firstname
```

Most Linux distributions have their own security mailing lists and Web sites. Table 17-18 lists the sites for each distribution.

Table 17-18
Distribution Security Lists

<i>Distribution</i>	<i>Site</i>
Mandrake	http://www.linux-mandrake.com/en/flists.php3
Red Hat	http://www.redhat.com/support/errata/index.html
SuSE	http://www.suse.com/us/support/security/index.html
Debian	http://www.debian.org/security/
Turbolinux	http://www.turbolinux.com/security/
Caldera	http://www.caldera.com/support/security/
Corel	http://www.corellinux.com/support/updates.htm

Limiting Users

Objective
1.14 Security

- **Setup user level security.** Set limits on user logins, processes, and memory usage

One type of security violation is known as a *denial-of-service (DoS) attack*. These attacks are caused by malicious users using an excessive amount of resources so that others cannot legitimately use the system. To stop some attacks, you can limit your users to certain constraints. This is done with the `/etc/security/limits.conf` file. The format for this file is as follows:

```
<domain>      <type> <item>          <value>
```

The *domain* can be set to a user name, a group name (*@group*), or an *** for everyone. The *type* field designates the type of limit this is, hard or soft. Table 17-19 lists the items that can be limited. The *value* field sets the limit amount.

Table 17-19
Items

Limit	Description
core	Limits the core file size, in K
data	Limits the maximum data size, in K
fsize	Limits the maximum file size, in K
memlock	Limits the maximum amount of locked-in-memory address space, in K
nofile	Limits the maximum number of open files.
rss	Limits the maximum resident set size, in K
stack	Limits the maximum stack size, in K
cpu	Limits the amount of CPU time given to the user, in minutes.
nproc	Limits the maximum number of processes.
as	Limits the address space.
maxlogins	Limits the maximum number of logins for this user.
priority	Sets the priority to run user processes at.

Take the following example:

```
#<domain>      <type> <item>          <value>
```

```

*          hard   core      10000
*          hard   rss       10000
@hr        hard   nproc    20
@marketing soft   nproc    20
@marketing hard   nproc    50
ftp        hard   nproc    0
@intern    -      maxlogins 4

```

There is no penalty for hitting a soft limit; it should be considered the normal operating level. The example gives everyone a limit of 10MB for core files and 10MB for resident set size. Users in the HR group have a limit of 20 processes while marketing group members can have 50 processes. The FTP limit stops users from logging in since they are not allowed any processes. Finally, users in the interns group are limited to four logins.

Users can also be limited with the `ulimit` command. The syntax for `ulimit` is as follows:

```
ulimit [-SHacdfmstpnuv [limit]]
```

Table 17-20 lists the options for `ulimit`. For example, to limit the size the users core files to 10MB, you use the following:

```
ulimit -c 10000
```

Normally the `ulimit` command is specified in the user's profile to be run at login.



Be sure to know both of the ways to limit users.

Table 17-20
ulimit Options

Option	Function
-a	Display all current limits.
-c < <i>limit</i> >	Limit the maximum core size.
-d < <i>limit</i> >	Limit the maximum size of a process's data segment.
-f < <i>limit</i> >	Limit the maximum size of files created.
-m < <i>limit</i> >	Limit the maximum resident set size.
-s < <i>limit</i> >	Limit the stack size limit.
-t < <i>n seconds</i> >	Limit the amount of CPU time in seconds.
-p < <i>limit</i> >	Limit the pipe size. Pipes are used for application communication.
-n < <i>limit</i> >	Limit the maximum number of open file descriptors.
-u < <i>limit</i> >	Limit the number of processes the user can start.
-v < <i>limit</i> >	Limit the amount of virtual memory available to the shell.

Checking Security

You can use several tools to check the security of your system. Some recommended applications are the following:

- ♦ nmap
- ♦ SAINT
- ♦ Nessus
- ♦ crack
- ♦ COPS
- ♦ Tripwire
- ♦ Bastille Linux

nmap

The `nmap` tool is used to scan a system to see which ports are open and to try and determine some information about the host. This is very useful for checking out a firewall or secured system. For example, the following is a simple scan of a nonsecure host:

```
europa:/etc/security# nmap -sT -O badserver
```

```
Starting nmap V. 2.54BETA7 ( www.insecure.org/nmap/ )
Interesting ports on (1.2.3.4):
(The 1506 ports scanned but not shown below are in state:
closed)
Port      State      Service
7/tcp    open      echo
9/tcp    open      discard
13/tcp   open      daytime
17/tcp   open      qotd
19/tcp   open      chargen
21/tcp   open      ftp
25/tcp   open      smtp
26/tcp   open      unknown
27/tcp   open      nsw-fe
42/tcp   open      nameserver
80/tcp   open      http
110/tcp  open      pop-3
119/tcp  open      nntp
135/tcp  open      loc-srv
139/tcp  open      netbios-ssn
143/tcp  open      imap2
389/tcp  open      ldap
443/tcp  open      https
```

```

515/tcp    open      printer
563/tcp    open      snews
593/tcp    open      http-rpc-epmap
636/tcp    open      ldapssl
993/tcp    open      imaps
995/tcp    open      pop3s
2401/tcp   open      cvspserver
6667/tcp   open      irc
6668/tcp   open      irc
8080/tcp   open      http-proxy

```

```

TCP Sequence Prediction: Class=trivial time dependency
                        Difficulty=73 (Easy)
Remote operating system guess: Windows NT4 / Win95 / Win98

Nmap run completed -- 1 IP address (1 host up) scanned in 3
seconds

```

As you can see, `nmap` shows which ports are open along with the type of operating system used. `nmap` has a large number of options and command-line parameters. If you use X11, a GUI front end makes using `nmap` much easier. It is called `nmapfe` and is usually available from wherever `nmap` is obtained.

SAINT

SAINT is an update of the original SATAN security scanner. SAINT probes networked computers to find out what services they offer. SAINT goes further than `nmap` in that it knows about the services run and can look for common exploits and misconfigurations. It provides a very easy to use Web interface that shows exploit information in an easy to read manner.

SAINT is available from <http://www.wwdsi.com> at no charge.

Nessus

Nessus is another security scanner built in a more modular way than SAINT. It also uses a client-server architecture so you can remotely scan machines from another network. There are also clients written for operating systems other than UNIX.

Nessus is available from <http://www.nessus.org>.

crack

The `crack` tool is used to check for bad passwords. It is a good idea to periodically check your users' passwords with `crack` to see if they are easily decrypted.

`crack` is available from <ftp://ftp.cert.org>.

COPS

COPS, Computer Oracle and Password System, is a set of programs that monitors the security situation of a system. It monitors the following:

- ♦ File, directory, and device permissions/modes.
- ♦ Poor passwords.
- ♦ Content, format, and security of password and group files.
- ♦ The programs and files run in the `rc` directories and `crontab` files.
- ♦ Existence of SUID root files, their writability, and whether or not they are shell scripts.
- ♦ A CRC check against important binaries or key files to report any changes therein.
- ♦ Writability of users home directories and startup files (`.profile`, `.cshrc`, and so on).
- ♦ Anonymous FTP setup.
- ♦ Unrestricted `tftp`, decode alias in `sendmail`, SUID `uudecode` problems, hidden shells inside `inetd.conf`, `rex` running in `inetd.conf`.
- ♦ Miscellaneous root checks — current directory in the search path, a “+” in `/etc/host.equiv`, unrestricted NFS mounts, root in `/etc/ftpusers` and so on.
- ♦ The dates of CERT advisories versus key files. This checks the dates that various bugs and security holes were reported by CERT against the actual date on the file in question. A positive result doesn’t always mean that a bug was found, but it is a good indication that you should look at the advisory and file for further clues. A negative result, obviously, does not mean that your software has no holes, merely that it has been modified in some way (perhaps merely touched) since the advisory was sent out.
- ♦ The Kuang expert system. This takes a set of rules and tries to determine if your system can be compromised.

Generally COPS is run via `cron` every night to perform its checks. This way you are alerted right away of any changes that could be a problem.

COPS is available from <http://www.cerias.purdue.edu>.

Tripwire

Tripwire is a tool that monitors the permissions and checksums of system files so that you can easily detect if they have been tampered with. When set up, Tripwire creates a database with the checksum information of all the configured files. It is best to keep this database and Tripwire’s configuration files on a secure system on read-only media.

Tripwire started off as a free product that was later turned commercial. However, a free version is still available and maintained. It is available at <http://www.tripwire.org>.

Bastille Linux

Bastille Linux is a hardening script for Red Hat. It runs through a series of questions, and based on the information given the administrator has the choice of taking the scripts advice or not. Bastille provides no services itself; it only locks down the security of an existing system.

If you are new to security it is a good idea to go through the Bastille script not only to secure your system, but also teach you about security. The script provides excellent information on what it does, and why it does the things it does.

Bastille Linux is available from <http://www.bastille-linux.org>.

Key Point Summary

Security is a very important piece of the system administrator's job. If done properly it can take up a large portion of your time, but it is time well spent. The information provided in this chapter is a very good starting point on the road of knowledge.

- ♦ TCP wrappers uses the `hosts.allow` and `hosts.deny` files to provide security for TCP services.
- ♦ `tcpdchk` is used to check the syntax of your `hosts.allow` and `hosts.deny` files.
- ♦ `tcpdmatch` is used to provide theoretical connections to your system to test your `hosts.allow` and `hosts.deny`.
- ♦ SUID executables and scripts can be very large security risks.
- ♦ The validity of RPM packages can be checked with PGP and GPG.
- ♦ Any package or file can be checked with the `md5sum` tool.
- ♦ The SGID bit can be used to create directories that let users share files.
- ♦ If a directory has the SGID bit set, new files belong to the group of the directory and not the user.
- ♦ The Berkeley `r` commands are very insecure.
- ♦ The `.rhosts` file is used to give remote users access to a local account.
- ♦ The `hosts.equiv` file is used to allow or deny access to the `r` commands from other hosts.
- ♦ `rlogin` is used to remotely log in to another system.

- ♦ `rcp` is used to copy files between systems.
- ♦ `rsh` is used to execute a single command on a remote system.
- ♦ `rwho` shows who is logged in on the local network.
- ♦ `ruptime` shows the uptime of systems on the network.
- ♦ SSH is a secure replacement for the Berkeley `r` commands.
- ♦ `sshd` is the SSH server daemon.
- ♦ `ssh` is the SSH client used to log in to a remote system.
- ♦ `scp` is used to copy files between systems.
- ♦ `sshd_config` is the SSH server configuration file.
- ♦ `ssh_config` is the system-wide SSH client configuration file.
- ♦ `ssh-keygen` is used to create a user's identity keys.
- ♦ `ssh-agent` is used to hold keys so that a user needs to enter a passphrase only once.
- ♦ `ssh-add` adds a user's private keys to the agent.
- ♦ Shadow passwords store the encrypted password in `/etc/shadow`.
- ♦ Normal passwords are converted to shadow passwords using `pwconv` and `grpconv`.
- ♦ Shadow passwords can be converted back to normal passwords using `pwunconv` and `grpunconv`.
- ♦ Unneeded services should be removed to reduce security risks.
- ♦ The `ipchains` tool is used to set up firewall rules.
- ♦ The three default chains are Input, Output, and Forward.
- ♦ Rules are made up of conditions and targets.
- ♦ The first rule a packet matches is the only one applied.
- ♦ The six standard targets are ACCEPT, DENY, REJECT, MASQ, REDIRECT, and RETURN.
- ♦ IP masquerading lets you share one real IP address with many systems.
- ♦ PAM lets an administrator customize the authentication of some applications.
- ♦ The PAM configuration files are stored in `/etc/pam.d`.
- ♦ Security lists and sites should be monitored for new exploits.
- ♦ The `/etc/security/limits.conf` file can limit certain user items.
- ♦ The `ulimit` command can also limit some user items such as number of processes.
- ♦ Security tools should be used to detect problems and watch for file changes.



STUDY GUIDE

The following questions and exercises will allow you to review the information covered in this chapter. Take your time when completing this section of the chapter, carefully reviewing any questions that you may have. Answering the question correctly is not as important as understanding the answer, so review any material that you might still be unsure of. Being comfortable with the questions and answers presented here will help you be more prepared for the certification exam questions.

Assessment Questions

1. Which would allow users from `somedomain.org` to access your FTP server?
 - A. `hosts.allow: in.ftpd : ANY`
 - B. `hosts.deny: ftpd : ALL EXCEPT somedomain.org`
 - C. `hosts.allow: in.ftpd : somedomain.org`
 - D. `hosts.deny: in.ftpd : ALL`
2. Which service cannot be used with TCP wrappers?
 - A. SSH
 - B. Telnet
 - C. tftp
 - D. HTTP
3. Which tool is used to check the logic of your TCP wrapper rules?
 - A. `tcpdmatch`
 - B. `tcpwmatch`
 - C. `tcpdtest`
 - D. `tcpwquery`
4. Which tool can be used to see if a package has been tampered with?
 - A. `verify`
 - B. `md5`
 - C. `integ`
 - D. `md5sum`

5. Which commands would you run to create a shared directory named Share?
- A. `mkdir Share`
 - B. `chmod 770 Share`
 - C. `chmod u+s Share`
 - D. `chmod g+s Share`
6. Which entry in `.rhosts` allows user Angie from the host named deedee?
- A. `deedee +Angie`
 - B. `Angie deedee`
 - C. `+Angie deedee`
 - D. `deedee Angie`
7. Which of the following creates a large security risk in `hosts.equiv`?
- A. `+`
 - B. `!`
 - C. `ALL`
 - D. `ANY`
8. Which command copies the file `/doc/MyDoc` from the system `webserver` to a local file named `MyNewDoc`?
- A. `rcp MyNewDoc webserver:/doc/MyDoc`
 - B. `rpc webserver:/doc/MyDoc MyNewDoc`
 - C. `rcp webserver:/doc/MyDoc MyNewDoc`
 - D. `rpc webserver:/doc/MyDoc MyNewDoc`
9. Which port is used by SSH?
- A. 22
 - B. 23
 - C. 19
 - D. 53
10. Which entry in `sshd_config` restricts root from a remote system?
- A. `LoginRootPermit`
 - B. `PermitRootLogin`
 - C. `NoRootLogin`
 - D. `DenyRootLogin`

- 11. Which entry in `sshd_config` allows the use of `.rhosts`?**
- A. `AllowRhosts`
 - B. `RhostsAllow`
 - C. `RSAAuthentication`
 - D. `RhostsAuthentication`
- 12. Which option causes `ssh` to use a different user name?**
- A. `-l`
 - B. `-L`
 - C. `-u`
 - D. `-U`
- 13. Which command lists all keys being represented by `ssh-agent`?**
- A. `ssh-agent -l`
 - B. `ssh-add -l`
 - C. `ssh-agent -L`
 - D. `ssh-add -L`
- 14. Which tool converts the `group` file to use shadow passwords?**
- A. `grpconv`
 - B. `groupconv`
 - C. `groupcv`
 - D. `grpconvert`
- 15. Which command clears the Input chain of rules?**
- A. `ipchains -C input`
 - B. `ipchains -F input`
 - C. `ipchains -F ALL`
 - D. `ipchains -P input`
- 16. Which command shows all of the current rules?**
- A. `ipchains -L`
 - B. `ipchains -l`
 - C. `ipchains -S`
 - D. `ipchains -s`

17. Which type of PAM module queries the user for a password?
- A. password
 - B. session
 - C. account
 - D. auth
18. Which command limits the amount of CPU time a user can use to 60 minutes?
- A. `ulimit -C 3600`
 - B. `ulimit -t 3600`
 - C. `ulimit -C 1h`
 - D. `ulimit -t 1h`
19. Which tool monitors for file changes?
- A. Nessus
 - B. crack
 - C. Tripwire
 - D. nmap
20. SSH provides security against bad passwords.
- A. True
 - B. False

Scenarios

1. Your users need to access servers at a remote site over the Internet. They sometimes need to log into several servers many times throughout the day. What is the best way to set this up for security and convenience?
2. You have some users that run processes that sometimes run away and create thousands of child processes, which cause severe performance issues for the rest of the company. What can you do to stop this from happening?

Lab Exercises

Lab 17-1 Creating a shared directory

This lab creates a shared directory to show the use of SGID.

1. Log into a Linux system as root.

2. Create a directory named “Share” off of the root. For example:

```
[root@rh7 /]# mkdir Share
```

3. Create a group named “Marketing”.

```
[root@rh7 /]# groupadd Marketing
```

4. Change the permissions on Share to 2770. This sets the SGID bit, along with read/write/execute for the owner and group.

```
[root@rh7 /]# chmod 2770 Share
```

5. Change the group for Share to Marketing.

```
[root@rh7 /]# chown root.Marketing Share
[root@rh7 /]# ls -ld Share
drwxrws--- 2 root Marketing 4096 Mar 15 17:51 Share
```

6. Add a normal user to the new Marketing group by editing the `/etc/group` file.

```
Marketing:x:500:jason
```

7. Log into the system as a normal user; make sure the user’s `umask` is 002.

```
[jason@rh7 jason]$ umask
002
```

8. Change directory to Share and create a new file using the `touch` command.

```
[jason@rh7 jason]$ cd /Share
[jason@rh7 /Share]$ touch MyFile
[jason@rh7 /Share]$
```

9. Now type `ls -l` and look at the group the file belongs to. It should be Marketing instead of the user’s group.

```
[jason@rh7 /Share]$ ls -l
total 0
-rw-rw-r-- 1 jason Marketing 0 Mar 15 18:03 MyFile
```

Lab 17-2 Using IP chains

This assumes you have two Linux systems, we’ll call them Host and Client. This lab goes through some IP chains configuration. The Host computer must have the Telnet daemon enabled in `/etc/inetd.conf`.

1. Log into Host as root.

2. Clear all of the chains using the following:

```
ipchains -F input
ipchains -F output
ipchains -F forward
```


3. Set the default policy for all chains using the following:

```
ipchains -P input ACCEPT
ipchains -P output ACCEPT
ipchains -P forward ACCEPT
```

4. Block the Telnet port using the following:

```
ipchains -A input -dport 23 -j DENY
```

5. Log into the Client as any user.

6. Try to Telnet to Host. Notice the connection must timeout.

7. Log into Host as root.

8. Clear the Input chain and set the default policy using the following:

```
ipchains -F input
ipchains -P input ACCEPT
```

9. Block the Telnet port again using the following:

```
ipchains -A input -dport 23 -j REJECT
```

10. Log into Client as any user.

11. Try to Telnet to Host. Notice the connection is now refused.

Answers to Chapter Questions

Chapter Pre-Test

1. SSH
2. `hosts.allow` and `hosts.deny`
3. Use either the `ulimit` command or `/etc/security/limits.conf`.
4. Create a directory, set the correct permissions, and use the SGID bit.
5. `sshd_config`
6. `rsh`
7. Any service that uses UDP
8. `/etc/pam.d`
9. IP chains
10. IP masquerading

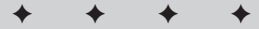
Assessment Questions

1. **C.** Option A is incorrect because to allow all hosts you must use the `ALL` wildcard, not `ANY`. Choice B specifies the wrong daemon name. Choice D denies all hosts from connecting to the FTP server. For more information see the “Configuring TCP wrappers” section.
2. **C.** `tftp` uses UDP, and TCP wrappers can be used only with TCP applications. For more information see the “Configuring TCP wrappers” section.
3. **A.** The other options are invalid. For more information see the “Configuring TCP wrappers” section.
4. **D.** The other commands are invalid. For more information see the “Managing packages” section.
5. **A, B, and D.** You must create the directory, set the proper permissions, and set the SGID bit. For more information see the “Configuring TCP wrappers” section and the “Using setgid” section.
6. **D.** The format is `HOST USER`. An optional `+` sign can be used to enable a host. For more information see the “Configuring access to the `r` commands” section.
7. **A.** By just putting the `+` sign, you allow any host to connect. For more information see the “Configuring access to the `r` commands” section.
8. **C.** The syntax for Choice A is incorrect. The other options are invalid. For more information see the “Using `rcp`” section.
9. **A.** Telnet uses port 23, and DNS uses port 53. For more information see the “Using SSH” section.
10. **B.** Unless this option is enabled, the root user will never be allowed to log in. The other options are invalid. For more information see the “Configuring the SSH server” section.
11. **D.** This option can be used to support older style `rhosts` files that may already be on the system. For more information see the “Configuring the SSH server” section.
12. **A.** The `-L` option is used to specify a listening port. The other options are invalid. For more information see the “Using `ssh`” section.
13. **B.** The `-L` option lists all public keys. For more information see the “User key management” section.
14. **A.** The other options are invalid. For more information see the “Using shadow passwords” section.
15. **B.** The `-C` option is used to check a packet against the selected chain. The `ALL` chain is invalid. The `-P` option is used to set the policy for the specified chain. For more information see the “Blocking unwanted connections with IP chains” section.

16. **A.** The `-l` option is used to enable kernel logging. The `-S` option is used to change the timeout for masquerading. The `-s` option specifies the source. For more information see the “Blocking unwanted connections with IP chains” section.
17. **D.** The `password` module is used to change authentication tokens. For more information see the “Controlling authentication with Pluggable Authentication Modules” section.
18. **B.** The time must be specified in seconds. The `-C` option is invalid. For more information see the “Limiting Users” section.
19. **C.** Tripwire performs several checks to make sure files have not been tampered with. For more information see the “Checking Security” section.
20. **B.** SSH encrypts the connection between systems. It does not stop someone from guessing a badly chosen password. For more information see the “Using SSH” section.

Scenarios

1. The best solution is to install SSH on the servers and clients. This will secure the connections so no one can sniff passwords or any other data. To ease the burden on the users you can set them up to use `ssh-agent` as soon as they log in. This way they need to enter their passphrase only once and can log into the remote systems any time without being prompted.
2. You can use either `ulimit` or `/etc/security/limits.conf`. The best solution is to add all of the problem users to a common group and add them to `limits.conf` with a process limit that should be reasonable. If `ulimit` is used, it can be put in the users’ profile scripts that get run when they log in.



What's on the CD-ROM

This appendix provides you with information on the contents of the CD-ROM that accompanies this book.

System Requirements

Make sure that your computer meets the minimum system requirements listed in this section. If your computer doesn't match up to most of these requirements, you may have a problem using the contents of the CD.

For Microsoft Windows 9x, NT 4.0, Me, or 2000:

- ◆ PC with a Pentium processor running at 90 MHz or faster
- ◆ At least 32MB of RAM installed on your computer; for best performance we recommend at least 64MB
- ◆ A CD-ROM drive
- ◆ Ethernet network interface card (NIC) or modem with a speed of at least 28,800 bps

For Linux:

- ◆ PC with a Pentium processor running at 90 MHz or faster
- ◆ At least 32MB of RAM installed on your computer; for best performance we recommend at least 64MB
- ◆ A CD-ROM drive
- ◆ Ethernet network interface card (NIC) or modem with a speed of at least 28,800 bps

You will need at least 300MB of hard drive space to install all the software from this CD.

Using the CD with Microsoft Windows

To install the items from the CD to your hard drive, follow these steps:

1. Insert the CD into your computer's CD-ROM drive.
2. A window will appear with the following options:
 - **Install**— This will give you the option to install the supplied software and/or the author-created samples from the CD-ROM.
 - **Explore**— Allows you to view the contents of the CD-ROM in its directory structure.
 - **Exit**— Closes the autorun window.

If you do not have autorun enabled or if the autorun window does not appear, follow the steps below to access the CD.

1. Click Start ⇨ Run.
2. In the dialog box that appears, type **d:\setup.exe**, where *d* is the letter of your CD-ROM drive; this will bring up the autorun window described above.
3. Choose the Install, Explore, or Exit option from the menu.

Using the CD with Linux

To install the items from the CD to your hard drive, follow these steps:

1. Log in as root.
2. Insert the CD into your computer's CD-ROM drive.
3. Mount the CD-ROM.
4. Launch a graphical file manager.

What's on the CD

Hungry Minds test engine

For Windows only: The test engine contained on the CD-ROM is powered by Boson Software and is loaded with questions that can help you test your facility with the information you will be responsible to know on the LPIC exams.

When installed and run, the test engine presents you with a multiple-choice, question-and-answer format. Each question deals directly with exam-related material. Once you select what you believe to be the correct answer for each question, the

test engine not only notes whether you are correct or not, but also provides information as to why the right answer is right and the wrong answers are wrong, providing you with valuable information for further review. Thus, the test engine gives not only valuable simulated exam experience, but useful tutorial direction as well.

Note: Since the Hungry Minds test engine will not work on the Linux platform, all of the test engine questions are also included in a PDF file. This file is located at the root of the CD.

Electronic version of LPIC 1 Certification Bible

The complete (and searchable) text of this book is on the CD-ROM in Adobe's Portable Document Format (PDF), readable with the Adobe Acrobat Reader (also included). For more information on Adobe Acrobat Reader, go to www.adobe.com.

Guides

A couple documentation guides that cover a variety of Linux-based tasks are included on the CD-ROM. The following sections briefly discuss both of these guides.

- ♦ **Rute User's Tutorial and Exposition** — The Rute guide is written to be read from beginning to end. It is designed to provide a beginner's guide to Linux- and UNIX-based systems. This guide contains a variety of information on a wide range of system tasks from installation to advanced configuration.
- ♦ **Linux + Windows HOWTO** — This guide provides information on combining Windows and Linux features on a single host. It is written to provide information to a variety of users who may have different needs. Information is included on the various Windows releases and how they can coexist with Linux on a single host.

FAQs

The CD-ROM also contains some useful Frequently Asked Questions (FAQs) documents, which ask and answer some of the questions a Linux user may have. The FAQs contained on this CD are the following:

- ♦ **Linux Frequently Asked Questions with answers** — This document provides general information about the questions most often asked about Linux. This includes questions ranging from "What is Linux?" to "How can I make X Window work?" All Linux users should read this document at least once. It is especially useful for those who are new to Linux.
- ♦ **Linux-RAID FAQ** — This FAQ is written as a guide to the Linux-RAID HOWTO. It answers many of the questions that might arise when trying to configure RAID devices on a Linux system. Information concerning RAID, disk failure, and where to locate more information is contained within this document.

HOWTOs

The HOWTO guides provide step-by-step information on many of the tasks involved on a Linux system. These tasks can range from hardware configuration to software installation and configuration. HOWTOs are an excellent source of information when trying to accomplish a specific task on a Linux system.

- ◆ **Advanced Bash-Scripting HOWTO**— This HOWTO provides information on creating advanced scripts using the bash shell.
- ◆ **Linux 2.4 Advanced Routing HOWTO**— This HOWTO provides information on configuring advanced routing options using Linux.
- ◆ **Apache Overview How-to**— This HOWTO provides an overview of the Apache server, its configuration, and function.
- ◆ **Bash Prompt HOWTO**— This HOWTO provides information for configuration of the prompt using the bash shell.
- ◆ **The Linux Bootdisk HOWTO**— This HOWTO provides information on creating a Linux boot disk that can be used to boot a computer to the Linux operating system.
- ◆ **The Linux BootPrompt HowTo**— This HOWTO provides information about the boot time arguments that can be passed to a Linux system.
- ◆ **Chroot-BIND HOWTO**— This HOWTO provides information on configuration of BIND and DNS using chroot.
- ◆ **Configuration HOWTO**— This HOWTO describes what should be done to tune a Linux system for optimal performance.
- ◆ **The Linux Distribution HOWTO**— This HOWTO provides information on the various Linux distributions so that new users can determine which distribution they wish to run.
- ◆ **DNS HOWTO**— This HOWTO provides information on configuring DNS on a Linux system.
- ◆ **From DOS/Windows to Linux HOWTO**— This HOWTO provides information for DOS/Windows users who are switching to the Linux operating system.
- ◆ **Emacs Beginner's HOWTO**— This HOWTO provides information on using emacs, geared towards the new user.
- ◆ **Linux Ethernet-Howto**— This HOWTO provides information on configuring Linux to function on an Ethernet network.
- ◆ **Filesystems HOWTO**— This HOWTO provides information on the creation and maintenance of file systems that are used on a Linux system.
- ◆ **Firewall and Proxy Server HOWTO**— This HOWTO provides information on the configuration of firewall and proxy servers.
- ◆ **From Power Up To Bash Prompt**— This document provides information about what happens on the system as it goes from the initial hardware power up to the bash prompt when a user logs into the system.

- ♦ **The Linux GCC HOWTO** — This document provides information for the GNU C compiler that runs under Linux.
- ♦ **Linux Hardware Compatibility HOWTO** — This HOWTO contains information about hardware that is supported under the Linux operating system.
- ♦ **The Linux Installation HOWTO** — This HOWTO provides information about the installation processes that are used when installing software on Linux.
- ♦ **Linux IP Masquerade HOWTO** — This HOWTO provides information about configuring IP Masquerading.
- ♦ **ISP-Hookup-HOWTO** — This HOWTO provides information about connecting to an Internet Service Provider using a modem.
- ♦ **“Pocket” ISP based on RedHat Linux** — This HOWTO provides information about providing Internet services using Red Hat.
- ♦ **The Linux Kernel HOWTO** — This HOWTO provides information about upgrading, installing, compiling, and configuring the kernel.
- ♦ **Linux Laptop-HOWTO** — This HOWTO provides information about configuring Linux on a laptop computer.
- ♦ **Linmodem-Mini-HOWTO** — This HOWTO provides information about configuring a Linmodem under Linux.
- ♦ **LILO, Linux Crash Rescue HOW-TO** — This HOWTO provides information about recovering from Linux system failures.
- ♦ **The Linux Electronic Mail Administrator HOWTO** — This HOWTO provides information about configuring an e-mail server on a Linux system.
- ♦ **The Linux Mail User HOWTO** — This HOWTO provides information about configuring an e-mail client on Linux systems.
- ♦ **Modem-HOWTO** — This HOWTO provides information about configuring modems for use on Linux systems.
- ♦ **Multi Disk System Tuning** — This HOWTO provides information about performance tuning a system with multiple disks.
- ♦ **The Linux Networking Overview HOWTO** — This HOWTO provides information about configuring Linux servers on a network.
- ♦ **NFS HOWTO** — This HOWTO provides information about configuring NFS servers and clients.
- ♦ **The Linux NIS(YP)/NIS/NIS+ HOWTO** — This HOWTO provides information about configuring Linux with NIS.
- ♦ **Online Troubleshooting Resources HOWTO** — This HOWTO provides information about finding more information using online resources.
- ♦ **Linux PCMCIA HOWTO** — This HOWTO provides information about configuring PCMCIA support on Linux.
- ♦ **Plug-and-Play-HOWTO** — This HOWTO provides information about configuring Plug and Play on Linux.
- ♦ **Linux PPP HowTo** — This HOWTO provides information about configuring PPP on Linux.

- ◆ **The Linux Printing HOWTO** — This HOWTO provides information about configuring Linux printing.
- ◆ **Linux Security HOWTO** — This HOWTO provides information about providing security to a Linux system.
- ◆ **SMB HOWTO** — This HOWTO provides information about configuring SMB.
- ◆ **The Linux Sound HOWTO** — This HOWTO provides information about configuring sound devices on a Linux system.
- ◆ **User Authentication HOWTO** — This HOWTO provides information about the process of user authentication on a Linux system.
- ◆ **Windows LAN server HOW-TO** — This HOWTO provides information about configuring Linux to provide server services on a network with Microsoft clients.
- ◆ **The Winmodems-and-Linux HOWTO** — This HOWTO provides information about configuring Winmodems on a Linux system.
- ◆ **Linux WWW HOWTO** — This HOWTO provides information about configuring Linux as a Web server and client.
- ◆ **Linux XDMCP HOWTO** — This HOWTO provides information about configuring the X Display Manager Control Protocol.
- ◆ **The Linux XFree86 HOWTO** — This HOWTO provides information about configuring X Window.
- ◆ **XFree86 Video Timings HOWTO** — This HOWTO provides information about configuring video settings for X Window.
- ◆ **The X Window User HOWTO** — This HOWTO provides information about configuring X Window for users.

Troubleshooting

If you have difficulty installing or using the CD-ROM programs, try the following solutions:

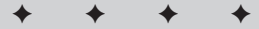
- ◆ **Turn off any antivirus software that you may have running.** Installers sometimes mimic virus activity and can make your computer incorrectly believe that a virus is infecting it. (Be sure to turn the antivirus software back on later.)
- ◆ **Close all running programs.** The more programs you're running, the less memory is available to other programs. Installers also typically update files and programs; if you keep other programs running, installation may not work properly.

If you still have trouble with the CD, please call the Hungry Minds Worldwide Customer Service phone number: (800) 762-2974. Outside the United States, call (317) 572-3993. Hungry Minds, Inc., will provide technical support only for installation and other general quality control items; for technical support on the applications themselves, consult the program's vendor or author.



Practice Exams

The practice exams in this appendix will test your knowledge using exams similar to the real ones. Use these exams after you finish your studies to see where your weak areas are.



Exam 101

1. Which of the following would cause `cmd2` to be run independently of `cmd1`?
 - A. `cmd1 ; cmd2`
 - B. `cmd1 | cmd2`
 - C. `cmd1 >> cmd2`
 - D. `cmd1 > cmd2`

2. _____ is the default signal number used with the `kill` command.

3. The `updatedb` command is used to update the information searched by which command?
 - A. `grep`
 - B. `find`
 - C. `locate`
 - D. `whois`

4. Which command is used to change the priority of a command as it is run?
 - A. `job`
 - B. `nice`
 - C. `renice`
 - D. `top`

5. Which command would be used to produce the following permissions?
-rw-rw-r-- myfile
- A. chown myfile
 - B. chmod 661 myfile
 - C. chmod 331 myfile
 - D. chmod 664 myfile
6. Which file is used to specify a user's home directory?
- A. /etc/passwd
 - B. /etc/skel
 - C. /etc/profile
 - D. /etc/bashrc
7. _____ is used to view jobs scheduled to run at regular intervals.
8. Which command will search the file `editorial` for all occurrences of the word *lady* and replace them with *Lady*?
- A. `sed -a s/lady/Lady editorial`
 - B. `tr l,L editorial`
 - C. `sed 's/lady/Lady/' editorial`
 - D. `grep 'lady' editorial | tr Lady`
9. Which command allows you to view system resource usage in real time?
- A. nice
 - B. renice
 - C. `ps -aux`
 - D. top
10. Which file contains a list of currently mounted file systems?
- A. /etc/proc/
 - B. /mnt
 - C. /etc/fstab
 - D. /etc/mstab
11. Which allows an environment variable declaration to be used system-wide?
- A. echo
 - B. export
 - C. cat
 - D. env

12. Which command would move the file and directory contents of the /ME directory to the /YOU directory?
- A. `mv -a /ME /YOU`
 - B. `cp /ME/* /YOU/* ; rm -r /ME/*`
 - C. `mv /ME/* /YOU/*`
 - D. `cp -r /ME/* /YOU ; rm -r /ME`
13. What command would produce a file named `xmaslist` using the contents of the `list` file, arranging the contents alphabetically and numbering each line?
- A. `fmt list > xmaslist`
 - B. `sort list | nl > xmaslist`
 - C. `ln list | sort | xmaslist`
 - D. `sort list | ln | xmaslist`
14. Which of the following commands would kill the process 1234 without giving it a chance to gracefully exit?
- A. `kill %1234`
 - B. `kill -now 1234`
 - C. `kill -9 1234`
 - D. `kill -12 1234`
15. Which of the following tools are used when creating a new partition for data storage on a Linux system?
- A. `fdisk`
 - B. `format`
 - C. `mkfs`
 - D. `mount`
16. The _____ utility provides a summary of disk space used in the `pwd`.
17. Which command is used to edit user quotas?
- A. `quota`
 - B. `quotaon`
 - C. `edquota`
 - D. `vi`

18. The `umask` of 002 would produce what default file permissions?
- A. `-rwxrwxrw-`
 - B. `-rw-rw-r--`
 - C. `-----x`
 - D. `-rwxrwxr-x`
19. What access rights are required for the `/etc/shadow` file?
- A. `-r-----`
 - B. `-rw-rw-r--`
 - C. `-r--r-----`
 - D. `-rwxr--r--`
20. Which command will immediately shut down a computer running Linux?
- A. `shutdown`
 - B. `telinit 6`
 - C. `shutdown now`
 - D. `telinit 0`
21. A new user (Debbie) needs access to the company's Linux system, what must be done so that she can access the system with the `userid` `debbie`? (Select all that apply.)
- A. `useradd debbie`
 - B. `group debbie`
 - C. `shadow debbie`
 - D. `passwd debbie`
22. Which command would give the owner and group of `myfile` the permissions of read and execute while all others have read access?
- A. `chmod 003 myfile`
 - B. `chmod 554 myfile`
 - C. `chmod 331 myfile`
 - D. `chmod 662 myfile`
23. Which command would be used to view a description of the `tar` utility including the options available for use?
- A. `more tar`
 - B. `grep tar`
 - C. `man tar`
 - D. `help tar`

24. _____ is the location of files that are copied to each user's home directory.
25. The `PS1` environment variable is used to define which of the following?
- A. command path
 - B. home directory
 - C. shell prompt
 - D. signature file
26. Which of the following would change the user's `pwd` to their home directory regardless of their current `pwd`? (Select all that apply.)
- A. `cd`
 - B. `cd ..`
 - C. `cd~`
 - D. `cd $HOME`
27. Which of the following directories is used to store system utilities used by the superuser?
- A. `/etc`
 - B. `/root`
 - C. `/sbin`
 - D. `/usr/local`
28. Where are the user shells assigned?
- A. `/etc/skel`
 - B. `/etc/passwd`
 - C. `/etc/profile`
 - D. LILO
29. You have recently installed a new disk drive in your Linux system. You wish to store a directory containing many data files used within your company on the new disk drive. Which of the following can be used so the change is invisible to users of the system?
- A. `cp`
 - B. `mv`
 - C. hard links
 - D. soft links

- 30.** Which command and options are used to delete a user's account and home directory?
- A. `userdel -r`
 - B. `usermod -f`
 - C. `passwd -e`
 - D. `usermod -e`
- 31.** The _____ utility is used to display files in hexadecimal format.
- 32.** What command can be used to locate specific text within a file?
- A. `grep`
 - B. `locate`
 - C. `find`
 - D. `where`
- 33.** Which of the following can be used to suspend a process running in the foreground?
- A. `&`
 - B. `fg`
 - C. `bg`
 - D. `Ctrl+Z`
- 34.** Where might you find instructions on installing and configuring a Winmodem on a Linux system?
- A. man pages
 - B. HOWTOs
 - C. `/usr/doc`
 - D. `dmesg`
- 35.** Which of the following would be used to run a script called `mozilla_update` as a background process?
- A. `bg mozilla_update`
 - B. `mozilla_update&`
 - C. ``mozilla_process``
 - D. `mozilla_process -bg`

36. Which environment variable would be used to specify the location of files that you wish to run without specifying an absolute path?
- A. HOME
 - B. PS1
 - C. PATH
 - D. TERM
37. The _____ command is used to change ownership of a file.
38. When booting your system you notice that your sound module failed to load; however, the error message scrolled by too quickly. What command can you use to view the errors?
- A. dmesg
 - B. messages
 - C. errors
 - D. lilo
39. What command is used to create a new archived file using compression?
- A. `gzip -t`
 - B. `tar -cfz`
 - C. `compress -tf`
 - D. `bzip2 -tbz`
40. Which command is used to view processes running in the background?
- A. dmesg
 - B. `ls`
 - C. `bg`
 - D. `jobs`
41. Which utility is used to view the beginning of a file?
- A. `tac`
 - B. `cat`
 - C. `head`
 - D. `top`
42. Which type of quota limit prevents a user from creating new files?
- A. stop limit
 - B. soft limit
 - C. hard limit
 - D. max limit

43. Which runlevel is used for administrative tasks performed locally, disallowing network connections to the system?
- A. 0
 - B. 1
 - C. 3
 - D. 6
44. User quotas are maintained in which file?
- A. /etc/fstab
 - B. /etc/passwd
 - C. /quota.user
 - D. /lilo
45. You are told by a co-worker that information pertaining to the `syslog` command can be found in man page 3. How would you view this information?
- A. `man syslog 3`
 - B. `man syslog -3`
 - C. `man -3 syslog`
 - D. `man 3 syslog`
46. Which command is not used to view the contents of a file?
- A. `cat`
 - B. `more`
 - C. `less`
 - D. `expand`
47. The _____ command is used to unmount a file system.
48. The _____ command will display previously entered commands.
49. What type of information might you find using `www.deja.com`?
- A. man pages
 - B. Usenet postings
 - C. software patches
 - D. kernel updates
50. Which directory utilizes no disk space?
- A. /proc
 - B. /boot
 - C. /root
 - D. /local

51. The _____ command is used to perform a consistency check on file systems.
52. Which command would create a file called `mydirlisting` that includes a detailed listing of all the files and directories located within your home directory?
- A. `ls -aR $HOME > mydirlisting`
 - B. `ls -d > mydirlisting`
 - C. `mydirlisting < ls -al $HOME`
 - D. `ls -al | mydirlisting`

Exam 102

1. The _____ command is used to create an ext2 file system.
2. You're repairing the master boot record on a system. You have booted the system via a boot disk and currently have the root volume mounted to `/recover`. How would you reinstall LILO?
 - A. `lilo -r /recover/boot`
 - B. `lilo -f /recover`
 - C. `lilo -r /recover`
 - D. `lilo -r /recover -b /recover/boot`
3. The _____ file needs to be updated when a new library is added.
4. Which command removes a package named `editor-2.0.i386.rpm`?
 - A. `rpm -i editor-2.0.i386.rpm`
 - B. `rpm -e editor-2.0.i386.rpm`
 - C. `dpkg -install editor-2.0.i386.rpm`
 - D. `apt-get install editor-2.0.i386.rpm`
5. To make a single change to the kernel configuration before compiling, you would edit which file?
 - A. `/usr/src/linux/Makefile`
 - B. `/usr/src/linux/.config`
 - C. `/usr/src/linux/config`
 - D. `/usr/src/linux/configure`

6. Which command will tell you which package owns the file `/etc/printcap`?
- A. `dpkg -F /etc/printcap`
 - B. `rpm -f /etc/printcap`
 - C. `who /etc/printcap`
 - D. `rpm -qf /etc/printcap`
7. The _____ file is executed for all nonlogin bash sessions.
8. Which command shows the route between your host and the remote host `www.somedomain.org`?
- A. `traceroute -n www.somedomain.org`
 - B. `route www.somedomain.org`
 - C. `ping -r www.somedomain.org`
 - D. `route -n www.somedomain.org`
9. Which command is the best to use to find a contact name for another domain?
- A. `nslookup`
 - B. `whoinfo`
 - C. `whois`
 - D. `dig`
10. Which command(s) must be run after the `inetd.conf` file is updated?
- A. `/etc/rc.d/init.d/inetd restart`
 - B. `kill -1 inetd`
 - C. `kill -1 `cat /var/run/inetd.pid``
 - D. No command needed.
11. The _____ file can be created to keep nonroot users from logging in to the system.
12. Which command(s) is/are used to enable shadow passwords?
- A. `shadow`
 - B. `pw-conv`
 - C. `pwconv`
 - D. `grpconv`
13. Which command(s) is/are used to decompress the file `mydoc.gz`?
- A. `gunzip mydoc.gz`
 - B. `tar -zxv mydoc.gz`
 - C. `gzip -d mydoc.gz`
 - D. `gunzip -d mydoc.gz`

14. To activate all swap partitions on the system you would use the _____ command (with options).
15. Which command displays the output of the `utmp` file?
- A. `what`
 - B. `who`
 - C. `last`
 - D. `when`
16. The _____ tool is used to control the Apache `httpd` service.
- A. `apached`
 - B. `httpdctl`
 - C. `htmgr`
 - D. `apachectl`
17. Which command displays queue information for all local printers?
- A. `lpq -a`
 - B. `lpc -a`
 - C. `printq`
 - D. `lpq -v`
18. Which disk partitioning tool provides a GUI?
- A. `fdisk`
 - B. `cfdisk`
 - C. Disk Druid
 - D. `gnodisk`
19. The _____ command shows all loaded kernel modules.
20. Which command in `vi` searches for the text *Linux*?
- A. ESC, s, Linux
 - B. ESC, f, Linux
 - C. `\Linux`
 - D. `/Linux`
21. Which command must be run when a new mail alias is added?
- A. `newalias`
 - B. `newaliases`
 - C. `genaliases`
 - D. `kill -1 `cat /var/run/sendmail.pid``

22. You want to make the directory `/share` available via NFS. All users on your local network should be allowed to read and write files. Which of the following is correct, assuming that your local network is 192.168.1.0 and your machine is part of the DNS domain `mydomain.org`?
- A. `/local 192.168.1.0/255.255.255.0(rw)`
 - B. `/local 192.168.1.0/*(rw)`
 - C. `/local *.org(rw)`
 - D. `/local ALL.mydomain.org(rw)`
23. Which script is run when X Window starts?
- A. `.xinit`
 - B. `.xrc`
 - C. `.startxrc`
 - D. `.xinitrc`
24. Which command adds a new default route to the system?
- A. `ifconfig add default gw 1.2.3.4`
 - B. `route add gw 1.2.3.4`
 - C. `route add default gw 1.2.3.4`
 - D. `ifconfig add 0.0.0.0 mask 255.255.255.0 gw 1.2.3.4`
25. Dave is going away on vacation for two weeks and wants his mail forwarded to Sue while he is away. Which change should he make?
- A. Add “sue” to `/etc/aliases`.
 - B. Add “sue” to `~/.forward`.
 - C. Add “sue” to `~/.aliases`.
 - D. Add a rule to `pine` to forward the mail.
26. The Samba daemon responsible for handling name resolution is which of the following?
- A. `nmbd`
 - B. `smbd`
 - C. `winsd`
 - D. `named`
27. The _____ command can be used to change to another user.
28. Which command should be used to allow the host `norbert` to connect to the local X server?
- A. `xhost +`
 - B. `cat norbert > .rhosts`

- C. `xhost +norbert`
 - D. `rhosts +norbert`
29. You have just configured a new mail server named `deathstar`. Which entry should you make into DNS to make this the primary mail server?
- A. `IN MX 20 deathstar`
 - B. `IN MX 10 deathstar`
 - C. `IN MX 0 deathstar`
 - D. `IN MX 100 deathstar`
30. Which command would you use to copy the `/Documents` directory and all subdirectories from the current system to the remote system `newman`?
- A. `cp -R /Documents newman:/`
 - B. `scp -r /Documents root@newman:/`
 - C. `rcp -R /documents newman:/`
 - D. `scp -R root@newman:/ /Documents`
31. Which command would update your Debian system to the latest version of all installed packages?
- A. `apt-get update ; apt-get upgrade`
 - B. `apt-get upgrade ; apt-get update`
 - C. `apt-get refresh`
 - D. `apt get update ; apt get upgrade`
32. You have set up a new mail server behind your firewall. You now need to open up port _____ for access to work.
33. The _____ tool outputs a file showing the PnP configuration of the system.
34. What does the following IP chains block accomplish? (Select all that apply.)
- ```
ipchains -F input
echo 1 > /proc/sys/net/ipv4/ip_forward
ipchains -A output -src 192.168.1.0/24 -dst ! 192.168.1.0/24
-j MASQ
ipchains -A input -dst 192.168.1.12 -j ACCEPT
```
- A. The `192.168.1.0/24` network is being masq'd.
  - B. IP forwarding is disabled.
  - C. IP forwarding is enabled.
  - D. Access to `192.168.1.12` is allowed.

35. Which of the following directories should be on their own partition? (Select all that apply.)
- A. /usr
  - B. /boot
  - C. /home
  - D. /tmp
36. The \_\_\_\_\_ command provides an easy-to-use interface to Debian package management.
37. Which command(s) would be used to display which kernel packages are installed on your system?
- A. `rpm -ql | grep kernel`
  - B. `rpm -qa | grep kernel`
  - C. `dpkg -l kernel`
  - D. `dpkg -l | grep kernel`
38. To use the X11 graphical interface to make a kernel configuration, you would use which of the following?
- A. `make gconfig`
  - B. `make xconfig`
  - C. `make menuconfig`
  - D. `make tc1config`
39. After editing the `/etc/lilo.conf` file, you must run the \_\_\_\_\_ command.
40. If you want to remotely display an X Window application to another system, which option would you use?
- A. `-display`
  - B. `-connect`
  - C. `-remote`
  - D. `-xhost`
41. You have a Red Hat system that currently boots in text mode. Which file do you need to modify to have it boot in GUI mode?
- A. `/etc/runlevel`
  - B. `/etc/init.conf`
  - C. `/etc/inittab`
  - D. `/etc/xdm`

42. The \_\_\_\_\_ file stores the monitor information for X Window v4 (no path).
43. Which X11 application is known to have memory leaks?
- A. Netscape
  - B. Internet Explorer
  - C. Sawfish
  - D. WindowMaker
44. The \_\_\_\_\_ command displays all e-mail messages waiting in the mail queue.
45. Given the following files, would the host `norbert` be allowed to connect to the FTP server?
- ```
hosts.deny:      in.ftpd : ALL
hosts.allow:    in.ftpd : norbert
```
- A. Yes
 - B. No
46. Which type of modem is usually not compatible with Linux?
- A. Internal
 - B. External
 - C. Integrated
 - D. Winmodem
47. You need to access a resource named `Documents` shared by the NFS server `bigserver`. Which command would you use?
- A. `mount -t nfs //bigserver/Documents /mnt/docs`
 - B. `mount bigserver:Documents /mnt/docs`
 - C. `mount -t nfs bigserver:/Documents /mnt/docs`
 - D. `nfsmount bigserver:Documents /mnt/docs`
48. After updating the `/etc/exports` file, the _____ command must be run for it to take effect (with option(s)).
49. When examining the network logs for the firewall, you notice a lot of connection attempts to port 110. Which service may your remote users be trying to use?
- A. IMAP
 - B. DNS
 - C. POP3
 - D. SMTP

50. After editing the `/etc/modules.conf` file you should run the _____ command.
51. The first line of a shell script should be which of the following?
- A. `#!/bin/sh`
 - B. `#!/bash`
 - C. `#!/usr/bin/bash`
 - D. `#!/bin/bash`
52. User application defaults are stored in which file?
- A. `Xresources`
 - B. `.Xdefaults`
 - C. `.xinit`
 - D. `.xinitrc`

Exam 101 Answers

1. A. Choice A would run `cmd1` and then `cmd2` regardless of the output of `cmd1`. Choice B would run `cmd1` sending the output to `cmd2`. Choice C would append the output of `cmd1` to the file `cmd2`. Choice D would send the output of `cmd1` to the file `cmd2`.
2. 15. The default signal used with the `kill` command is 15, `sigterm`.
3. C. Choice A, the `grep` utility, searches files for the specified expression. Choice B, the `find` utility, searches directories for files. Choice C, the `locate` utility, searches the `slocate` database. This database is updated using the `updatedb` command. Choice D does not search for files.
4. B. Choice A, the `job` utility, is used to view backgrounded processes. Choice B, `nice`, is used to change the priority of a process as it is run. Choice C, `renice`, is used to change the priority of a process before it is run. Choice D, `top`, is used to view process information.
5. D. Choice A would change the ownership of `myfile` if the `userid` of the new owner were specified. Choice B would change the permissions of `myfile` to `-rw-rw---x`. Choice C would change the permissions on `myfile` to `--wx-wx--x`. Choice D would change the permissions on `myfile` to `-rw-rw-r--`.
6. A. The `/etc/passwd` file contains the user's home directory settings. Choice B, `/etc/skel`, contains files that are copied to all users' home directories. Choice C, `/etc/profile`, contains systemwide environment settings. Choice D, `/etc/bashrc`, contain global bash settings.
7. `crontab`. The `crontab` command is used to view jobs scheduled to run using `crond`.

8. **C.** Choice A does not follow the correct syntax for the `sed` command. Choice B would locate all instances of the letter *l* and replace them with *L*. Choice C would search for the word *lady* and replace it with *Lady*. Choice D would search for the word *lady* but would pipe it to the `tr` utility, not replace *lady* with *Lady*.
9. **D.** Choice A, `nice`, is used to set the priority of a process when it is run. Choice B, `renice`, would change the priority of a process after it has been run. Choice C, `ps -aux`, would view a snapshot of all running processes. Choice D, `top`, displays a real-time view of processes.
10. **D.** Choice A, `/etc/proc/`, is not valid. Choice B, `/mnt`, is simply a directory. Choice C, `/etc/fstab`, contains a listing of file systems that can be mounted at boot. Choice D, `/etc/mtab`, contains a listing of all currently mounted file systems.
11. **B.** Choice A, `echo`, can be used to view the assignment of environment variables. Choice B, `export`, makes the environment variable assignment available system-wide. Choice C, `cat`, is used to view the contents of a file. Choice D, `env`, is used to view all environment variables declared on the system.
12. **D.** Choice A uses an incorrect option for the `mv` command. Choice B would copy all files from `/ME` to `/YOU` and delete `/ME`, but directories would not be copied. Choice C would produce a syntax error because the last argument when moving a directory must be a directory name. Choice D would recursively copy the files and directories in `/ME` to `/YOU`, after which they would be deleted from `/ME`.
13. **B.** Choice A would create the `xmaslist` file from the contents of the `list` file while trying to create each line of the same length. Choice B would alphabetize the list, then number it and send the output to the `xmaslist` file. Choices C and D would produce errors because the `ln` command and the pipes are used incorrectly.
14. **C.** Only Choice C is correct. The `-9` signal is used to kill a process without allowing it to end gracefully. The default signal, used with answer A, is `-15`, `sigterm`, the `-now` signal is not valid, and the `-12` option is `sigusr2`.
15. **A, C, and D.** When creating a new partition for use, the `fdisk` utility is used to create the partition, the `mkfs` utility is used to format the partition, and the `mount` utility is used to mount the file system for use.
16. **du.** The `du` utility is used to display directory space usage information.
17. **C.** The `quota` command is used to view user quotas. The `quotaon` command is used to enable quotas. The `edquota` command is used to edit disk quotas. The `vi` editor is started by the `edquota` command when editing quotas.
18. **D.** The `umask` of `002` would be used to filter the write permission for other users.
19. **A.** The `/etc/shadow` file must be readable by only the root user.

20. **D.** The `shutdown` command, when used alone, will switch to single-user mode. The `telinit` command can be used to change runlevels, with level 6 used for rebooting and level 0 used to halt or shut down the system.
21. **A and D.** The `useradd` command is used to create new user accounts while the `passwd` command is used to set the user's password.
22. **B.** The permissions of read and execute have a numeric value of 5 while the read permission has a value of 4.
23. **C.** The `more` utility is used to view a file one page at a time. The `grep` utility is used to search a file. The `man` utility is used to present manual pages that contain information about commands. The `help` utility displays information only for built-in commands.
24. **/etc/skel.** Files that are to be copied to each user's home directory should be placed in the `/etc/skel` directory.
25. **C.** The shell prompt is configured with the `PS1` environment variable. The `PATH` variable is used for the command path while the home directory is contained in the `HOME` variable.
26. **A, C, and D.** The `cd` command, when used alone, changes the user's `pwd` to their home directory. The `$HOME` variable and the `~` also can be used to represent the user's home directory. The `cd ..` command is used to move up one directory in the directory tree.
27. **C.** The `/etc` directory contains many of the configuration files for the system. The `/root` directory is the home directory for the root user. The `/sbin` directory contains utilities used by the superuser. The `/usr/local` directory is used to store software that isn't part of the operating system.
28. **B.** The `/etc/passwd` file is used to store shell script assignments along with home directory assignments. The `/etc/skel` directory contains the files to be copied to each user's home directory. The `/etc/profile` file contains global user settings. The LILO boot loader is used to configure booting options.
29. **D.** The `cp` and `mv` commands can be used to relocate the files. Hard links cannot span file systems. Using soft links, you can have the files appear to be stored on one file system while actually existing on another.
30. **A.** The `userdel -r` command is used to remove a user's home directory as well as their account. The `usermod` command can be used to disable a user's account. The `passwd` utility is used to change a user's password.
31. **od.** The `od` utility allows files to be viewed in octal and hexadecimal formats.
32. **A.** The `grep` utility is used to search a file for the specified string. The `locate` command searches the `slocate` database for specified files. The `find` command searches directories for the specified files. The `where` command is invalid.

33. **D.** The `&` character is used to background a process when it is run. The `fg` command is used to bring a process to the foreground. The `bg` command is used to background a suspended process. The `Ctrl+Z` sequence can be used to suspend a foreground process.
34. **B.** Man pages can be used to discover more information on commands and utilities. The HOWTOs contain information on configuring specific software and hardware. The `/usr/doc` directory contains information on installed applications. The `dmesg` command is used to display system messages.
35. **B.** The `&` character is used when running a process to specify that it runs in the background.
36. **C.** The `HOME` variable stores the user's home directory. The `PS1` variable stores the command prompt settings. The `PATH` variable contains the paths to search when running commands. The `TERM` variable stores the terminal setting.
37. **chown.** Ownership of files and directories is changed using the `chown` command.
38. **A.** The `dmesg` command is used to view system messages. Neither `messages` nor `errors` is a valid command. The `lilo` command is used when updating boot loader options using the `/etc/lilo.conf` file.
39. **B.** Only the `tar` utility is capable of creating an archive while compressing the file. The `gzip`, `compress`, and `bzip2` utilities are used to compress and uncompress files.
40. **D.** The `dmesg` command is used to view system messages. The `ls` command is used to view directory contents. The `bg` command is used to run a process in the background. The `jobs` command is used to view jobs running in the background.
41. **C.** The `tac` utility displays files in reverse. The `cat` utility is used to display file contents. The `head` utility is used to view the beginning of a file. The `top` utility is used to view real-time information on system processes.
42. **C.** Stop limits and max limits are invalid. Soft limits can be exceeded for a period of time, while hard limits prevent a user from creating new files.
43. **B.** Runlevel 0 is used to halt the system; level 1 is used to enter single-user mode; level 3 can vary according to the distribution; and level 6 is used to reboot the system.
44. **C.** The `/etc/fstab` file contains file system information. The `/etc/passwd` file contains information on user accounts. The `/quota.user` file contains quota information for users. The `/lilo` file is invalid.
45. **D.** The man page can be specified using the syntax `man NUMBER command`. This correct syntax is shown as `man 3 syslog`.
46. **D.** The `cat`, `more`, and `less` utilities are all used to view file contents. The `expand` utility is used to convert tab characters to spaces.
47. **umount.** File systems are unmounted using the `umount` command.

- 48. `history`. The `history` command allows you to view previously entered commands.
- 49. **B.** The Web site `www.deja.com` enables you to search Usenet postings.
- 50. **A.** The `/proc` directory is virtual. It is used to store information about system resources and processes.
- 51. `fsck`. The `fsck` utility is used to verify and correct problems on file systems.
- 52. **A.** The `-R` option specifies that a recursive directory listing be performed. Data is sent to a file when the `>` symbol is used for redirection.

Exam 102 Answers

- 1. `mkfs`. The `mkfs` tool is used to create a file system.
- 2. **C.** The `-r` parameter on `lilo` specifies a different root directory. This is used when you are doing recovery and `/etc` and `/boot` are actually mounted under a new root.
- 3. `ld.so.conf`. Whenever a new library is added manually, you must edit the `ld.so.conf` file and run `ldconfig`.
- 4. **B.** The `-e` parameter is used to remove a package with RPM. The other choices install packages.
- 5. **B.** By default, kernel configuration options are stored in the `.config` file.
- 6. **D.** The `-qf` parameters tell RPM to query (`-q`) the database to see who owns a certain file (`-f`).
- 7. `~/.bashrc` or `/etc/bashrc`. The `.bashrc` script is executed when a bash shell is started, but not when logging in. For example, if you open a new `xterm` in X11 with `bash`, this script will run.
- 8. **A.** `traceroute` is used to show the path a packet takes to a destination. The `ping` command tells you only if a remote host is up, and the `route` command is used to modify the local routing table.
- 9. **C.** The `whois` command shows contact information relating to a domain. The `nslookup` and `dig` tools are used to check DNS information. The `whoinfo` command is invalid.
- 10. **A and C.** When a change is made to `inetd.conf`, the daemon must be restarted by either using the script or issuing a HUP.
- 11. `/etc/nologin`. This file causes the system to disallow all nonroot logins.
- 12. **C and D.** The `pwconv` and `grpconv` tools convert the `passwd` and `group` files to use shadow passwords. The other commands are fake.
- 13. **A, C, and D.** All three methods can be used to un`gzip` the file. The `tar` command does support `gzip`, but is used only when uncompressing a `tar` file.

14. **swapon -a.** The `swapon -a` command enables all swap partitions in the `/etc/fstab` file.
15. **C.** The `last` command shows the `utmp` file by default, and the `who` command uses the `wtmp` file. The other two commands are invalid.
16. **D.** The Apache package includes the `apachectl` tool to manage the `httpd` daemon. The other answers are invalid.
17. **A.** The `lpq` tool is used to manage and view the print queue, while `lpc` is used to manage the printing system. The `lpq -a` command will display information for all local printers. The `printq` command is invalid.
18. **C.** Disk Druid is a GUI tool supplied by Red Hat. Answers A and B provide text interfaces. Answer D is invalid.
19. **lsmod.** The `lsmod` command displays all loaded modules.
20. **D.** The forward slash tells `vi` to search the file for the given expression. The `s` command is used for substitution, while the `f` command searches for a single character in the current line. Answer C is invalid.
21. **B.** Any time an alias is added, the `newaliases` command needs to be run. Answers A and C are invalid. Restarting the `sendmail` process will not activate the new aliases.
22. **A.** Choice B is invalid due to the subnet mask. Choice C allows anyone from a `.org` domain to have access. The `ALL` wildcard is invalid as well.
23. **D.** The `.xinitrc` script is run from your home directory when `startx` is called. The other script names are invalid.
24. **C.** The `ifconfig` command is used to configure an interface, not a route.
25. **B.** Only root can change `/etc/aliases`. Answers C and D are invalid.
26. **A.** The `smbd` daemon handles file and printer sharing. Answer C is invalid. The `named` daemon is used to provide DNS.
27. **su.** The `su`, or substitute user, command is used to change to another user while logged in.
28. **C.** Choice A allows anyone to connect, which is a security risk. The X server does not use the `rhosts` file. Choice D is invalid.
29. **C.** The lower the number is, the higher the priority of the mail server.
30. **B.** The `cp` tool only works locally. Choice C uses the wrong path name, and choice D has incorrect syntax.
31. **A.** The `update` command is first used to update the available package list, and then the `upgrade` command is used to actually upgrade the packages.
32. **25.** The SMTP protocol uses port 25.
33. **pnpdump.** This tool is used to scan the system and output any Plug-and-Play information to a text file for configuration.

34. **C and D.** IP Masq is not functioning since the rule must be set in the Forward chain. The second line enables IP forwarding.
35. **A, B, C, and D.** In most cases all of the directory structures should have their own partition for security and fault tolerance.
36. **dselect.** The `dselect` tool is an easy-to-use front end for `dpkg`.
37. **B and D.** Answer A is invalid, since no query string is present. Answer C will only display a package with the exact name “kernel”.
38. **B.** Answers A and D are invalid. The `make menuconfig` command provides a text mode menu.
39. **lilo.** You must rerun `lilo` after editing `/etc/lilo.conf` so that a new boot record is written with the updated information.
40. **A.** The `-display` parameter tells the application where to connect to for its output display. The other answers are invalid.
41. **C.** The `inittab` file tells the system which runlevel to boot to by default. In Red Hat, runlevel 3 is text mode and runlevel 5 is GUI mode. The other answers are invalid.
42. **XF86Config-4.** XFree86 v3 uses the `XF86Config` file.
43. **A.** Netscape is known to have stability problems and memory leaks.
44. **mailq.** The `mailq` command shows all messages in the mail queue, as well as any errors that may keep the messages from being delivered.
45. **A.** While the `hosts.deny` rule blocks everyone, the `hosts.allow` rule overrides it.
46. **D.** Winmodems use software to handle most of their functionality, which is not supported in Linux due to lack of information from which to write drivers.
47. **B.** The `mount` command uses the syntax `server:share mount_point` for NFS shares.
48. **exportfs -a.** The `exportfs` command is used to control the directories shared via NFS. The `-a` option tells `exportfs` to share or unshare all configured directories.
49. **C.** IMAP uses port 143, DNS uses port 53, and SMTP uses port 25.
50. **depmod.** The `depmod` command sets up the module dependency links.
51. **D.** The other answers are invalid.
52. **B.** The `Xdefaults` file lists the default settings for X applications, as configured by the user. The `xinit` and `xinitrc` scripts are used to start X.



Objective Mapping

In this appendix, you'll find two tables listing the exam objectives for each of the LPI certification exams. Each table is an exhaustive cross-reference chart that links every exam objective to the corresponding materials in this book where the subject matter is covered.

The tables you'll find in this appendix are the following:

- ◆ Table C-1: Exam 101
- ◆ Table C-2: Exam 102



Table C-1
Exam 101

<i>Exam Objective</i>	<i>Chapter</i>	<i>Section</i>
Topic 1.3: GNU and UNIX Commands		
<p>Obj 1: Work Effectively on the UNIX Command Line. Interact with shells and commands using the command line. Includes typing valid commands and command sequences, defining, referencing and exporting environment variables, using command history and editing facilities, invoking commands in the path and outside the path, using command substitution, and applying commands recursively through a directory tree.</p>	Chapter 2	All sections
<p>Obj 2: Process Text Streams Using Text Processing Filters. Send text files and output streams through text utility filters to modify the output in a useful way. Includes the use of standard UNIX commands found in the GNU textutils package such as sed, sort, cut, expand, fmt, head, join, nl, od, paste, pr, split, tac, tail, tr, and wc.</p>	Chapter 4	All sections
<p>Obj 3: Perform Basic File Management. Use the basic UNIX commands to copy and move files and directories. Perform advanced file management operations such as copying multiple files recursively and moving files that meet a wildcard pattern. Use simple and advanced wildcard specifications to refer to files.</p>	Chapter 6	Managing Files
<p>Obj 4: Use UNIX Streams, Pipes, and Redirects. Connect files to commands and commands to other commands to efficiently process textual data. Includes redirecting standard input, standard output, and standard error; and piping one command's output into another command as input or as arguments (using xargs); sending output to stdout and a file (using tee).</p>	Chapter 4	Working with Input and Output
<p>Obj 5: Create, Monitor, and Kill Processes. Includes running jobs in the foreground and background, bringing a job from the background to the foreground and vice versa, monitoring active processes, sending signals to processes, and killing processes. Includes using commands ps, top, kill, bg, fg, and jobs.</p>	Chapter 2	Managing Processes

<i>Exam Objective</i>	<i>Chapter</i>	<i>Section</i>
Obj 6: Modify Process Execution Priorities. Run a program with higher or lower priority, determine the priority of a process, change the priority of a running process. Includes the command nice and its relatives.	Chapter 2	Modifying Process Priorities
Obj 7: Perform Searches of Text Files Making Use of Regular Expressions. Includes creating simple regular expressions and using related tools such as grep and sed to perform searches.	Chapter 4	Working with Input and Output Enhancing Searches with Regular Expressions
Topic 2.4: Devices, Linux File Systems, Filesystem Hierarchy Standard		
Obj 1: Create partitions and filesystems. Create disk partitions using fdisk, create hard drive and other media filesystems using mkfs.	Chapter 5	Creating Partitions and File Systems
Obj 2: Maintain the integrity of filesystems. Verify the integrity of filesystems, monitor free space and inodes, fix simple filesystem problems. Includes commands fsck, du, df.	Chapter 5	Checking the File System
Obj 3: Control filesystem mounting and unmounting. Mount and unmount filesystems manually, configure filesystem mounting on bootup, configure user-mountable removable file systems. Includes managing file /etc/fstab.	Chapter 5	Mounting and Unmounting File Systems
Obj 4: Set and view disk quota. Setup disk quota for a filesystem, edit user quota, check user quota, generate reports of user quota. Includes quota, edquota, repquota, quotaon commands.	Chapter 6	Managing Quotas
Obj 5: Use file permissions to control access to files. Set permissions on files, directories, and special files, use special permission modes such as suid and sticky bit, use the group field to grant file access to workgroups, change default file creation mode. Includes chmod and umask commands. Requires understanding symbolic and numeric permissions.	Chapter 6	Working with Permissions
Obj 6: Manage file ownership. Change the owner or group for a file, control what group is assigned to new files created in a directory. Includes chown and chgrp commands.	Chapter 6	Working with Permissions

Continued

Table C-1 (continued)

<i>Exam Objective</i>	<i>Chapter</i>	<i>Section</i>
Obj 7: Create and change hard and symbolic links. Create hard and symbolic links, identify the hard links to a file, copy files by following or not following symbolic links, use hard and symbolic links for efficient system administration.	Chapter 6	Creating File Links
Obj 8: Find system files and place files in the correct location. Understand the filesystem hierarchy standard, know standard file locations, know the purpose of various system directories, find commands and files. Involves using the commands: find, locate, which, updatedb . Involves editing the file: /etc/updatedb.conf.	Chapter 6	Locating Files
Topic 2.6: Boot, Initialization, Shutdown, Run Levels		
Obj 1: Boot the system. Guide the system through the booting process, including giving options to the kernel at boot time, and check the events in the log files. Involves using the commands: dmesg (lilo). Involves reviewing the files: /var/log/messages, /etc/lilo.conf, /etc/conf.modules /etc/modules.conf.	Chapter 8	All sections
Obj 2: Change runlevels and shutdown or reboot system. Securely change the runlevel of the system, specifically to single user mode, halt (shutdown) or reboot. Make sure to alert users beforehand, and properly terminate processes. Involves using the commands: shutdown, init.	Chapter 8	Understanding Runlevels and init
Topic 1.8: Documentation		
Obj 1: Use and Manage Local System Documentation. Use and administer the man facility and the material in /usr/doc/. Includes finding relevant man pages, searching man page sections, finding commands and manpages related to one, configuring access to man sources and the man system, using system documentation stored in /usr/doc/ and related places, determining what documentation to keep in /usr/doc/.	Chapter 7	Using Documentation Stored in /usr/doc
Obj 2: Find Linux documentation on the Internet. Find and use Linux documentation at sources such as the Linux Documentation Project, vendor and third-party websites, newsgroups, newsgroup archives, mailing lists.	Chapter 7	Documentation on the Internet

<i>Exam Objective</i>	<i>Chapter</i>	<i>Section</i>
Obj 3: Write System Documentation. Write documentation and maintain logs for local conventions, procedures, configuration and configuration changes, file locations, applications, and shell scripts.	Chapter 7	Creating Documentation
Obj 4: Provide User Support. Provide technical assistance to users via telephone, email, and personal contact.	Chapter 7	Providing Technical Support
Topic 2.11: Administrative Tasks		
Obj 1: Manage users and group accounts and related system files. Add, remove, suspend user accounts, add and remove groups, change user/group info in passwd/group databases, create special purpose and limited accounts. Includes commands useradd, userdel, groupadd, gpasswd, passwd, and file passwd, group, shadow, and gshadow.	Chapter 10	All sections
Obj 2: Tune the user environment and system environment variables. Modify global and user profiles to set environment variable, maintain skel directories for new user accounts, place proper commands in path. Involves editing /etc/profile and /etc/skel/.	Chapter 10	Configuring Global and User Settings
Obj 3: Configure and use system log files to meet administrative and security needs. Configure the type and level of information logged, manually scan log files for notable activity, arrange for automatic rotation and archiving of logs, track down problems noted in logs. Involves editing /etc/syslog.conf.	Chapter 11	System Logging
Obj 4: Automate system administration tasks by scheduling jobs to run in the future. Use cron to run jobs at regular intervals, use at to run jobs at a specific time, manage cron and at jobs, configure user access to cron and at services.	Chapter 11	Scheduling Jobs
Obj 5: Maintain an effective data backup strategy. Plan a backup strategy, backup filesystems automatically to various media, perform partial and manual backups, verify the integrity of backup files, partially or fully restore backups.	Chapter 11	Performing Backups

Table C-2
Exam 102

<i>Exam Objective</i>	<i>Chapter</i>	<i>Section</i>
Topic 1.1: Hardware & Architecture		
Obj 1: Configure fundamental system hardware. Demonstrate a proper understanding of important BIOS settings, set the date and time, ensure IRQ's and I/O addresses are correct for all ports including serial and parallel, make a note of IRQ's and I/O's, be aware of the issues associated with drives larger than 1024 cylinders.	Chapter 1	Preparing Hardware
Obj 2: Setup SCSI and NIC Devices. Manipulate the SCSI BIOS to detect used and available SCSI ID's, set the SCSI ID to the correct ID number for the boot device and any other devices required, format the SCSI drive - low level with manufacturer's installation tools - and properly partition and system format with Linux fdisk and mke2fs, set up NIC using manufacturer's setup tools setting the I/O and the IRQ as well as the DMA if required.	Chapter 1	Preparing Hardware
Obj 3: Configure Modem, Sound cards. Ensure devices meet compatibility requirements (particularly that the modem is NOT a win-modem), verify that both the modem and sound card are using unique and correct IRQ's, I/O, and DMA addresses, if the sound card is PnP install and run sndconfig and isapnp, configure modem for outbound dial-up, configure modem for outbound PPP SLIP CSLIP connection, set serial port for 115.2 Kbps.	Chapter 1	Modems, Sound cards
	Chapter 15	pppd
Topic 2.2: Linux Installation and Package Management		
Obj 1: Design hard-disk lay-out. Design a partitioning scheme for a Linux system, depending on the hardware and system use (number of disks, partition sizes, mount points, kernel location on disk, swap space).	Chapter 1	Partitioning Schemes
Obj 2: Install a boot manager. Select, install and configure a boot loader at an appropriate disk location. Provide alternative and backup boot options (like a boot floppy disk). Involves using the command: lilo. Involves editing the file: /etc/lilo.conf.	Chapter 8	Troubleshooting the Boot Process

<i>Exam Objective</i>	<i>Chapter</i>	<i>Section</i>
<p>Obj 3: Make and install programs from source. Manage (compressed) archives of files (unpack “tarballs”), specifically GNU source packages. Install and configure these on your systems. Do simple manual customization of the Makefile if necessary (like paths, extra include dirs) and make and install the executable. Involves using the commands: gunzip, tar, ./configure, make, make install. Involves editing the files: ./Makefile.</p>	Chapter 3	Installing Software from Source Code
<p>Obj 4: Manage shared libraries. Determine the dependencies of executable programs on shared libraries, and install these when necessary. Involves using the commands: ldd, ldconfig. Involves editing the files: /etc/ld.so.conf.</p>	Chapter 3	Managing Shared Libraries
<p>Obj 5: Use Debian package management. Use the Debian package management system, from the command line (dpkg) and with interactive tools (dselect). Be able to find a package containing specific files or software; select and retrieve them from archives; install, upgrade or uninstall them; obtain status information like version, content, dependencies, integrity, installation status; and determine which packages are installed and from which package a specific file has been installed. Be able to install a non-Debian package on a Debian system. Involves using the commands and programs: dpkg, dselect, apt, apt-get, alien. Involves reviewing or editing the files and directories: /var/lib/dpkg/*.</p>	Chapter 3	Debian Package Management
<p>Obj 6: Use Red Hat Package Manager (rpm). Use rpm, from the command line. Familiarize yourself with these tasks: Install a package, uninstall a package, determine the version of the package and the version of the software it contains, list the files in a package, list documentation files in a package, list configuration files or installation or uninstallation scripts in a package, find out for a certain file from which package it was installed, find out which packages have been installed on the system (all packages, or from a subset of packages), find out in which package a certain program or file can be found, verify the integrity of a package, verify the PGP or GPG signature of a package, upgrade a package. Involves using the commands and programs: rpm, grep.</p>	Chapter 3	Red Hat Package Manager

Continued

Table C-2 (continued)

<i>Exam Objective</i>	<i>Chapter</i>	<i>Section</i>
Topic 1.5: Kernel		
Obj 1: Manage kernel modules at runtime. Learn which functionality is available through loadable kernel modules, and manually load and unload the modules as appropriate. Involves using the commands: <code>lsmod</code> , <code>insmod</code> , <code>rmmmod</code> , <code>modinfo</code> , <code>modprobe</code> . Involves reviewing the files: <code>/etc/modules.conf</code> <code>/etc/conf.modules</code> (* depends on distribution *), <code>/lib/modules/{kernel-version}/modules.dep</code> .	Chapter 13	Managing modules
Obj 2: Reconfigure, build and install a custom kernel and modules. Obtain and install approved kernel sources and headers (from a repository at your site, CD, <code>kernel.org</code> , or your vendor); Customize the kernel configuration (i.e., reconfigure the kernel from the existing <code>.config</code> file when needed, using <code>oldconfig</code> , <code>menuconfig</code> or <code>xconfig</code>); Make a new Linux kernel and modules; Install the new kernel and modules at the proper place; Reconfigure and run <code>lilo</code> . N.B.: This does not require to upgrade the kernel to a new version (full source nor patch). Requires the commands: <code>make</code> (<code>dep</code> , <code>clean</code> , <code>menuconfig</code> , <code>bzImage</code> , <code>modules</code> , <code>modules_install</code>), <code>depmod</code> , <code>lilo</code> . Requires reviewing or editing the files: <code>/usr/src/linux/.config</code> , <code>/usr/src/linux/Makefile</code> , <code>/lib/modules/{kernelversion}/modules.dep</code> , <code>/etc/conf.modules</code> <code>/etc/modules.conf</code> , <code>/etc/lilo.conf</code> .	Chapter 13	Reconfiguring and Installing a New Kernel
Topic 1.7: Text editing, Processing, Printing		
Obj 1: Perform basic file editing operations using vi. Edit text files using <code>vi</code> . Includes <code>vi</code> navigation, basic modes, inserting, editing and deleting text, finding text, and copying text.	Chapter 14	Using <code>vi</code>
Obj 2: Manage printers and print queues. Monitor and manage print queues and user print jobs, troubleshoot general printing problems. Includes the commands: <code>lpc</code> , <code>lpq</code> , <code>lprm</code> and <code>lpr</code> . Includes reviewing the file: <code>/etc/printcap</code> .	Chapter 12	Installing Printers Managing the Printer Services Printing Files
Obj 3: Print files. Submit jobs to print queues, convert text files to postscript for printing. Includes <code>lpr</code> command.	Chapter 12	Printing Files

<i>Exam Objective</i>	<i>Chapter</i>	<i>Section</i>
<p>Obj 4: Install and configure local and remote printers. Install a printer daemon, install and configure a print filter (e.g.: apsfilter, magicfilter). Make local and remote printers accessible for a Linux system, including postscript, non-postscript, and Samba printers. Involves the daemon: lpd . Involves editing or reviewing the files and directories: /etc/printcap , /etc/apsfilterrc , /usr/lib/apsfilter/filter/*/ , /etc/magicfilter/*/ , /var/spool/lpd/*/.</p>	Chapter 12	Installing Printers Using Print Filters
Topic 1.9: Shells, Scripting, Programming, Compiling		
<p>Obj 1: Customize and use the shell environment. Customize your shell environment: set environment variables (e.g. PATH) at login or when spawning a new shell; write bash functions for frequently used sequences of commands. Involves editing these files in your home directory: .bash_profile .bash_login .profile ; .bashrc ; .bash_logout ; .inputrc.</p>	Chapter 14	Customizing the Shell Environment
<p>Obj 2: Customize or write simple scripts. Customize existing scripts (like paths in scripts of any language), or write simple new (ba)sh scripts. Besides use of standard sh syntax (loops, tests), be able to do things like: command substitution and testing of command return values, test of file status, and conditional mailing to the superuser. Make sure the correct interpreter is called on the first (!) line, and consider location, ownership, and execution- and suid-rights of the script.</p>	Chapter 14	Writing Simple Scripts
Topic 2.10: X		
<p>Obj 1: Install & Configure XFree86. Verify that the video card and monitor are supported by an X server, install the correct X server, configure the X server, install an X font server, install required fonts for X (may require a manual edit of /etc/X11/XF86Config in the "Files" section), customize and tune X for videocard and monitor. Commands: XF86Setup, xf86config. Files: /etc/X11/XF86Config, .xresources.</p>	Chapter 9	Installing X Versions of XFree86 Configuring X
<p>Obj 2: Setup XDM. Turn xdm on and off, change the xdm greeting, change default bitplanes for xdm, set-up xdm for use by X-stations.</p>	Chapter 9	Starting X Running X and Clients Remotely

Continued

Table C-2 (continued)

<i>Exam Objective</i>	<i>Chapter</i>	<i>Section</i>
Obj 3: Identify and terminate runaway X applications. Identify and kill X applications that won't die after user ends an X-session. Example: netscape, tkcat, etc.	Chapter 9	Using X
Obj 4: Install & Customize a Window Manager Environment. Select and customize a system-wide default window manager and/or desktop environment, demonstrate an understanding of customization procedures for window manager menus, configure menus for the window manager, select and configure the desired x-terminal (xterm, rxvt, aterm etc.), verify and resolve library dependency issues for X applications, export an X-display to a client workstation. Commands: Files: .xinitrc, .Xdefaults, various .rc files.	Chapter 9	Using X Running X and Clients Remotely
Topic 1.12: Networking Fundamentals		
Obj 1: Fundamentals of TCP/IP. Demonstrate an understanding of network masks and what they mean (i.e. determine a network address for a host based on its subnet mask), understand basic TCP/IP protocols (TCP, UDP, ICMP) and also PPP, demonstrate an understanding of the purpose and use of the more common ports found in /etc/services (20, 21, 23, 25, 53, 80, 110, 119, 139, 143, 161), demonstrate a correct understanding of the function and application of a default route. Execute basic TCP/IP tasks: FTP, anonymous FTP, telnet, host, ping, dig, traceroute, whois.	Chapter 15	The TCP/IP Protocol Suite
	Chapter 16	Using DNS
Obj 3: TCP/IP Troubleshooting & Configuration. Demonstrate an understanding of the techniques required to list, configure and verify the operational status of network interfaces, change, view or configure the routing table, check the existing route table, correct an improperly set default route, manually add/start/stop/restart/delete/reconfigure network interfaces, and configure Linux as a DHCP client and a TCP/IP host and debug associated problems. May involve reviewing or configuring the following files or directories: /etc/HOSTNAME /etc/hostname, /etc/hosts, /etc/networks, /etc/host.conf, /etc/resolv.conf, and other network configuration files for your distribution. May involve the use of the following commands and programs: dhcpcd, host, hostname (domainname, dnsdomainname), ifconfig, netstat, ping, route, traceroute, the network scripts run during system initialization.	Chapter 15	Configuration and Troubleshooting

<i>Exam Objective</i>	<i>Chapter</i>	<i>Section</i>
Obj 4: Configure and use PPP. Define the chat sequence to connect (given a login example), setup commands to be run automatically when a PPP connection is made, initiate or terminate a PPP connection, initiate or terminate an ISDN connection, set PPP to automatically reconnect if disconnected.	Chapter 15	Configuring PPP
Topic 1.13: Networking Services		
Obj 1: Configure and manage inetd and related services. Configure which services are available through inetd, use tcpwrappers to allow or deny services on a host-by-host basis, manually start, stop, and restart internet services, configure basic network services including telnet and ftp. Includes managing inetd.conf, hosts.allow, and hosts.deny.	Chapter 16	Using the Internet Super Server
	Chapter 17	Configuring TCP wrappers
Obj 2: Operate and perform basic configuration of sendmail. Modify simple parameters in sendmail config files (modify the DS value for the "Smart Host" if necessary), create mail aliases, manage the mail queue, start and stop sendmail, configure mail forwarding (.forward), perform basic troubleshooting of sendmail. Does not include advanced custom configuration of sendmail. Includes commands mailq, sendmail, and newaliases. Includes aliases and mail/config files.	Chapter 16	Using sendmail
Obj 3: Operate and perform basic configuration of apache. Modify simple parameters in apache config files, start, stop, and restart httpd, arrange for automatic restarting of httpd upon boot. Does not include advanced custom configuration of apache. Includes managing httpd conf files.	Chapter 16	Using Apache
Obj 4: Properly manage the NFS, smb, and nmb daemons. Mount remote filesystems using NFS, configure NFS for exporting local filesystems, start, stop, and restart the NFS server. Install and configure Samba using the included GUI tools or direct edit of the /etc/smb.conf file (Note: this deliberately excludes advanced NT domain issues but includes simple sharing of home directories and printers, as well as correctly setting the nmbd as a WINS client).	Chapter 16	Using NFS
		Using Samba

Continued

Table C-2 (continued)

<i>Exam Objective</i>	<i>Chapter</i>	<i>Section</i>
<p>Obj 5: Setup and configure basic DNS services. Configure hostname lookups by maintaining the /etc/hosts, /etc/resolv.conf, /etc/host.conf, and /etc/nsswitch.conf files, troubleshoot problems with local caching-only name server. Requires an understanding of the domain registration and DNS translation process. Requires understanding key differences in config files for bind 4 and bind 8. Includes commands nslookup, host. Files: named.boot (v.4) or named.conf (v.8).</p>	Chapter 16	Using DNS
Topic 1.14: Security		
<p>Obj 1: Perform security admin tasks. Configure and use TCP wrappers to lock down the system, list all files with SUID bit set, determine if any package (.rpm or .deb) has been corrupted, verify new packages prior to install, use setgid on dirs to keep group ownership consistent, change a user's password, set expiration dates on user's passwords, obtain, install and configure ssh.</p>	Chapter 17	Performing Security Administration Tasks
<p>Obj 2: Setup host security. Implement shadowed passwords, turn off unnecessary network services in inetd, set the proper mailing alias for root and setup syslogd, monitor CERT and BUGTRAQ, update binaries immediately when security problems are found.</p>	Chapter 17	Providing Host Security
<p>Obj 3: Setup user level security. Set limits on user logins, processes, and memory usage.</p>	Chapter 17	Limiting Users



Exam Tips

This appendix contains helpful information about the LPIC exam and the testing process.

Where Can I Take the Test?

The LPI certification exams are available globally in both English and Japanese at the 2,400 testing centers operated by Virtual University Enterprises (VUE). To locate the nearest testing center visit the VUE Web site at

<http://www.vue.com/testing/index.html>

How Do I Register?

Registration is done online and requires a VUE user account. This can be created at the following VUE Web site.

<http://www.vue.com/contact/obtainLogin.html>

If you have previously used the VUE testing centers for other tests, it is possible to use the existing account information that has been created for you. Once you have the necessary account created, you can register for the exam at the following:

<http://www.vue.com/linux/linuxexam.html>

Once you have registered for the exam, you can cancel at any time until the day before the scheduled exam without forfeiting the exam fee. Once you have registered for the exam, you will receive a confirmation along with instructions on cancellations.

How Much Does It Cost?

The cost of each English exam is \$100 U.S. The cost of each Japanese exam is 15,000 yen.

How Long Is the Test?

It is recommended that you plan on spending a maximum of 1.5–2 hours for each exam.

Can I Bring Anything with Me into the Testing Center?

When taking the exam you are not allowed to bring any materials into the testing center. You will be supplied with a pencil and scratch paper upon request.

How Do I Get My Results?

The results of the tests are immediately available upon completion of the exam. You will also receive a printed copy of your exam scores to save for your records.

What Happens If I Pass?

Level 1 certification is awarded after both exams have been passed. When you have successfully passed both exams, you will receive certification material from the Linux Professional Institute.

What Do I Do If I Fail?

If you do not pass the exam, you should carefully review the objectives and focus on any areas that require improvement. You will need to schedule a time with VUE to retake the exam and will be required to pay the examination fee again.

When preparing to retake the exam, you will want to carefully review all exam objectives. Spend more time working with Linux in a hands-on environment. Spend time familiarizing yourself with the Linux system and commands.

Can I Retake the Test? How Often?

If you fail any exam, you are eligible to retake that exam. No limits currently exist on when you can retake the exam or how often it can be done. Be sure to remember that you must pay each time you take an exam, so be sure not to schedule to retake the exam until you are properly prepared.

What If I Have a Problem with the Test, or a Question on the Test?

Information concerning the LPIC exams can be located at the LPI Web site.

<http://www.lpi.org>

If further assistance is required, you can send e-mail to any of the following addresses:

Dan York, President, dan@lpi.org

Kara Pritchard, Director of Exam Development, kara@lpi.org

General Information, General information about LPI, info@lpi.org

What's the Next Step (the Next Exam To Take)?

Once you have completed both the 101 and 102 exams you will receive the LPI Level 1 certification. You can then begin to prepare for the Level 2 and Level 3 certifications. Each level consists of two exams with each level requiring more in-depth knowledge of Linux systems. More information about these exams can be located at the following Web site:

<http://www.lpi.org>



Glossary

. The period character used to specify the current directory.

.. Two period characters used to specify the parent of the current directory.

A record DNS address record providing IP-address-to-domain-name mapping.

Accelerated Graphics Port (AGP) An interface used with graphics adapters that utilizes the system's RAM to provide faster graphics displays.

access control list (ACL) A table used to specify which users and processes have access right to files, directories, and devices that exist on the operating system. It also specifies which rights these users and processes have.

ACL See access control list.

Address Resolution Protocol (ARP) A protocol in the TCP/IP suite that provides hardware address-to-TCP/IP address mapping.

AGP See Accelerated Graphics Port.

alias An alternative or more relevant reference to a defined device or command.

alien A program that converts Linux application packages between RPM, DEB, TGZ, and SLP.

Alpha The RISC-based CPU produced by Digital Equipment Company, which is now owned by Compaq.

Apache The freely available Web server created by the Apache Software Foundation, which is distributed under the open source license.

ARCnet A proprietary token-bus networking architecture that supports coaxial, twisted-pair, and fiber-optic cable-based implementations and is capable of 2.5 Mbps transmissions.

argument An optional parameter used with commands.

ARP See Address Resolution Protocol.

ASCII A standard format for text files developed by ANSI (American National Standards Institute).

AT Attachment Packet Interface (ATAPI) An interface used by computers for hard drives, CD-ROM, CD-RW, and tape devices that utilizes the IDE and Enhanced IDE interfaces.

ATAPI See AT Attachment Packet Interface.

bash (Bourne-Again Shell) The shell used by UNIX systems to provide the shell environment.

Berkley Internet Name Domain See BIND.

binary A numbering scheme that uses ones and zeros as the possible values of the bits.

BIND (Berkeley Internet Name Domain) A free DNS server.

bit A binary digit, the smallest data unit on a computer.

boot The process of loading an operating system into the computer's RAM.

Bourne-Again Shell See bash.

byte A unit of data that is made of eight bits.

C Shell See csh.

canonical name See CNAME.

CD-ROM A compact disc using read-only memory.

CD-RW A rewriteable compact disc that can be written to repeatedly.

central processing unit See CPU.

CGI See Common Gateway Interface.

CIDR See Classless Inter-Domain Routing.

Classless Inter-Domain Routing (CIDR) A routing system that allows IP addresses to be allocated and utilized more efficiently than they were by the original IP address class system.

CNAME (canonical name) Provides an alias for a host in DNS.

command An order sent to the operating system or an application to perform a specific function.

command line An interface between the user and an operating system or application, which requires commands to be typed at a command prompt in order for them to be processed.

Common Gateway Interface (CGI) An interface used by Web servers to send data between the user and an application program.

compile The process of converting source code into machine instructions used by applications and operating systems.

compression Reducing the size of data so that it uses less space. This can also allow for a reduction in transmission time.

core dump An event that occurs when the data stored in RAM is written to the hard disk. This often occurs for debugging purposes.

CPU (central processing unit) The central processor used by a computer system.

cron A daemon that executes commands at a repeating time and date.

crontab A command used to create a list of commands, which are to be executed on the system at a specified time.

csh (C Shell) A UNIX shell created as an alternative to the Bourne shell at the University of California at Berkeley.

daemon A process that always runs, waiting for requests, which it must receive and transfer to the correct application.

DNS See domain name system.

domain A domain name is used to identify a computer on a network. Several different levels of domains exist to help locate the computer. The top-level domain provides geographic or organization type. The second-level domain specifies the organization name. The third-level domain specifies a host within the organization.

domain name system (DNS) The system used to provide domain-name-to-IP-address mapping.

dpkg The main package management tool for Debian distributions.

DVD A disk similar to a compact disc but capable of storing 4.7GB on each of its two sides. Each side can contain two layers for a maximum storage capacity of 17GB.

EISA (Extended Industry Standard Architecture) The bus architecture that extends the ISA architecture to a 32-bit architecture.

emacs A UNIX-based text editor.

Enhanced System Device Interface See ESDI.

environment variable A variable setting used to customize the UNIX operating system.

ESDI (Enhanced System Device Interface) A hard disk drive interface that predates the Enhanced Integrated Drive Electronics interface.

Ethernet A baseband network cable and access protocol scheme used for local area networks. Ethernet networks use either coaxial or twisted-pair cable in collision detection or collision avoidance access schemes.

ext2 The file system used as the native file system by the Linux operating system.

Extended Industry Standard Architecture See EISA.

FAQ (Frequently Asked Questions) A document designed to provide answers to common questions on a specific topic.

fdisk A utility used to create, delete, and view partition tables on hard disk drives.

file system A procedure that defines the way files are stored and retrieved from the system. This includes naming schemes and the directory structure used for files.

File Transfer Protocol See FTP.

filter A program that receives data and examines it to determine whether any alteration should be done before passing it on to another program.

firewall A piece of software, hardware, or both that is used to protect a network from unwanted access.

Frequently Asked Questions See FAQ.

fsck A file system checker for Linux systems that can verify and repair various file systems.

FTP (File Transfer Protocol) A protocol that is used to allow files to be transferred to and from remote systems.

General Public License See GPL.

General Purpose Mouse See GPM.

GNU A UNIX-type operating system that allows source code to be copied, modified, and distributed freely. The acronym stands for “GNU is Not UNIX.”

GNU is Not UNIX See GNU.

GNU Privacy Guard See GPG.

GPG (GNU Privacy Guard) A free replacement to PGP that allows encryption and data verification.

GPL (General Public License) The free software license under which Linux is released.

GPM (General Purpose Mouse) A tool for text-mode Linux that provides copy and paste support.

Graphical User Interface See GUI.

Group User ID See `guid`.

GUI (Graphical User Interface) An interface used to display data and applications in a graphical format.

guid (Group User ID) A numeric ID assigned to groups on Linux systems.

halt The command used to shut down a Linux system.

hard link A pointer to a file that acts like the original file. One copy of the file exists on the system with the hard link pointer located in another location. Hard links must exist on the same file system as the file they point to.

hexadecimal A base-16 number system used to express bytes of data.

High Performance File System See HPFS.

HOWTO A document that contains step-by-step instructions on completing a task. HOWTOs exist to explain many different tasks.

HPFS (High Performance File System) The file system used by the OS/2 operating system.

HTML (Hypertext Markup Language) The code used to display World Wide Web pages.

HTTP (Hypertext Transfer Protocol) The protocol used in the TCP/IP suite to communicate with Web servers.

Hypertext Markup Language See HTML.

Hypertext Transfer Protocol See HTTP.

ICMP (Internet Control Message Protocol) The protocol in the TCP/IP suite used for message control and error-reporting.

IDE (Integrated Drive Electronics) An interface with a 16-bit bus that allows storage devices to interact with the system.

in-addr.arpa The top-level domain for reverse DNS entries.

Industry Standard Architecture See ISA.

inetd The Internet super server that manages smaller daemons that do not need to be constantly loaded.

init The first process the kernel starts upon booting. It always has a process ID of 1.

inode A descriptor for each file located on a system. The inodes are created when the partition is created, thus setting the number of files that can be stored on the partition.

Integrated Drive Electronics See IDE.

Integrated Services Digital Network See ISDN.

Internet Control Message Protocol See ICMP.

Internet Protocol See IP.

IP (Internet Protocol) The protocol used for transmitting data between computers on a network using the TCP/IP suite.

IP address A unique address assigned to each host on the network for identification purposes.

IP masquerading The process of sharing a single IP address between multiple computers on the network. This process is used to let a network share a single Internet connection.

ipchains The firewall and port blocking system that the Linux kernel v2.2 uses.

ISA (Industry Standard Architecture) The bus interface used by AT system boards.

ISDN (Integrated Services Digital Network) A standard that allows digital transmission over analog lines. These lines are capable of transmitting both voice and data.

ISO 9660 A standard file system used by CD-ROMs regardless of the computer platform.

Java A programming language designed to allow for portability among programs. Java programs can be run locally or distributed throughout the network using virtual machines.

Kerberos A security system that uses “tickets” and “keys” to provide for encrypted authentication across the network without transmitting the user’s password.

kernel The core of the Linux operating system. The kernel is accessed by other parts of the operating system and applications to provide functionality of the system.

LILO (Linux LOader) The most popular boot manager used with Linux. It is capable of booting many other operating systems in addition to Linux.

Line Printer Daemon See lpd.

Linux LOader See LILO.

lpd (Line Printer Daemon) The daemon that receives print jobs from the user and processes them through the print filter before sending them to the print device.

lpr The client tool used to manually print a file.

ls The command used to provide a listing of directory contents on Linux and UNIX systems.

Mail Delivery Agent See MDA.

mailing list A mailing distribution system. Users subscribe to the mailing list and receive all e-mail sent to the list. Mailing lists are a useful way of sharing information about topics of interest.

Makefile A file that provides instructions for a program that is being compiled on the system. It provides information on files that are to be compiled and linked.

man pages The system used by Linux to provide information about commands and utilities on the system. Short for “manual pages.”

MCA (Micro Channel Architecture) A bus architecture used as an interface between adapter cards and the system board. Designed by IBM to overcome the problems associated with ISA architecture, this proprietary architecture is not widely used.

MDA (Mail Delivery Agent) The service responsible for getting e-mail from the mail MTA to the user’s mail file.

message transfer agent See MTA.

metacharacter A character with special meaning, typically used at the command line.

Micro Channel Architecture See MCA.

Million Instructions Per Second See MIPS.

MIME (Multipurpose Internet Mail Extensions) The extensions that allow various types of files to be sent using e-mail. Headers are used to specify the type of file being transmitted.

Minix An older operating system that is used in educational institutions. Linux was built to get around some limitations of Minix.

MIPS (Million Instructions Per Second) A number that provides a measure of computing performance.

mkfs The make file system command, used to create file systems on Linux computers.

module An addition to the system that can be loaded and unloaded interactively. In Linux, modules can be used for software and hardware devices that are not configured as part of the system kernel.

mount The command used to make a file system or device accessible to the system.

MTA (message transfer agent) A program used to transfer e-mail messages from sender to recipient.

Multipurpose Internet Mail Extensions See MIME.

MX record The record used to specify e-mail servers using the domain name system.

NAT (Network Address Translation) A procedure used to translate inside IP addresses to and from outside addresses. This is useful for protecting the internal network from external users.

netmask A procedure used to separate the host address from the network address in an IP address.

Network Address Translation See NAT.

Network File System See NFS.

newsgroup A discussion group available on the Internet via Usenet using the Network News Transfer Protocol. These boards are available worldwide and arranged in a specific hierarchy of categories, topics, and subtopics.

NFS (Network File System) A system that allows a remote file system to appear as a local file system.

NMB The Samba service that is used to map IP addresses to NetBIOS names.

NS record The name server record used to specify additional DNS servers.

octal A base-8 number system used to represent one byte of data.

package A way of distributing and managing software used by some Linux systems. Some common packaging systems include the Red Hat rpm system and the Debian deb system.

PAM (Pluggable Authentication Module) A system of libraries used to handle authentication tasks for services.

partition A logical division on a physical hard disk that separates the disk into separate segments of space.

path The location to search when running commands. A path can be specified along with the command; this is known as the absolute path. The *relative path* is the path in relation to the current location on the file system.

PCI (Peripheral Component Interface) The interface used between the system board and adapter cards. PCI can operate at either 32 or 64-bits for transmissions.

Peripheral Component Interface See PCI.

PGP (Pretty Good Privacy) A program used for encryption with e-mail messages.

pipe A technique used to pass data from one application to another. The output from the first application is sent to the following application where it is further processed.

Pluggable Authentication Module See PAM.

Point-to-Point Protocol See PPP.

port A channel used with the TCP/IP protocol suite to specify the server program will manage the data being transmitted.

PostScript A file format that specifies the elements and layout of a document, often used when printing documents.

PowerPC A CPU architecture developed by Apple, IBM, and Motorola as an open standard.

PPP (Point-to-Point Protocol) The protocol in the TCP/IP suite that provides communication between two computers using serial interfaces. This protocol is most frequently used to provide communications using modems.

present working directory See `pwd`.

Pretty Good Privacy See PGP.

protocol A set or rules that define the communications between devices and hosts on a network.

pwd (present working directory) The user's current location within the file system.

RAM (random access memory) Memory used to store the operating system and data that needs to be accessed quickly. Accessing this type of memory is faster than accessing the hard disk but RAM is more expensive and is available only when the computer is running.

random access memory See RAM.

Red Hat Package Manager See RPM.

redirect A technique used to send application output data to a specified file instead of standard output.

regular expressions Expressions that use normal and special characters for searching and working with files and data.

Reiser A journaling file system being developed to provide a higher degree of fault tolerance on Linux systems.

root The user that has full access to the system including all files, permissions, and user accounts. Also called the *superuser*.

RPM (Red Hat Package Manager) A system used in Red Hat Linux to provide ease of installation and tracking of applications on Linux systems.

runlevels A feature used to specify which services are stopped or started automatically. Some runlevels used include single user, network access, and system halt.

Samba A popular suite of protocols that allow Linux computers to share resources with Windows-based PCs.

SCSI (Small Computer System Interface) A standard interface used for connecting a range of devices to a computer system. Each device is assigned a unique ID number and attached to a cable in a daisy chain configuration.

Secure Shell See SSH.

sendmail A popular program for UNIX and Linux systems that provides SMTP services.

Serial Line Internet Protocol See SLIP.

Server Message Block See SMB.

shadow passwords A system that uses the `/etc/shadow` file to store passwords in encrypted format. The file is accessible only by the root user.

shell A program that runs as a command-line interface between the user and the operating system.

Simple Mail Transport Protocol See SMTP.

SLIP (Serial Line Internet Protocol) A protocol that provides communications over a serial interface using the TCP/IP suite.

Small Computer System Interface See SCSI.

SMB (Server Message Block) The communications protocol used by Windows computers to share files and request services.

SMTP (Simple Mail Transport Protocol) A protocol that handles the transfer of e-mail between servers.

SPARC A RISC-based processor developed by Sun Microsystems.

SSH (Secure Shell) A shell that allows for secure terminal sessions to remote systems.

sticky bit A special bit used with directories to provide greater control of permissions.

SUID The SUID file setting allows programs to be run as a specified user.

superblock An area on the file system that contains crucial information about the inodes and file system configuration.

swap space A designated space on the system used when programs require more memory than the system has available. The hard disk is used to store this data instead.

symbolic link A link that points to a file from another location on the system. These links can exist for files that don't actually exist and can span file systems.

TCP (Transmission Control Protocol) A protocol that works with the IP protocol to transmit data packets across a network.

Telnet A program that allows access to terminal sessions on remote systems.

Transmission Control Protocol See TCP.

UDP (User Datagram Protocol) A connectionless protocol in the TCP/IP suite that transports datagrams across the network.

ulimit A program that is used to limit the size of core dump files that are created.

umask A utility that is used to set the default permissions for files created on the system.

User Datagram Protocol See UDP.

vfat A 32-bit file system used by Windows computers.

vi A popular text editor used on Linux and UNIX systems.

Windows Internet Naming Service See WINS.

WINS (Windows Internet Naming Service) A service that provides dynamic mapping of NetBIOS names to IP addresses. This is used with NT servers to provide name-to-IP-address mappings automatically.

x86 Designation for CPUs manufactured by Intel, and processors with similar architectures. These include the 386, 486, and Pentium I, II, III, and IV.

zone A portion of the DNS hierarchy that a single name server is responsible for serving. A zone may contain a single domain, multiple domains, or only a portion of a domain.

Index

SYMBOLS AND NUMBERS

(pound sign), 389, 414, 542
\$ address, 186
\$ (dollar sign), 188
\$ key, 518
\$(date) setting, 77
% key, 518
& command, 530–531
* (asterisk), 188, 414
+ key, 130, 517
+ option, 372
+ (plus) sign, 169
0 code, 214
0 priority level, 82
1 code, 214
1 console, Red Hat installation, 28
-1 FIELD option, 178
1 signal level, 80
2> symbol, 164
2 code, 214
2 console, Red Hat installation, 28
-2 FIELD option, 178
3 console, Red Hat installation, 29
4 code, 214
4 console, Red Hat installation, 29
5 console, Red Hat installation, 29
5 package verification code, 116
7 console, Red Hat installation, 29
8 code, 214
9 signal level, 80
15 signal level, 80
16 code, 214
128 code, 214
:% command, 527
:1,5 command, 527
: (colon), 74, 399
: key, 130
:.,\$ command, 527
;(semicolon), 69, 526
< (less than) symbol, 161
= (equal) sign, 414
= key, 130
>& symbol, 163
[], 189
>> symbol, 162

> (greater than) symbol, 162
[!] -f option, 723
! address, 186
, (comma), 414
.. (double periods), 791
! (exclamation point), 374, 414
. (period), 74–75, 116, 188, 791
! value, 586
-#<copies> option, 463
-#<num> option, 462
-1 signal, 412
-2 option, 714
-9 signal, 412
-15 signal, 412
-[1234] option, 462
? command, 454
- (dash), 68, 415
- key, 130, 517, 518
- option, 372, 461
/<pattern> key, 130
./, 74
/ (slash), 68
\ command, 69–70
\ metacharacter, 188
\\# setting, 77
\\\$ setting, 77
\\ key, 130
\\] setting, 77
\\! setting, 77
\\[setting, 77
\\< \\>, 189
\\ (backslash) command, 69–70
\\ (backslash) metacharacter, 188
\\[\\], 189
\\ (double backslashes), 77
[^], 189
^ key, 518
|& symbol, 165
| (pipes), 113, 164–165, 799

A

a2ps tool, 464
“a5dd command, 523
a command, 23, 209, 520
A command, 520

- A function, 430
- a option
 - cp command, 242
 - cpio utility, 432
 - depmod command, 489
 - df utility, 216
 - du utility, 216
 - lpc command, 456
 - lpq command, 457
 - lprm command, 461
 - ls command, 236
 - modinfo command, 486
 - modprobe command, 486
 - mount command, 218
 - netstat utility, 584
 - nl utility, 184
 - pr tool, 182
 - rmmod command, 484
 - ssh tool, 713
 - tcpdchk command, 700
 - touch command, 241
 - ulimit command, 733
- A option
 - depmod command, 489
 - fsck utility, 213
 - ipchains tool, 721
 - lpr command, 463
 - lprm command, 461
 - ls command, 236
 - ssh tool, 713
- A RADIX option, 180
- A record, 665, 791
- a switch, 78
- abort command, 454
- About to Install screen, Red Hat 7.0, 39
- absolute paths, specifying, 75
- Accelerated Graphics Port (AGP),
 - described, 791
- access control directives, 630
- access control list (ACL), described, 791
- Access menu option, 128–129
- access methods
 - apt, 129
 - cdrom, 129
 - dselect tool, 129
 - floppy, 129
 - ftp, 129
 - harddisk, 129
 - mounted, 129
 - nfs, 129
- AccessFileName directive, 629
- accessing
 - at command, 422
 - man pages, configuring, 284
 - memory without going through CPU, 18
 - partitions in UNIX, 25
 - printers, 449
- Account Configuration screen, Red Hat 7.0, 36
- account module, 730
- accounts
 - bin, 389–390
 - configuring, 36
 - dummy, 393
 - nobody, 389–390
 - root, 36, 389–390
 - superuser. *See* root account
 - user. *See* users
- ACL. *See* access control list
- acl statement, 658
- Activate on boot option, 34
- active command, 454
- Adaptec, 13
- Address Resolution Protocol (ARP), 570, 791
- addresses
 - \$, 186
 - !, 186
 - configuration, viewing, 18–20
 - hardware, configuring, 18
 - Internet Protocol (IP), 796
 - number, 186
 - Transmission Control Protocol/Internet Protocol (TCP/IP), 564–565
- adduser command, 390–391
- administration
 - security. *See* security administration
 - system. *See* system administration
- AFSTokenPassing entry, 709
- AfterStep window manager, 334
- agents
 - Mail Delivery (MDA), 797
 - message transfer (MTA), 798
 - tkrat, 371
- AGP. *See* Accelerated Graphics Port
- algorithms, MD5, 108
- Alias directive, 630
- alias option, 488
- aliases
 - assigning to printers, 473
 - described, 791
 - mail, 623–624
- alien tool, 137–138, 151–152
- aliens, 791

- all option, 484, 486, 489
- allexport option, 539
- allow files, 423
- AllowGroups entry, 709
- AllowMouseOpenFail entry, 343
- AllowNonLocalModInDev entry, 343
- AllowNonLocalXvidtune entry, 343
- AllowTcpForwarding entry, 709
- AllowUsers entry, 709
- Alpha, 791
- ALSA project, 17
- Alt key, 343
- Alt-1 console, Debian installation, 42
- Alt-2 console, Debian installation, 42
- Alt-3 console, Debian installation, 42
- Alt-4 console, Debian installation, 42
- Amateur radio support menu, 508
- American National Standards Institute (ANSI), 792
- anonymous FTP, 573, 607
- ANSI. *See* American National Standards Institute
- answers to assessment questions
 - boot process, 328–329
 - documentation, 293
 - installing Linux, 61–62
 - installing software, 153–155
 - kernels, 510–511
 - managing files, 272–273
 - managing networks, 692–693
 - managing security, 745–746
 - managing users and groups, 406–407
 - networking, 596–597
 - partitions and file systems, 228–229
 - printing, 474–475
 - processing text, 197–198
 - scripts, 555–556
 - shells, 90–92, 555–556
 - system administration, 441–442
 - X Window System, 382–383
- answers to pre-tests
 - boot process, 328
 - documentation, 292–293
 - installing Linux, 61
 - installing software, 153
 - kernels, 510
 - managing files, 271
 - managing networks, 691–692
 - managing security, 744
 - managing users and groups, 406
 - networking, 595–596
 - partitions and file systems, 228
 - printing, 474
 - processing text, 197
 - scripts, 554–555
 - shells, 90, 554–555
 - system administration, 440–441
 - X Window System, 382
- answers to scenarios
 - boot process, 240
 - documentation, 293
 - installing Linux, 62
 - installing software, 155
 - kernels, 511
 - managing files, 272–273
 - managing networks, 693
 - managing security, 746
 - managing users and groups, 407
 - networking, 597
 - partitions and file systems, 220
 - printing, 475
 - processing text, 199
 - scripts, 556–557
 - shells, 92, 556–557
 - system administration, 442
 - X Window System, 383–384
- answers to study guides
 - documentation, 292–293
 - installing Linux, 61–62
 - installing software, 153–155
 - kernels, 510–511
 - managing users and groups, 406–407
 - networking, 595–597
 - printing, 474–475
 - processing text, 197–199
 - scripts, 554–557
 - shells, 90–92, 554–557
 - system administration, 440–442
 - X Window System, 382–384
- “ap command, 523
- Apache servers
 - configuring, 627–636, 688–691
 - described, 625–626, 791
- starting and stopping httpd daemon, 626–627
- Apache Software Foundation, 791
- apachectl command, 626
- append option, 721
- appending test, 519
- applications
 - bad, managing in X Window System, 370–371
 - customizing, X Window System, 369–370
 - remote X, displaying across networks, 380–381
 - used by Transmission Control Protocol/Internet Protocol (TCP/IP), 573–579

- apropos command, 282–283
 - APS TokenPassing entry, 709
 - Apsfilter, 450–451
 - apt access method, 129
 - apt-get command, 148, 151–152
 - apt-get tool, 132–136
 - architecture
 - ARCnet, 791
 - Extended Industry Standard (EISA), 794
 - Industry Standard (ISA), 796
 - Linux, 9–10
 - Micro Channel (MCA), 797
 - X Window System, 333–334
 - archives, package, clearing, 136
 - ARCnet, 791
 - arguments
 - described, 68, 792
 - LogFormat directive, 636
 - passing to other commands, 166–167
 - arguments field, 730
 - arguments. *See also* options
 - arithmetic tests, 548
 - ARP. *See* Address Resolution Protocol
 - arrow keys, 72, 130
 - as limit, 732
 - ASCII, 791
 - ascii command, 574
 - Asente, Paul, 333
 - ash shell, 66
 - assessment questions
 - boot process, 323–326
 - documentation, 290–293
 - installing Linux, 59–62
 - installing software, 141–144, 153–155
 - kernels, 502–505, 510–511
 - managing files, 268–271
 - managing network services, 679–682
 - managing security, 739–742, 745–746
 - managing users and groups, 403–407
 - networking, 592–597
 - partitions and file systems, 222–225
 - printing, 469–472, 474–475
 - processing text, 192–195, 197–198
 - scripts, 551–556
 - shells, 85–92, 551–556
 - system administration, 436–439, 441–442
 - X Window System, 377–380, 382–383
 - assigning, aliases to printers, 473
 - asterisk (*), 188, 414
 - AT Attachment Packet Interface (ATAPI),
 - described, 792
 - at utility, 421–423
 - ATAPI. *See* AT Attachment Packet Interface
 - atarm terminal emulator, 369
 - atime option, 249
 - auth facility, 414
 - auth module, 730
 - authentication
 - Apache servers, 632–633
 - controlling, Pluggable Authentication Modules (PAM), 728–730
 - DNS servers, 663
 - public key, 715–718
 - Secure Shell (SSH), 705–712
 - author option, 486
 - authpriv facility, 414
 - autoclean option, 483, 486
 - Automatic Partitioning screen, Red Hat 7.0,
 - 33–34
 - available file, 126
 - “ayy command, 523
 - “Ayy command, 523
- ## B
- b <n> option, 712, 716
 - b BYTES option, 179
 - b command, 23
 - b key, 518
 - b option
 - at utility, 422
 - du utility, 216
 - file command, 239
 - ipchains tool, 723
 - tac utility, 182
 - tar utility, 430
 - whereis command, 251
 - B option, 236, 251, 715
 - b SUPERBLOCK option, 213
 - ba option, 118
 - background, processes running in, 81
 - background <color> option, 368
 - backing up, data before partitioning disks, 33
 - backups, 425–434, 440
 - bad block checks, 44–45
 - bad-path option, 120
 - banners, print, disabling, 473
 - base systems, 42–43, 49–51
 - bash2 shell, 66
 - bash (Bourne-Again Shell), 65–66, 792
 - .bashrc file, 65, 537–538
 - .bash_logout file, 538
 - Bastille Linux, 737
 - batch utility, 423
 - BaudRate <baud rate> entry, 344
 - bd <color> option, 368
 - Bell Labs, 5

- benchmarks, example of, 9
 - Berkeley Internet Name Domain (BIND)
 - described, 792
 - version 4, 668–669
 - version 8, 654–667
 - bg <color> option, 368
 - bg jobnumber command, 81
 - bidirectional option, 722
 - bin account, 389–390
 - binary, 792
 - binary packages, 95, 116–118, 337–338
 - BIND. *See* Berkeley Internet Name Domain
 - BIOS, 13–14
 - BIOS settings, installing Linux, 14
 - BIOSBase <Base Address> entry, 347
 - bits, 258, 792, 801
 - Blackbox window manager, 334
 - BlankTime <Time> entry, 348
 - blob=NAME option, 484
 - Block devices menu, 508
 - block directory, 482
 - blocking unwanted connections, IP chains, 721–728
 - blocks, bad, 44–45
 - BoardName “Model” entry, 347
 - boot, 792
 - boot directory/boot directory, 22
 - boot disks, creating, 29, 41, 320
 - boot managers, 28
 - boot services, 326–328
 - boot.img, 29
 - booting
 - customizing process of, 311–316
 - Debian for Linux installations, 41–42
 - init process, 305–311
 - kernels, 509
 - runlevels, 305–311
 - SCSI controllers from SCSI IDs, 14
 - troubleshooting, 317–321
 - bootnet.img, 29
 - bordercolor <color> option, 368
 - borderwidth <number> option, 368
 - Bourne-Again Shell. *See* bash
 - braceexpand option, 539
 - branches, development and stable, 480
 - Broadcast option, 34
 - browsers, Netscape Navigator, 371
 - bs=BYTES option, 244
 - BSD startup, 311
 - bsh shell, 66
 - buffers, clearing, 419
 - Build section, spec files, 117
 - Buttons N entry, 344
 - bw <number> option, 368
 - bye command, 574
 - byte, 792
 - bzip2 option, 498
 - bzip compression, 431
 - bzip utility, 262–263
- ## C
- c\$ command, 520
 - “c3P command, 523
 - c=filename option, 136
 - C <class> option, 462–463
 - C <comment> option, 716
 - c <limit> option, 733
 - C BYTES option, 179
 - c command, 23
 - C command, 520
 - c comment option, 394–395
 - C config-file option, 280
 - c function, 430
 - c job option, 422
 - c NUMBER option, 176–177, 576
 - c option
 - clearing buffers, 419
 - cut utility, 170
 - du utility, 216
 - fmt utility, 174
 - fsck utility, 213
 - gzip utility, 433
 - locate command, 250
 - lpr command, 462
 - ls command, 237
 - modprobe command, 486
 - scp tool, 715
 - sort utility, 168
 - ssh tool, 713
 - ssh-agent tool, 717
 - ssh-keygen tool, 716
 - touch command, 241
 - wc utility, 183
 - C option, 430, 487, 586, 714–715, 722
 - C Shell. *See* csh
 - C value, 586
 - Caldera, 11
 - canonical name (CNAME), 793
 - carat (^), 188
 - Card tab, 352
 - cards
 - network. *See* network cards
 - sound. *See* sound cards
 - video. *See* video cards
 - case statement, 546–547
 - cat utility, 162–167, 195–196

- cbs=BYTES option, 244
- CD
 - HOWTO documentation, 749–752
 - Hungry Minds test engine, 748–749
 - installing from Windows, 747
 - system requirements, 747
 - troubleshooting, 752
- cd command, 234–235, 574
- CD-ROM drives, 15, 29–32, 428, 792
- cd-rom protocol, 133
- CD-RW, described, 792
- cdable_vars option, 541
- CDPATH variable, 535
- cdrom access method, 129
- cdrom device, 205
- cdrom directory, 482
- cdspell option, 541
- central processing unit (CPU), 793, 802
- cdisk tool, 26–27, 44
- CGI. *See* Common Gateway Interface
- chains, IP, 721–728, 743–743
- Changelog section, spec files, 117
- changes, verifying in files, 115–116
- changing
 - characters, 175
 - commands, Readline Library, 70–71
 - compile process, spec files, 118
 - configuration files manually, 496–497
 - consoles, Red Hat installations, 28
 - default prompts, 89
 - default search path, man pages, 284
 - directories, 233–235
 - etc/profile/etc/profile, 75
 - etc/services file/etc/services file, 195–196
 - files, 516
 - groups, 397
 - Makefiles, 100–101
 - partition types, 24
 - passwords, 398
 - PATH variable, 73–75
 - priority levels, processes, 82
 - shells, 65
 - sources.list, 133–134
 - text, 519, 527
 - time stamps, 240–241
 - user accounts, 395–396
- channels, direct memory access (DMA), 18
- Character devices menu, 508
- characters
 - deleting, 520
 - matching, 524–525
 - substituting, 175
- chat scripts, 588–589
- checkhash option, 541
- checking
 - available packages, 126–127
 - bad blocks, 44–45
 - boot services, 326–328
 - dependencies, 37–38, 122
 - file ownership, 145
 - file systems, 212–217
 - package file changes, 115–116
 - package integrity, 107–108
 - packages that installed files, 113, 126
 - paths, 74–75
 - print queue status, 458–459
 - security on systems, 734–737
 - source code, kernels, 493
 - status of printers, 473
- CheckMail entry, 709
- checkoption, 722
- checksig option, 108
- chgrp command, 255–256
- child process, 73
- Chipset “Chipset Type” entry, 347
- chipsets, support in XFree v4, 339
- chmod command, 256–257, 405
- choosers, configuring XDM to provide, 374
- choosing
 - absolute paths, 75
 - desktop environments, 367
 - group ids, 392
 - languages, Red Hat 7.0, 30–31
 - monitors, 39
 - packages to install, 52–54
 - time zones, 14, 35–36
 - user ids (uid), 392
 - video cards, 39
 - window managers, 367
- ChordMiddle entry, 345
- chown command, 254–255, 405
- chsh command, 66
- CIDR. *See* Classless Inter-Domain Routing, 792
- Ciphers entry, 709
- class command, 454
- class field, 663
- classes, query, 673
- classes of networks, 565–566
- Classless Inter-Domain Routing (CIDR), 792
- Clean section, spec files, 117
- ClearDTR entry, 345
- clearing
 - buffers, 419
 - package archives, apt-get tool, 136
- ClearRTS entry, 345
- CLF. *See* Common Log Format

- client command, 454
- client entry, 374
- clients
 - connections, 647–650
 - DNS, configuring, 669–671
 - running remotely, 371–374
 - X, 368
- ClockChip “ClockChip Type” entry, 347
- clockchip setting, 56
- clocks, configuring, 14, 35
- Clocks <clock> ... entry, 347
- closing
 - dselect tool, 132
 - selection screen, 131
 - vi text editor, 516–517
- cmdhist option, 541
- CNAME record, 666
- CNAME. *See* canonical name
- Code maturity level options menu, 507
- colon (:), 74, 399
- color depths, 353–354
- COLUMN option, 182
- columns
 - package lists, 123
 - syslog.conf, 414–415
- combining
 - lines from multiple files, 177–178
 - options, 68
- comma (,), 414
- command completion, 70
- command lines, 67–72, 793
- command substitution, 71
- commands. *See specific commands*
- comment lines, 147, 542
- Common Gateway Interface (CGI), 631, 793
- Common Log Format (CLF), 635
- compile, 793
- compiling
 - kernels, 497–499, 508–509
 - modules, 498, 509
 - software, 101–102
 - spec files, 118
- completion, command, 70
- components
 - contrib, 133
 - main, 133
 - non-free, 133
- compress command, 418
- compress utility, 261–262, 433
- compression, 259–263, 431, 793
- Computer Oracle and Password System (COPS), 736
- conditions, error, 124
- config configfile option, 487
- Config menu option, 132
- config-files value, 124
- configuration addresses, viewing, 18–20
- configuration files
 - creating, 20
 - editing manually, 496–497
 - FTP servers, 608–615
 - networks, managing, 586–588
 - removing, 121
 - saving, 57, 727–728
 - sendmail, 618
- configuration scripts, running for new packages, 132
- configuration tools, X Window System, 349–356
- configure script, 98–100
- Configure using DHCP option, 34
- configure-any option, 120
- configuring
 - access to man pages, 284
 - access to r commands, 703–705
 - accounts, 36
 - addresses of hardware, 18
 - Apache servers, 627–636, 688–691
 - Apsfilter, 450–451
 - base systems, 42–43
 - Berkeley Internet Name Domain (BIND), 654–667
 - caching-only name servers, 667–668
 - clocks, 14, 35
 - cron jobs, 440
 - DNS clients, 669–671
 - DNS servers, 682–685
 - environment variables, 73
 - exports, 638–639
 - fonts, 358–359
 - FTP servers, 605–617
 - global settings, 400–402
 - hardware for installing Linux, 12–18
 - hostnames, 47
 - inetd daemon, 602–604
 - init process, 308–310
 - installers, Red Hat, 29–32
 - IP chains, 743–744
 - kernels, 493–497, 506–508
 - keyboards, Red Hat 7.0, 30–31
 - Linux LOader (LLO), 298–303, 499–500, 509
 - Magicfilter, 451–452
 - maximum print job sizes, 473
 - mice, Red Hat 7.0, 32
 - modems, 16–17
 - monitors, 55

Continued

- configuring (*continued*)
 - network cards, 16, 34–35
 - networks, 47–49, 580–590
 - NFS servers, 685–688
 - plug-and-play hardware, 18–21
 - printers, 445, 473
 - remote clients, 373
 - remote logins, 373–374, 381
 - root passwords, 36
 - Samba, 643–646
 - SCSI and NIC devices, 12
 - Secure Shell (SSH) servers, 708–712
 - security, X Window System, 371–372
 - sendmail.cf, 619–623
 - shell prompts, 66–67
 - sound cards, 17
 - TCP wrappers, 697–701
 - Telnet, 617
 - user settings, 400–402
 - video cards, 55–56, 346–347
 - volumes, 43–46
 - X Window System, 39–40, 54–57
 - xdm package, 362–366
 - XDM for remote logins, 381
 - XDM to provide choosers, 374
 - XF86Config file, 341–348
- conflicts, plug-and-play hardware, 18–21
- conflicts option, 121
- confnew option, 121
- confold option, 121
- connect SCRIPT option, 590
- connections
 - blocking unwanted with IP chains, 721–728
 - clients, 647–650
 - hosts, 372
- Console drivers menu, 508
- consoles
 - Debian installations, 42
 - Red Hat installations, 28–29
- contrib component, 133
- control-flag field, 730
- controller requirements, installing Linux, 13–14
- controllers, SCSI, 13–14
- controls statement, 662
- converting
 - files for printing, 181–182
 - files to PostScript, 464
 - RPM packages to .deb packages, 152–153
 - tabs to spaces, 172–173
- Coordinated Universal Time (UTC), 35
- COPS. *See* Computer Oracle and Password System
- copy backup method, 427
- copy-in mode, 432
- copy-out mode, 432
- copy-pass mode, 432
- copying
 - files, 241–245
 - kernels, 509
 - packages, 40–41
 - standard input to standard output, 165–166
 - text, 521–523
- copyleft, 6
- copyright, 6
- copytruncate command, 418
- core dump, 434, 793
- core limit, 732
- count=BLOCKS option, 244
- cp command, 241–243
- cpio utility, 432
- CPU, accessing memory without going through, 18
- cpu limit, 732
- CPU requirements, installing Linux, 13
- crack tool, 735
- create command, 418
- cron daemon, 793
- cron facility, 414
- cron jobs, setting up, 440
- crond daemon, 423–425
- crontab command, 440, 793
- crontab files, 423–425
- cs command, 520
- csh (C Shell), 66, 793
- ctime option, 249
- Ctrl key, 70–71, 343
- Ctrl-a, 71
- Ctrl-b, 70, 518
- Ctrl-d, 71, 518
- Ctrl-e, 71
- Ctrl-f, 70, 518
- Ctrl-G, 518
- Ctrl-k, 71
- Ctrl-l, 71
- Ctrl-u, 518
- Ctrl-w, 71
- cua device, 205
- current status values, 124
- cursors
 - deleting text to, 521
 - moving in vi text editor, 517–518
- Custom System option, 32
- customizing
 - applications, X Window System, 369–370
 - boot process, 311–316
 - FTP server configuration files, 608–615
 - prompts, 76
- cut utility, 170–171

cutting, text, 170–171
 cw command, 520
 cycling through commands, 72

D

d#\$ command, 521
 d1G command, 521
 -D <debug options>, 456–457, 461
 -d <client> option, 733
 -D <debug options>, 463
 -d <STRING> option, 241
 d command, 23, 208
 D command, 521
 d(^) command, 521
 -d delimit-list option, 171
 -d function, 430
 d /home_dir option, 394–395
 -d option

- at utility, 422
- cp command, 242
- cpio utility, 432
- gzip utility, 433
- ip chains tool, 722
- lpr command, 462
- ls command, 236
- modinfo command, 486
- modprobe command, 486
- ping utility, 576
- pr tool, 182
- rm command, 246
- sort utility, 168
- ssh-add tool, 718
- sshd command, 712
- tcpdchk command, 700
- tr utility, 175

 -D option, 718, 721
 D package verification code, 116
 -d PATH option, 249–250
 \d setting, 77
 D value, 586
 DacSpeed <speed> entry, 347
 daemon facility, 414
 daemons

- cron, 793
- cron, 423–425
- described, 793
- httpd, 626–627
- inetd, 602–605, 796
- Line Printer (lpd), 453, 797
- starting and stopping, 411–412
- syslogd, 413–414
- tcpd, 697–701

 daily command, 418
 dash (-), 68, 415

data

- backing up, 33, 425–434
- sending. *See* processing text

 data limit, 732
 data volumes, configuring, 43–46
 databases

- package, querying, 122–123
- RPM, 105, 107, 112–115

 date fields, 424
 day of month field, 424
 day of week field, 424
 dd command, 243–244, 521
 deb files, clearing from archives, 136
 .deb package, converting RPM package to, 152–153
 Debian

- configuring init process, 315–316
- described, 11–12
- finding information about packages, 150
- installing Linux, 41–58
- installing X Window System, 336
- labs, 147–153
- listing installed files, 148–150
- refreshing available package list, 148
- removing comment lines from source lines, 147

 Debian package management system

- apt-get tool, 132–136
- dpkg tool, 119–127, 793
- dselect tool, 127–132

 debug command, 454
 -debug option, 486
 decompressing, tarballs, 97–98
 default prompts, changing, 89
 DefaultColorDepth <bpp number> entry, 348
 defaultq command, 454
 defaults command, 454
 DefaultType directive, 631
 define macro, 620
 definitions, etc/printcap file, 446–447
 Del command, 70
 Del key, 70–71
 delaycompress command, 418
 -delete function, 430
 -delete option, 721
 deleting

- apt-get tool, 135
- characters, 175
- comment lines from source lines, 147
- dpkg tool, 121–122
- dselect tool, 131–132
- files, 246
- groups, 397

Continued

- deleting (*continued*)
 - hosts, 372
 - modules, 484
 - packages, 112, 150
 - partitions, 26–27
 - PCMCIA packages, 51
 - text, 520–521
 - unused services from hosts, 720–721
 - users, 395
- deny files, 423
- DenyGroups entry, 709
- DenyUsers entry, 709
- dependencies, checking, 37–38, 122
- dependency lists, 497
- depends option, 121
- depends-version option, 121
- depmod command, 489–490, 498
- Depth <bpp> entry, 349
- description option, 138, 486
- descriptors, inode, 796
- desktop environments, X Window System, 335–336, 367
- destination option, 722
- development branch, 480
- Device “Device ID” entry, 348
- Device “Device Path” entry, 344
- Device section, XF86Config file, 345–348
- devices
 - file systems, 205
 - ISA, 20
 - plug-and-play, configuring, 20–21
 - SCSI and NIC, configuring, 12
- df utility, 216–217
- dG command, 521
- DHCP. *See* Dynamic Host Configuration Protocol
- differential backups, 428
- dig tool, 672–674
- Digital Equipment, 333
- direct memory access (DMA) channels, 18, 20
- directives, 628–631, 635–636
- directories
 - block, 482
 - boot/boot, 22
 - cdrom, 482
 - changing, 233–235
 - creating, 246–247
 - exported, mounting, 639–640
 - fs, 482
 - home, creating backups, 440
 - home/home, 22
 - ipv4, 482
 - listing contents of, 235–239
 - misc, 483
 - misc font, 341
 - net, 482
 - present working (pwd), 800
 - representing with HOME variable, 77
 - scsi, 482
 - separating by volume, 21–22, 44
 - shared, creating, 742–743
 - spool, 448–449, 473
 - system, 248
 - troubleshooting printing of, 467
 - /usr/doc/usr/doc, 284–285
 - /usr/src/redhat/usr/src/redhat, 117
 - /var/lib/dpkg/var/lib/dpkg, 126
 - /var/lib/rpm/var/lib/rpm, 107, 113
 - video, 483
- directory block directives, 629–630
- DirectoryIndex directive, 631
- disable command, 454, 459
- DisableModInDev entry, 343
- DisableVidMode entry, 343
- disabling
 - print banners, 473
 - print queues, 459
 - xdm package, 362
- disconnect SCRIPT option, 590
- Disk Druid, creating partitions, 25–26
- disks
 - boot, creating, 29, 41, 320
 - floppy, 50, 428
 - partitioning, 33, 43–46
 - ramdisk, 499
 - repair, 320–321
 - space requirements, installing Linux, 14
- display hostname:display.screen option, 368
- Display subsection, XF86Config file, 349
- DISPLAY variable, 372
- displaying
 - beginning of files, 176
 - configuration addresses, 18–20
 - current settings, 76
 - end of files, 176–177
 - environment variable settings, 73
 - files backwards, 182
 - files in other formats, 180–181
 - members of groups, 397–398
 - messages, 419
 - module information, 485–490
 - numeric details of files, 183
 - package information, 114–115
 - priority levels, 82
 - processes, 78
 - remote X applications across networks, 380–381

- signal levels, 80–81
 - status of packages, 124–125
 - .dist extension, 515
 - distributions
 - apt-get tool, upgrading, 135–136
 - Linux, 10–12
 - divert macro, 620
 - dividing files, 179–180
 - dL command, 521
 - DMA channels. *See* direct memory access channels
 - dmesg command, 418–421
 - dnl macro, 620
 - DNS. *See* domain name system, 793
 - DNS servers
 - adding to networks, 48
 - Berkeley Internet Name Domain (BIND), 654–667, 668–669
 - caching-only name servers, 667–668
 - configuring, 682–685
 - configuring clients, 669–671
 - dig tool, 672–674
 - managing, 675
 - namespaces, 651–654
 - nslookup tool, 671–672
 - overview, 651
 - dnsdomainname command, 581
 - documentation
 - creating, 287
 - HOWTO, 285–286, 749–752, 795
 - on Internet, 284–286
 - man pages, 277–284
 - technical support, 287–288
 - /usr/doc/usr/doc, 284–285
 - dollar sign (\$), 188
 - domain name system (DNS), 793
 - domainname command, 581
 - domains
 - Berkeley Internet Name (BIND), 792
 - described, 793
 - in-addr.arpa, 796
 - root, 652–653
 - top-level (TLD), 652
 - DontZap entry, 342
 - DontZoom entry, 342
 - dosfsck utility, 212
 - dotglob option, 541
 - double backslashes (\), 77
 - double periods (..), 791
 - down arrow key, 72
 - down command, 454
 - down option, 582
 - downloading
 - kernels, 505–506
 - RPM files, 145–146, 150–151
 - dpkg tool, 119–127, 793
 - dport option, 722
 - Driver “Driver Name” entry, 348
 - drivers.img, 29
 - drives
 - CD. *See* CD drives
 - DVD, support of by Linux, 15
 - low level formatting, 14
 - tape, 15, 428
 - Zip, 428
 - dselect tool, 127–132
 - dst option, 722
 - du utility, 215–216
 - dual booting, operating systems, 22
 - dummy accounts, 393
 - dumps, core, 793
 - duplicating
 - files, 241–245
 - kernels, 509
 - packages, 40–41
 - standard input to standard output, 165–166
 - text, 521–523
 - DVD, 515, 793
 - dw command, 521, 523
 - Dynamic Host Configuration Protocol (DHCP), 580–581
- E**
- e2fsck utility, 212
 - e <character> option, 713
 - e command, 24
 - e command option, 186
 - e date option, 394–395
 - e DIR option, 249–250
 - e key, 518
 - E key, 518
 - e option, 112, 424, 489, 586
 - E option, 187
 - E. *See* Enlightenment window manager
 - e-mail
 - Multipurpose Internet Mail Extension (MIME), 798
 - Pretty Good Privacy (PGP), 799
 - echo command, 73–75, 528–529
 - editing
 - characters, 175
 - commands, Readline Library, 70–71
 - compile process, spec files, 118
 - configuration files manually, 496–497
 - default prompts, 89

Continued

- editing (*continued*)
 - default search path, man pages, 284
 - etc/profile/etc/profile, 75
 - etc/services file/etc/services file, 195–196
 - files, 516
 - groups, 397
 - Makefiles, 100–101
 - partition types, 24
 - passwords, 398
 - PATH variable, 73–75
 - priority levels, processes, 82
 - sources.list, 133–134
 - text, 519, 527
 - time stamps, 240–241
 - user accounts, 395–396
- EDITOR environment variable, 401
- editors
 - emacs, 794
 - stream. *See* sed utility
 - vi. *See* vi text editor
- edquota utility, 264–265
- EISA. *See* Extended Industry Standard
 - Architecture
- emacs editor, 794
- emacs option, 539
- Emulate3Buttons entry, 344
- emulation
 - Math Coprocessor, 13
 - three-button, 55
- enable command, 454, 459
- enabling
 - packet forwarding, 727
 - print queues, 459
 - xdm package, 362
- encryption
 - DNS servers, 663
 - public key, 706
- engines, test, 748–749
- Enhanced System Device Interface (ESDI), 794
- Enlightenment (E) window manager, 334
- Enter key, 67, 69, 518
- <ENTER> option, 30, 42
- entering commands on the same line, 69
- entries. *See specific entries*
- environment variables
 - CDPATH, 535
 - described, 72–73, 794
 - EDITOR, 401
 - exporting, 535
 - HISTCMD, 72, 530–531
 - HISTFILESIZE, 72, 531
 - HISTIGNORE, 530–531
 - HISTSIZE, 531
 - HOME, 77
 - MAIL, 531–532
 - MAILCHECK, 531–532
 - MAILPATH, 531–532
 - MANPATH, 279, 284
 - PATH, 73–76, 284, 535
 - prompt, 532–534
 - PS1, 76
 - quoting, 528–530
 - SHELL, 65
- equal (=) sign, 414
- erasing
 - apt-get tool, 135
 - characters, 175
 - comment lines from source lines, 147
 - dpkg tool, 121–122
 - dselect tool, 131–132
 - files, 246
 - groups, 397
 - hosts, 372
 - modules, 484
 - packages, 112, 150
 - partitions, 26–27
 - PCMCIA packages, 51
 - text, 520–521
 - unused services from hosts, 720–721
 - users, 395
- erexit option, 539
- error condition values, dpkg tool, 124
- error streams, text, 161–164
- errors
 - Linux LOader, 317–319
 - redirected files, 163–164
- errors mailto command, 418
- errsyms option, 489
- Esc-b, 71
- Esc-d, 71
- Esc-Del, 71
- Esc-f, 71
- escaping metacharacters, 524–525
- ESDI. *See* Enhanced System Device Interface
- /etc/fstab file, 219–220
- /etc/group/etc/group file, 392, 397–398
- /etc/hostname file, 588
- /etc/HOSTNAME file, 588
- /etc/hosts file, 587, 670
- /etc/hosts.lpd, 449
- /etc/mstab file, 220
- /etc/passwd/etc/passwd file, 65, 391–392, 398
- /etc/printcap file, 446–447
- /etc/profile/etc/profile file, 75, 400–401

- /etc/rc.d scripts, 411–412
- /etc/resolv.conf, 588
- /etc/services file, 587
- /etc/services file/etc/services file, changing, 195–196
- /etc/shadow file, 398–400, 719
- /etc/shells/etc/shells file, 65
- /etc/skel/etc/skel file, configuring user settings, 401–402
- etc/syslog.conf file, 414–416
- Eterm terminal emulator, 369
- etheral package, 110
- Ethernet, 794
- exam objectives
 - documentation, 275
 - installing Linux, 3–4
 - installing software, 93–94
 - kernels, 477
 - managing security, 695
 - managing users and groups, 387
 - mapping to practice exams, 775–786
 - networking, 561–562
 - printing, 443
 - processing text, 159
 - scripts, 513
 - shells, 63, 513
 - system administration, 409
 - X Window System, 331
- exam objectives. *See also* LPIC certification exam
- exclamation point (!), 374, 414
- exit codes, fsck utility, 214
- exit command, 65, 454
- exiting
 - dselect tool, 132
 - selection screen, 131
 - vi text editor, 516–517
- expand utility, 172–173
- expanding. *See* decompressing
- expert <ENTER> option, 30
- export command, 73, 75, 381, 535
- exportfs command, 638–639
- exporting environment variables, 535
- exports, configuring, 638–639
- expressions, regular, 800
- ext2, 794
- Extended Industry Standard Architecture (EISA), 794
- extended partitions, 24, 207
- extensions
 - .dist, 515
 - Multipurpose Internet Mail (MIME), 798

F

- F1 key, 30, 42
- F2 key, 30
- F3 key, 30, 42
- F4 key, 30
- F5 key, 30
- F10 key, 42
- f <FILENAME> option, 239
- F <filter format> option, 463
- f <limit> option, 733
- f <string> option, 486
- f days option, 394–395
- f file option, 186
- f FILE option, 422
- f option
 - apt-get tool, 136
 - cp command, 242
 - cpio utility, 432
 - cut utility, 170
 - fsck utility, 213
 - gzip utility, 433
 - insmod command, 483
 - lpr command, 462
 - mount command, 218
 - mv command, 245
 - ping utility, 576
 - pr tool, 182
 - rm command, 246
 - sort utility, 168
 - ssh tool, 713
 - ssh-keygen tool, 716
 - syslogd daemon, 413
 - tail utility, 177
 - tar utility, 430
 - traceroute utility, 578
- F option, 111, 187, 432, 578, 721
- F Virtual Window Manager. *See* FVWM window manager
- fa command, 527
- Fa command, 527
- facilities, syslog.conf files, 414–415
- failed-config value, 124
- FancyIndexing option, 732
- FAQ. *See* Frequently Asked Questions
- fc utility, 72
- fd device, 205
- fdisk tool, 23–24, 794
- fdisk utility, 208–210
- feature macro, 620
- fg <color> option, 368
- fg jobnumber command, 81

- fields
 - ftpconversions file, 613
 - syslog.conf, 414
 - time and date, 424
 - xferlog file, 614–615
- file command, 239–240
- File List section, spec files, 117
- File protocol, 133
- file settings, SUID, 801
- file systems
 - checking, 212–217
 - creating, 207–211, 225–227
 - described, 794
 - ext2, 794
 - hierarchies of, 247–248
 - High Performance (HPFS), 795
 - mounting and unmounting, 217–220
 - Network (NFS), 392, 799
 - overview, 203–207
 - Reiser, 800
 - superblocks, 801
 - vfat, 802
- File Transfer Protocol (FTP)
 - described, 573–574, 794
 - downloading kernels, 505–506
 - downloading RPM files, 150–151
 - sources.list support for, 133
- filename option, 486
- files. *See also specific files*
 - adding line numbers to, 183–185
 - changing directories, 233–235
 - changing time stamps, 240–241
 - checking ownership of, 145
 - Common Gateway Interface (CGI), 631
 - compression tools, 259–263
 - converting for printing, 181–182
 - converting to PostScript, 464
 - copying, 241
 - copying and moving text between, 522–523
 - deleting, 246
 - determining types of, 239–240
 - displaying backwards, 182
 - displaying in other formats, 180–181
 - dividing, 179–180
 - downloading, 145–146, 150–151
 - finding, 248–251
 - finding packages that installed, 113, 126
 - joining, 177–178
 - linking, 251–252
 - listing in packages, 114, 125
 - managing quotas, 263–266
 - moving, 245
 - opening for editing, 516
 - overwriting, 110
 - permissions, 253–258
 - printing, 461–464, 467
 - saving, 516–517
 - sorting lines of, 167–170
 - verifying changes in, 115–116
 - viewing beginning and end of, 176–177
- Files section, XF86Config file, 341–342
- Filesystems menu, 508
- filtering text
 - adding line numbers to files, 183–185
 - converting files for printing, 181–182
 - converting tabs to spaces, 172–173
 - cut utility, 170–171
 - deleting and substituting characters, 175
 - displaying files backwards, 182
 - displaying files in other formats, 180–181
 - displaying numeric details of files, 183
 - dividing files, 179–180
 - formatting paragraphs, 173–175
 - grep utility, 187–188
 - joining files, 177–178
 - paste utility, 171–172
 - sorting lines of files, 167–170
 - stream editor, 185–187
 - viewing beginning and end of files, 176–177
- filtering tools, 195–196
- filters, 450–452, 794
- find utility, 249
- finding
 - files, 248–251
 - information about packages, 150
 - list of installed files, 148–150
 - man pages, 279–283
 - module files, 482–483
 - packages that installed files, 113, 126
 - text, 524–527
- finger utility, 579
- firewalls, 9, 794
- flags
 - fstab command, 640
 - g, 186
 - l, 186
 - NUMBER, 186
 - p, 186
 - s command, 186
 - testing scripts, 543–544
 - values, 586
 - w filename, 186
- floppy0 rescue option, 42
- floppy1 rescue option, 42

floppy access method, 129
 floppy disks, 50, 428
 flow control, scripts, 545–548
 –flush option, 721
 fmt utility, 166, 173–175
 –fn option, 368
 folders. *See* directories
 –font option, 368
 fonts, 358–359
 fonts.alias file, 359
 fonts.dir file, 359
 for loop, 546
 –force option, 110, 483
 force options, dpkg tool, 120–121
 foreground, processes running in, 81
 –foreground <color> option, 368
 –format <string> option, 486
 formatting
 files, 180–181
 low level, hard drives, 14
 paragraphs, 173–175
 forwarding
 mail, 623–624
 packets, 727
 –fragment option, 722
 free, definition of, 7
 Free Software Foundation, 6–7
 Frequently Asked Questions (FAQ), 749, 794
 –freshen option, 111
 fs directory, 482
 fs-options, 210
 fsck tool, 794
 fsck utility, 212–215
 fsize limit, 732
 fstab command, 640
 ftp access method, 129
 ftp command, 573–574
 FTP. *See* File Transfer Protocol
 FTP servers, configuring, 605–617
 ftpaccess file, 608–612
 ftpconversions file, 613–614
 ftpcount command, 617
 ftpd command, 606
 ftpgroups file, 612–613
 ftphosts file, 612
 ftpshut command, 615–616
 ftpusers file, 612–613
 ftpwho command, 616
 full backups, 427
 functions, tar command, 430
 Future Domain, 13
 FVWM window manager, 34

G

–g <n> option, 712
 g flag, 186
 –g gid option, 396
 –g group option, 394, 396
 –G groups option, 394, 396
 G key, 518
 –g option, 82, 137, 462, 713
 –G option, 187, 237
 G package verification code, 116
 G value, 586
 Gamma <gamma value(s)> entry, 346
 Gateway option, 34
 GatewayPorts entry, 709
 gdb. *See* GNU Debugger
 General Purpose Mouse (GPM), 795
 General setup menu, 507
 –geometry <width>x<height>+<x>+<y> option, 368
 get file command, 574
 global settings, configuring, 400–402
 GMT. *See* Greenwich Mean Time
 GNOME. *See* GNU Network Object Model Environment, 335–336
 gnome-terminal emulator, 369
 GNU Debugger, 434
 GNU General Public License (GPL), 6–7, 795
 GNU is Not UNIX (GNU), 795
 GNU Network Object Model Environment (GNOME), 335–336
 GNU Privacy Guard (GPG), 794
 GNU Project, 6
 GNU. *See* GNU is Not UNIX
 GnuPG package
 importing public keys, 146–147
 retrieving files, 145–146
 validating package integrity, 108
 gpg command, 108
 GPG. *See* GNU Privacy Guard
 GPL. *See* GNU General Public License
 GPM. *See* General Purpose Mouse
 Graphical User Interface (GUI), 795
 greater than (>) symbol, 162
 Greenwich Mean Time (GMT), 75
 grep command, 113
 grep utility, 187–188
 group id, choosing, 392
 –group option, 249
 group permissions, 254–258
 Group User ID (guid), 795
 groupadd command, 396
 groupdel command, 397
 groupmod command, 397

- groups
 - adding, 390–393, 396
 - deleting, 397
 - lp, 448
 - managing, 393–400
 - modifying, 397
 - package, selecting, 37–38
 - viewing members of, 397–398
 - groups command, 397–398
 - GUI. *See* Graphics User Interface
 - guid. *See* Group User ID
 - guides, 286
 - gunzip utility, 97, 260–261, 434
 - gzip compression, 431
 - gzip utility, 97, 260–261, 433
- H**
- h <file> option, 712
 - h -h parameter, 111
 - \h setting, 77
 - h column, 123
 - h HEADER option, 182
 - h key, 518
 - h option
 - df utility, 216
 - du utility, 216
 - gzip utility, 433
 - insmod command, 483
 - ipchains tool, 721
 - lpr command, 462, 463
 - modinfo command, 486
 - mount command, 201
 - sed utility, 186
 - syslogd daemon, 413
 - tar utility, 430
 - umount command, 219
 - H value, 586
 - half-installed value, 124
 - halt command, 795
 - hard disk controller requirements, installing
 - Linux, 13–14
 - hard disks, space requirements, installing
 - Linux, 14
 - hard drives, low level formatting, 14
 - hard link, 795
 - hard links, 252
 - harddisk access method, 129
 - hardware
 - plug-and-play, configuring, 18–21
 - preparing for installation of Linux, 12–18
 - supported by Linux, 9
 - Hardware-HOWTO, 12
 - hashall option, 539
 - hd device, 205
 - head utility, 176
 - Header section, spec files, 117
 - Helix Code, 335
 - help command, 454
 - help option
 - cut utility, 171
 - expand utility, 172
 - head utility, 176
 - insmod command, 483
 - join utility, 178
 - modinfo command, 486
 - od utility, 181
 - sort utility, 168
 - split utility, 179
 - tail utility, 177
 - tr utility, 175
 - wc utility, 183
 - hexadecimal, 795
 - High Performance File System (HPFS), 795
 - histappend option, 541
 - HISTCMD variable, 72, 530–531
 - histexpand option, 539
 - HISTFILESIZE variable, 72, 531
 - HISTIGNORE variable, 530–531
 - history command, 72
 - history files, 71–72
 - history option, 540
 - HISTSIZ variable, 531
 - hold option, 120
 - hold printer command, 454
 - hold value, 124
 - holdall command, 454
 - holdoff TIME option, 590
 - home directories, creating backups, 440
 - home directory/home directory, 22
 - HOME variable, 77, 423
 - HorizSync <horizontal sync range> entry, 346
 - host tool, 674
 - host utility, 575
 - HostKey entry, 709
 - hostname command, 581
 - Hostname option, 34
 - hostnames, configuring, 47
 - hosts, 372, 564, 567, 633–636, 719–729
 - hosts.equiv, 704
 - hour field, 424
 - HOWTO documents, 285–286, 749–752, 795
 - HPFS. *See* High Performance File System
 - ht device, 205
 - HTML. *See* Hypertext Markup Language
 - htpasswd command, 633
 - HTTP. *See* Hypertext Transfer Protocol

- httpd daemon, 626–627
 - Hungry Minds test engine, 748–749
 - Hypertext Markup Language (HTML), 795
 - Hypertext Transfer Protocol (HTTP), 133, 795
- I**
- I2O device support menu, 508
 - i <file> option, 713, 715
 - i <indent columns> option, 463
 - i <inetd_conf> option, 700
 - i column, 123
 - i command, 520
 - I command, 520
 - I flag, 186
 - i NUMBER option, 184
 - i [numcols] option, 462
 - i option
 - alien tool, 137
 - cp command, 242
 - df utility, 217
 - ip chains tool, 723
 - locate command, 250
 - ls command, 237
 - mv command, 245
 - netstat utility, 584
 - ping utility, 576
 - rm command, 246
 - sshd command, 712
 - tar utility, 430
 - tee utility, 165
 - I option, 172, 178, 430, 578, 721
 - IBM, 333
 - IceWM window manager, 335
 - ICMP. *See* Internet Control Message Protocol
 - icmp-type option, 722
 - iconic option, 368
 - IDE. *See* Integrated Drive Electronics
 - Identifier “ID String” entry, 346–347
 - identifiers
 - group, 392
 - Group User (guid), 795
 - Uniform Resource (URI), 133
 - user (uid), 390, 392
 - idle SEC option, 590
 - if=FILE option, 244
 - if printer definition, 446
 - if/else statement, 545–546
 - ifconfig utility, 581–583
 - ifempty command, 458
 - ignoreeof option, 540
 - IgnoreRhosts entry, 709
 - IgnoreUserKnownHosts entry, 709
 - ignoring Enter key, 69
 - images, fine-tuning, 357–358
 - import option, 108
 - importing, public keys, 146–147
 - in-addr.arpa, 796
 - include filename command, 418
 - include macro, 620
 - include statement, 656
 - Individual Package Selection screen, Red Hat 7.0, 37–38
 - Industry Standard Architecture (ISA), 796
 - inetd daemon, 602–605, 796
 - inetd.conf, 720
 - init, 305–311
 - init process, 76, 796
 - initializing, data volumes, 45
 - inode descriptor, 796
 - inodes, 206
 - input streams, text, 161–167
 - .inputrc file, 538–539
 - insert option, 721
 - insmod command, 483–484
 - INSTALL files, 96, 98, 100
 - Install menu option, 131
 - install mode, 124
 - Install section, spec files, 117
 - installed value, 124
 - installers, Red Hat, configuring, 29–32
 - installing
 - alien packages, 151–152
 - Apsfilter, 450–451
 - apt-get tool, 134–135
 - base systems, 42–43, 49–51
 - CD from Windows, 747
 - dpkg tool, 119–120
 - dselect tool, 131–132
 - fonts, 358
 - hard drives, 14
 - kernels, 47, 490–500, 505–509
 - Linux LOader (LILO), 50, 303–304
 - Magicfilter, 451–452
 - modules, 47, 498, 509
 - packages, 40–41, 57–58, 109–111
 - packages from remote sites, 148
 - printers, 445–449
 - software from source code, 95–102
 - X Window System, 336–337
 - installing Linux
 - boot managers, 28
 - configuring plug-and-play hardware, 18–21
 - Debian, 41–58
 - partitioning schemes, 21–28
 - preparing hardware for, 12–18
 - Red Hat, 28–41

Integrated Drive Electronics (IDE), 796
 Integrated Services Digital Network (ISDN), 796
 integrity, packages, validating, 107–108
 in.telnetd, 617
 –interface option, 722
 interfaces

- AT Attachment Packet (ATAPI), 792
- Common Gateway (CGI), 793
- Enhanced System Device (ESDI), 794
- Graphical User (GUI), 795
- Industry Standard (ISA), 796
- Integrated Drive Electronics (IDE), 796
- networks, managing, 580–586
- Peripheral Component (PCI), 799
- Small Computer System (SCSI), 800

 Internet, documentation on, 284–286. *See also* Web sites
 Internet Control Message Protocol (ICMP), 571, 796
 Internet Protocol (IP), 569–570, 796
 Internet Protocol (IP) address, 796
 Internet Protocol (IP) masquerading, 726–727, 796
 Internet Super Server, 602–605
 interrupt requests (IRQs), 18–19
 introduction screen, package lists, 129–130
 IOBase <Base Address> entry, 347
 io_addr ADDRESS option, 582
 IP Address option, 34
 IP address. *See* Internet Protocol address
 IP chains, 721–728, 743–744
 IP masquerading. *See* Internet Protocol masquerading
 –ip option, 584
 IP. *See* Internet Protocol
 ipchains, 796
 ipchains tool, 721–728
 ipv4 directory, 482
 IrDA support menu, 508
 irq ADDRESS option, 582
 IRQs. *See* interrupt requests
 ISA devices, 20
 isapnp command, 21
 ISDN. *See* Integrated Services Digital Network
 ISDN subsystem support menu, 508
 ISO 9660, 797

J

–J <job> option, 462
 –J <job name> option, 463
 –j BYTES option, 180
 j key, 518
 –j option, 723

Java programming language, 797
 <job id> ... all option, 458
 jobs

- cron, setting up, 440
- moving in print queue, 458
- print, 459–461, 473
- scheduling, 421–425
- sending to print queues, 473

 jobs command, 81
 join utility, 177–178
 joining, files, 177–178
 Joy, William, 515
 js device, 205
 –jump option, 722
 jumpers, 18

K

–K <copies> option, 463
 –k <time in seconds> option, 712
 –k file option, 394
 k key, 518
 k option

- df utility, 216
- du utility, 216
- insmod command, 483
- lpr command, 463
- ls command, 238
- modprobe command, 486
- ssh tool, 713
- ssh-agent tool, 717

 –K option, 108, 280, 430
 –kallsyms option, 483
 KDE desktop environment, 335
 KeepAlive entry, 710
 Kerberos, 797
 KerberosAuthentication entry, 710
 KerberosOrLocalPasswd entry, 710
 KerberosTgtPassing entry, 710
 KerberosTicketCleanup entry, 710
 kern facility, 414
 kernel development system, 479–480
 Kernel hacking menu, 508
 kernel space, 9–10
 kernels

- booting, 509
- compiling, 497–499, 508–509
- configuring, 506–508
- copying, 509
- described, 10, 797
- downloading, 505–506
- installing, 47, 490–500, 505–509
- Linux LOader (LILO), 298
- managing modules, 482–490

- reconfiguring, 490–500
 - testing, 500, 509
 - types of, 480–481
 - key combinations, 70
 - key pair, 715–716
 - key statement, 663
 - Keyboard Configuration screen, Red Hat 7.0, 30–31
 - Keyboard section, XF86Config file, 343–344
 - Keyboard tab, 351
 - keyboards, configuring, Red Hat 7.0, 30–31
 - KeyRegenerationInterval option, 710
 - keys
 - +, 130, 517
 - ;, 130
 - =, 130
 - \, 130
 - /`<pattern>`, 130
 - Alt, 343
 - arrow, 72, 130
 - Ctrl, 70–71, 343
 - cursor movement, 518
 - Del, 70–71
 - Enter, 67, 69
 - Esc, 71
 - F1, 30, 42
 - F2, 30
 - F3, 30, 42
 - F4, 30
 - F5, 30
 - F10, 42
 - Meta, 343
 - O, 130
 - public, importing, 146–147
 - q, 79, 130
 - Q, 279
 - spacebar, 279
 - special combinations in X Window System, 370
 - Tab, 70
 - V, 130
 - X, 130
 - kill command, 80–81, 412, 454
 - konsole terminal emulator, 369
 - Korn shells, 65
 - ksymoops option, 484
- L**
- l `<login name>` option, 713
 - l BADBLOCKS-FILE option, 213
 - L BADBLOCKS-FILE option, 213
 - l command, 23
 - L command, 24
 - l hostnames option, 413
 - l key, 518
 - l LINES option, 179
 - l name option, 396
 - l option
 - at utility, 422
 - cp command, 242
 - cron daemon, 424
 - df utility, 217
 - gzip utility, 433
 - ipchains tool, 723
 - lpq command, 457
 - lpr command, 462
 - ls command, 237–238
 - modprobe command, 486
 - mount command, 218
 - ssh keygen tool, 716
 - ssh tool, 713
 - ssh-add tool, 718
 - ssh-keygen tool, 716
 - tar utility, 430
 - wc utility, 183
 - L option
 - insmod command, 483
 - lpq command, 457
 - ls command, 236
 - mount command, 218
 - ssh tool, 714
 - ssh-add tool, 718
 - L package verification code, 116
 - l PAGESLENGTH option, 182
 - l switch, 78
 - labs
 - boot process, 326–328
 - checking ownership of files, 145
 - Debian packages, 147–153
 - GnuPG package, 145–147
 - installing kernels, 505–509
 - listing installed packages, 145
 - managing network services, 682–691
 - managing security, 742–744
 - managing users and groups, 405–406
 - partitions and file systems, 225–227
 - processing text, 195–196
 - setting up printers, 473
 - system administration, 440
 - using shells, 89
 - working with kernels, 505, 509
 - X Window system, 380–381
 - Language Selection screen, Red Hat 7.0, 30–31
 - languages, selecting, Red Hat 7.0, 30–31
 - lastlog command, 415
 - lbs=BYTES option, 244

- lcd command, 574
- less command, 421
- less than (<) symbol, 161
- lf option, 449
- lf printer definition, 446
- libraries
 - ncurses, 495
 - Readline, changing commands, 70–71
 - X Toolkit, 369–370
- licenses, GNU General Public (GPL), 6–7
- LILO. *See* Linux LOader
- line numbers, adding to files, 183–185
- Line Printer Daemon (lpd), 453, 464–465, 797
- line printer (LPR), 445
- lines
 - command, 67–72, 793
 - comment, 147, 542
 - entering commands on, 69
 - mode, 345
 - pattern matching, 524
 - source. *See* source lines
 - text, 519–520
- linking, files, 251–252
- links
 - hard, 795
 - symbolic, 801
- Linux
 - ability to multitask, 8
 - architecture of, 9–10
 - benchmark example, 9
 - distributions of, 10–12
 - GNU General Public License (GPL), 6–7
 - hardware supported by, 9
 - history of, 5–6
 - installing CD from, 748
 - installing. *See* installing Linux
 - as multiuser platform, 8
 - peripherals supported by, 15–17
 - reasons to use, 7–9
 - software available for, 9
 - speed of, 9
 - stability of, 8
- linux dd option, 30
- Linux Documentation Project, 285–286
- Linux LOader (LILO), 28
 - configuring, 298–303, 499–500, 509
 - described, 297, 797
 - installing, 50, 303–304
 - reconfiguring, 509
 - troubleshooting, 317–319
 - updating, 303–304
- linux rescue option, 30, 42
- list option, 66
- list option, 721
- list PATTERN option, 486
- Listen directive, 629
- ListenAddress entry, 710
- listing
 - contents of print queues, 458
 - current environment variables and values, 535
 - direct memory access (DMA) channels, 20
 - directory contents, 235–239
 - files in packages, 114, 125
 - I/O ports, 19–20
 - installed files, 148–150
 - installed packages, 113, 123–124, 145
 - interrupt requests (IRQs), 19
 - modules, 485
 - printers, 456
- lists
 - access control (ACL), 791
 - dependency, 497
 - mailing, 286, 797
 - security, 730–732
- lo printer definition, 446
- Loadable module support menu, 507
- loading, modules, 483–484
- LOC record, 666
- local0–local, 7 facilities, 415
- locate command, 249–250
- Location directive, 630
- lock option, 483
- log files, 416, 418–421, 448–449
- log option, 723
- log rotation, 416–418
- LogFormat directive, 636
- logging directives, 635–636
- logging statement, 659
- login files, 536–537
- login shell, 536
- LoginGraceTime entry, 710
- logins, remote, 373–374, 381
- LogLevel entry, 710
- LOGNAME variable, 423
- logrotate utility, 416–418
- logs, system, 287, 413–421
- loops, for, 546
- low level formatting, hard drives, 14
- lp, 448
- lp device, 205
- lpc command, 453–459
- lpd command, 454
- lpd. *See* Line Printer Daemon

- LPIC certification exam
 - answers to exam, 101, 768–772
 - answers to exam, 102, 772–774
 - objectives for exam, 101, 776–780
 - objectives for exam, 102, 780–786
 - practice exam, 101, 753–761
 - practice exam, 102, 761–768
 - tips on testing process, 787–789
- LPIC certification exam. *See also* study guides
- lpq command, 455, 457–458
- lpr command, 461–463
- lpr facility, 414
- LPR. *See* line printer
- lpr tool, 797
- lprm command, 455, 459–461
- ls command, 74, 235–239, 574, 743, 797
- lsmmod command, 485

- M**
- m4 macros, 620
- m <limit> option, 733
- m <mail to> option, 463
- m command, 23, 209
- m interval option, 414
- M key, 518
- m option
 - at utility, 422
 - df utility, 217
 - du utility, 216
 - insmod command, 483
 - lpr command, 462
 - ls command, 238
 - pr utility, 182
 - sort utility, 168
 - tar utility, 430
 - traceroute utility, 578
 - whereis command, 251
- M option, 168, 251, 430, 722
- M package verification code, 116
- M path option, 280
- m switch, 78
- m system option, 280
- M value, 586
- macros, m4, 620
- Magicfilter, 451–452
- mail, forwarding, 623–624
- mail agents, tkrat, 371
- Mail Delivery Agent (MDA), 797
- mail facility, 415
- mail mailto command, 418
- MAIL variable, 531–532
- MAILCHECK variable, 531–532
- mailing lists, 286, 797
- MAILPATH variable, 531–532
- mailwarn option, 541
- main component, 133
- Main Menu options, X Window System, 507–508
- make bzImage command, 498
- make bzlilo command, 498
- make clean command, 497
- make command, 101–102
- make config command, 494–495
- make install command, 102
- make menuconfig command, 495–496
- make modules command, 498
- make xconfig command, 496, 506
- make zImage command, 498
- make zlilo command, 498
- makebzdisk command, 498
- Makefile, 96, 98–102, 797
- makewhatis command, 282
- man command, 277
- man pages, 277–284, 797
- managers, boot, 28
- managing
 - bad applications, X Window System, 370–371
 - DNS servers, 675
 - network configuration files, 586–588
 - NFS servers, 640–642
 - packages, 701–702
 - printer services, 453–461
 - processes, 77–81
 - quotas, 263–266
 - Samba, 646–647
 - sendmail, 624–625
 - users and groups, 393–400
- Mandrake, 11–12
- MANPATH variable, 279, 284
- map option, 483
- masks, subnet, 566–569
- masquerading, Internet Protocol (IP), 726–727, 796
- masquerading option, 722
- Massachusetts Institute of Technology (MIT), 333
- matching
 - pattern, 524–525
 - TCP wrappers, 698
- Math Coprocessor emulation, 13
- maxlogins limit, 732
- MaxStartups entry, 710
- MCA. *See* Micro Channel Architecture
- MD5 algorithm, validating package integrity, 108
- MDA. *See* Mail Delivery Agent

- media
 - backup, 428–429
 - removable, support of by Linux, 15
- media TYPE option, 582
- MemBase <Base Address> entry, 347
- members of groups, viewing, 397–398
- memlock limit, 732
- memory
 - accessing without going through CPU, 18
 - random access (RAM), 800
 - virtual, 22
- memory requirements, installing Linux, 13
- mem_start ADDRESS option, 582
- menu options, dselect tool, 128–132
- menus
 - Amateur radio support, 508
 - Block devices, 508
 - Character devices, 508
 - Code maturity level options, 507
 - Console drivers, 508
 - Filesystems, 508
 - General setup, 507
 - I2O device support, 508
 - IrDA support, 508
 - ISDN subsystem support, 508
 - Kernel hacking, 508
 - Loadable module support, 507
 - Main, X Window System, 507–508
 - Network device support, 508
 - Networking options, 508
 - Old CD-ROM drives, 508
 - Plug and Play support, 507
 - Processor type and features, 507
 - SCSI support, 508
 - Sound, 508
 - Telephony support, 508
 - USB support, 508
- merging lines from multiple files, 177–178
- message transfer agent (MTA), 798
- messages, displaying, 419
- Meta keys, 343
- metacharacters, 524–525, 798
- metadata, 119
- mget file, 574
- mice
 - configuring, Red Hat 7.0, 32
 - support by Linux, 15
 - three-button emulation, 55
- Micro Channel Architecture (MCA), 797
- midi device, 205
- Million Instructions Per Second (MIPS), 798
- Minix, 6, 798
- MinSpareServers directive, 629
- minute field, 424
- minux file system, 204
- MIPS. *See* Million Instructions Per Second
- misc directory, 483
- misc font directory, 341
- MIT. *See* Massachusetts Institute of Technology
- mkdir command, 247–248, 405
- mkfontdir command, 358–359
- mkfs utility, 210–211, 798
- mkraid utility, 211
- mkswap utility, 211
- mode lines, 345
- Mode “name” entry, 346
- Modeline “name” <mode description> entry, 346
- ModelName “Model” entry, 346
- modem device, 205
- modem option, 590
- modems, configuring, 16–17
- modes
 - copy-in, 432
 - copy-out, 432
 - copy-pass, 432
 - install, 124
 - purse, 124
 - remove, 124
 - selection status, 124
 - single-user, 319
 - unknown, 124
- Modes “Mode Name” ... entry, 349
- Modeselection tab, 353–354
- modifying
 - characters, 175
 - commands, Readline Library, 70–71
 - compile process, spec files, 118
 - configuration files manually, 496–497
 - default prompts, 89
 - default search path, man pages, 284
 - etc/profile/etc/profile, 75
 - etc/services file/etc/services file, 195–196
 - files, 516
 - groups, 397
 - Makefiles, 100–101
 - partition types, 24
 - passwords, 398
 - PATH variable, 73–75
 - priority levels, processes, 82
 - sources.list, 133–134
 - text, 519, 527
 - time stamps, 240–241
 - user accounts, 395–396
- modinfo command, 485–486

modprobe command, 486–487
 modular kernels, 481
 module-path field, 730
 module-type field, 730
 modules

- compiling, 509
- described, 798
- installing, 47, 509
- managing, 482–490
- Pluggable Authentication (PAM), 799

 Monitor “Monitor ID” entry, 348
 Monitor section, XF86Config file, 345–346
 Monitor tab, 352–353
 monitors

- color depths, 353–354
- configuring, 55
- fine-tuning, 357–358
- selecting, 39

 monolithic kernels, 481
 month field, 424
 monthly command, 418
 more command, 162–163, 421
 motherboards, 13
 mount command, 203, 218, 798
 mount point, 25
 mounted access method, 129
 mounting

- exported directories, 639–640
- file systems, 217–220
- volumes, 22, 46

 Mouse Configuration screen, Red Hat 7.0, 32
 mouse. *See* mice
 Mouse tab, 350–351
 move command, 455
 moving

- cursors in vi text editor, 517–518
- files, 245
- jobs in print queues, 458
- processes, 81
- text, 521–523

 mput file, 574
 msg command, 455
 MTA. *See* message transfer agent
 multiboot setup, 28
 Multipurpose Internet Mail Extension (MIME), 798
 multitasking, 8
 multiusers, 8
 mv command, 245
 mx printer definition, 446
 MX record, 665, 798

N

-n <limit> option, 733
 N <passphrase> option, 716
 \n setting, 77
 -N BYTES option, 180
 n column, 123
 n command, 23–24, 208, 527
 N command, 527
 -n level option, 419
 -n NUMBER option, 176–177, 182
 -n option, 394

- depmod command, 489
- file command, 240
- fsck utility, 212
- insmod command, 483
- ip chains command, 721
- ipchains tool, 723
- lpr command, 462
- ls command, 238
- modinfo command, 486
- modprobe command, 486
- mount command, 218
- netstat utility, 584
- nl utility, 184
- ping utility, 576
- route utility, 586
- sed utility, 186
- sort utility, 168
- ssh tool, 713
- traceroute utility, 578
- umount command, 219
- using with rules, 725

 -N option, 213, 586, 713
 -name=Name option, 484
 -name <name> option, 368
 -name option, 249
 named.conf, 654–663
 names

- canonical (CNAME), 792
- in month and day of week fields, 424
- user, 390

 namespaces, DNS servers, 651–654
 naming printers, 473
 NAT. *See* Network Address Translation
 ncurses library, 495
 nds command, 675
 Nessus, 735
 net directory, 482
 netmask ADDRESS option, 582
 Netmask option, 34
 netmasks, 798
 netmasks. *See* subnet masks
 Netscape Navigator, 371

- netscape process, 81–82
- netstat utility, 583–584
- Network Address Translation (NAT), 798
- network cards
 - configuring 16, 34–35
 - support of by Linux, 16
- Network Configuration screen, Red Hat 7.0, 34–35
- Network device support menu, 508
- Network File System (NFS), 392, 799
- Network option, 34
- network services
 - Apache server, 625–636
 - configuring FTP servers, 605–617
 - configuring Telnet, 617
 - Internet Super Server, 602–605
 - Samba, 643–650
 - sendmail, 618–625, 801
- Networking options menu, 508
- networking services, NFS servers, 637–643
- networks
 - addresses, 563–564
 - ARCnet, 791
 - classes of, 565–566
 - configuring, 47–49, 580–590
 - displaying remote X applications across, 380–381
 - Integrated Services Digital (ISDN), 796
 - runlevels, 800
 - subnet masks, 566–569
 - Transmission Control Protocol/Internet Protocol (TCP/IP). *See* Transmission Control Protocol/Internet Protocol
 - troubleshooting, 580–590
- new-chain option, 721
- newer option, 249
- news facility, 415
- newsgroups, 286, 798
- nfs access method, 129
- NFS. *See* Network File System
- NFS servers, 637–643, 685–688
- nfsstat utility, 641
- nG keys, 518
- NIC devices, configuring, 12
- nice command, 82
- nl command, 164–165
- nl utility, 183–185
- nmap tool, 734–735
- NMB service, 799
- \nnn setting, 77
- noauto-config parameter, 17
- nobody account, 389–390
- noclobber option, 540

- nocompress command, 418
- nocopytruncate command, 418
- nocreate command, 418
- nodelaycompress command, 418
- nodeps option, 110, 112
- noexec option, 540
- noexport option, 484
- nofile limit, 732
- noglob option, 540
- noholdall command, 455
- noifempty command, 418
- noksymoops option, 484
- noload option, 483
- non-free component, 133
- none value, 124
- nonlogin shell, 536
- noolddir command, 418
- nopatch option, 137
- noprobe parameter, 17
- not installed value, 124
- not-root option, 120
- NoTrapSignals entry, 342
- nounset option, 540
- np command, 523
- nproc limit, 732
- NS record, 665, 799
- nslookup tool, 671–672
- nsswitch.conf, 670
- NTFS file system, 204
- NUBMER flag, 186
- number address, 186
- number, number address, 186
- +NUMBER option, 177
- NUMBER option, 174, 177
- numbers
 - line, adding to files, 183–185
 - ranges of, time and date fields, 424
- numeric permissions, 253
- nY command, 523
- nyw command, 523
- nyy command, 523
- n| keys, 518

O

- o=option option, 136
- o <option>, 713, 715
- o command, 23, 520
- O command, 520
- o FILE option, 168, 249–250
- O key, 130, 518
- o NAME option, 484
- O NAME option, 484
- o option, 237, 396, 540

- O option, 431–432
- objectives. *See* exam objectives
- obs=BYTES option, 244
- octals, 799
- od utility, 180–181
- of=FILE option, 244
- OffTime <Time> entry, 348
- Old CD-ROM drives menu, 508
- olddir directory command, 418
- open host file, 574
- opening
 - files for editing, 516
 - XF86Setup configuration tool, 350
- operating systems
 - dual booting, 22
 - Linux. *See* Linux
 - Minix, 6
- Option “Option String” entry, 347, 349
- Optional Install and Uninstall Scripts section,
 - spec files, 117
- options option, 488
- options. *See* specific options
- options statement, 656–657
- order statement, 630
- Other tab, 354
- output streams, text, 161–167
- overwrite option, 110, 121
- overwrite-dir option, 121
- overwriting files, 110
- ownerships
 - files, checking, 145
 - fixing, 405

P

- P<printer> option, 461–462
- P<printer name> option, 456
- p <limit> option, 733
- p <n> option, 712–713, 715
- P <passphrase> option, 716
- P <printer> option, 457, 463
- p column, 123
- p command, 23–24, 208, 523
- P command, 523
- p flag, 186
- p option
 - cp command, 242
 - fsck utility, 212
 - insmod command, 483
 - ip chains tool, 722
 - listing current environment variables and values, 535
 - lpr command, 462
 - ls command, 238
 - modinfo command, 486
 - nl utility, 184
 - scp tool, 715
 - shopt command, 540
 - ssh-keygen tool, 716
 - tar utility, 431
- P option, 213, 431, 713, 722
- p pad option, 576
- P pager option, 280
- p passwd option, 394
- p PREFIX option, 174, 484
- package databases, querying, 122–123
- package files, Red Hat Package Manager (rpm),
 - 105–106
- Package Group Selection screen, Red Hat 7.0, 37
- package groups, selecting, 37–38
- package lists, introduction screen, 129–130
- package scripts, printing, 115
- packages. *See also* specific packages; tools
 - apt-get tool, 134–135
 - binary, 95, 337–338
 - checking availability of, 126–127
 - choosing for installation, 52–54
 - clearing archives, 136
 - deb.deb, converting RPM package to,
 - 152–153
 - deleting, 150
 - described, 799
 - displaying information about, 114–115
 - displaying status of, 124–125
 - dpkg tool, 121–122
 - dselect tool, 131–132
 - finding information about, 150
 - finding which one installed a file, 113, 126
 - installed, listing, 145
 - installing, 40–41, 57–58, 109–111
 - installing from remote sites, 148
 - listing, 113, 123–124
 - listing files in, 114, 125
 - managing, 701–702
 - new, running configuration scripts for, 132
 - overwriting files, 110
 - PCMCIA, removing, 51
 - printing information about, 114–115, 122
 - refreshing list of from Debian site, 148
 - removing, 112
 - RPM, converting to .deb package, 152–153
 - task, 119
 - upgrading, 111, 134–135
 - validating integrity, 107–108
 - verifying file changes, 115–116
 - “W” windows, 333
 - xdm, 55, 361–366
 - xinit, 360–361

- packet forwarding, enabling, 727
- pages, man, 277–284, 797
- pairs, key, 715–716
- PAM. *See* Pluggable Authentication Modules (PAM)
- paragraphs, formatting, 173–175
- parallel ports, 18
- parameters
 - ftpd command, 606
 - ftshut command, 615–616
 - h, 111
 - htpasswd command, 633
 - httpd daemon, 627
 - in.telnetd, 617
 - noauto-config, 17
 - noprobe, 17
 - smbclient command, 647–648
 - v, 111
- parameters option, 486
- parent process, 73
- partial backups, 428
- partition type, 24
- partitioning
 - disks, 33, 43–46
 - schemes, 21–28
- partitions
 - changing types of, 24
 - creating, 24–27, 207–211
 - described, 21, 799
 - extended, 207
 - primary, 207
 - removing, 26–27
 - swap, 22, 43–46, 208
- passing arguments to other commands, 166–167
- passwd command, 398, 406
- password module, 730
- PasswordAuthentication entry, 710
- passwords
 - changing, 398
 - root, setting, 36
 - shadow, 391, 398–400, 719–720, 801
- paste utility, 171–172
- pasting text, 521–523
- pasting text, 171–172
- patch=<filename> option, 137
- patches, 491–493
- PATH variable
 - described, 535
 - editing, 73–75
 - init process, 76
 - /usr/bin, 284
- paths
 - absolute, 75
 - checking, 74–75
 - default search, changing for man pages, 284
 - described, 799
 - relative, 799
- <pattern> option, 123
- pattern matching, 524–525
- patterns, tcpd daemon, 699
- PCI network cards, 16
- PCI. *See* Peripheral Component Interface
- PCMCIA packages, removing, 51
- pcmcia.img, 29
- performance, Linux, 9
- period (.), 74–75, 116, 188, 791
- period-slash (./), 74
- Peripheral Component Interface (PCI), 799
- peripherals, Linux support of, 15
- permissions
 - anonymous FTP, 607
 - etc/shadow/etc/shadow file, 399
 - files, 253–258
 - fixing, 405
- PermitEmptyPasswords entry, 710
- PermitRootLogin entry, 710
- PGP, 108
- PGP. *See* Pretty Good Privacy
- PidFile entry, 710
- ping utility, 575–576, 584
- pipes (|), 113, 164–165, 799
- Plug and Play support menu, 507
- Pluggable Authentication Modules (PAM), 728–730, 799
- plus sign (+), 169
- pnpdump command, 21
- Point-to-Point Protocol (PPP), 588–590, 800
- Pointer section, XF86Config file, 344–345
- points, mount, 25
- policy option, 722
- Port “Device Path” entry, 344
- Port entry, 710
- PortMotd entry, 710
- ports
 - Accelerated Graphics (AGP), 791
 - described, 799
 - I/O, listing, 19–20
 - NFS servers, 642
 - parallel, 18
 - serial, 18
 - Transmission Control Protocol (TCP), 571–572
 - User Datagram Protocol (UDP), 571–572

- postrotate command, 418
- PostScript, converting files, to, 464
- PostScript files, 799
- pound sign (#), 389, 414, 542
- PowerPC, 799
- PPP. *See* Point-to-Point Protocol
- pppd utility, 589–590
- pr utility, 181–182
- pre-tests
 - boot process, 296
 - documentation, 276, 292–293
 - installing Linux, 4, 61
 - installing software, 94, 153
 - kernels, 478, 510
 - managing files, 232
 - managing network services, 601
 - managing security, 696, 744
 - managing users and groups, 388, 406
 - networking, 563, 595–596
 - partitions and file systems, 202
 - printing, 444, 474
 - processing text, 160, 197
 - scripts, 514, 554–555
 - shells, 64, 90, 514, 554–555
 - system administration, 410, 440–441
 - X Window System, 332, 382
- pre-tests. *See also* study guides
- prefix option, 100, 484
- Prep section, spec files, 117
- preparing hardware for installation of Linux, 12–18
- prerotate command, 418
- present working directory (pwd), 800
- Pretty Good Privacy (PGP), 799
- Primary DNS option, 34
- primary partitions, 207
- print banners, disabling, 473
- print filters, 450–452
- print jobs, 459–461, 473
- print queues
 - managing, 457–459
 - sending jobs to, 473
 - troubleshooting, 465–466
- printer services, 453–461
- printers
 - assigning aliases to, 473
 - installing, 445–449
 - managing, 453–457
 - setting up, 473
 - support of by Linux, 15
 - troubleshooting, 466
- printing
 - converting files for, 181–182
 - files, 461–464
 - managing printer services, 453–461
 - package information, 114–115, 122
 - package scripts, 115
 - print filters, 450–452
 - troubleshooting, 464–467
- PrintMold option, 710
- PrintTool, 452
- priority field, 414
- priority levels, changing, 82
- priority limit, 732
- probe option, 483
- probing for supported hardware, 20
- processes
 - changing priority levels of, 82
 - child, 73
 - described, 73
 - init, 76
 - managing, 77–81
 - moving, 81
 - netscape, 81–82
 - parent, 73
 - running, 81
 - starting and stopping, 81
 - user, 10
 - viewing, 78
- processing text
 - adding line numbers to files, 183–185
 - converting files for printing, 181–182
 - converting tabs to spaces, 172–173
 - cut utility, 170–171
 - deleting and substituting characters, 175
 - displaying files backwards, 182
 - displaying files in other formats, 180–181
 - displaying numeric details of files, 183
 - dividing files, 179–180
 - enhancing searches with regular expressions, 188–190
 - formatting paragraphs, 173–175
 - grep utility, 187–188
 - joining files, 177–178
 - paste utility, 171–172
 - pipes (|), 164–165
 - redirection, 161–164
 - sorting lines of, 167–170
 - stream editor, 185–187
 - viewing beginning and end of files, 176–177
- Processor type and features menu, 507
- processors, SPARC, 801

programs. *See* applications
 prompt file, 574
 prompt variables, 532–534
 prompts
 customizing, 76
 default, changing, 89
 shell, 67, 76–77
 Protocol entry, 710, 711
 –protocol option, 722
 Protocol “Protocol Name” entry, 344
 protocols
 Address Resolution (ARP), 570, 791
 cd-rom, 133
 described, 800
 Dynamic Host Configuration (DHCP),
 580–581
 Ethernet, 794
 File, 133
 File Transfer (FTP), 133, 150–151, 505–506,
 573–574, 794
 Hypertext Transfer (HTTP), 133, 795
 Internet Control Message (ICMP), 571, 796
 Internet (IP), 569–570, 796
 Point-to-Point (PPP), 588–590, 800
 Samba, 800
 Serial Line Transport (SLIP), 801
 Server Message Block (SMB), 801
 Simple Mail Transport (SMTP), 801
 Telnet, 575, 617, 802
 Transmission Control Protocol/Internet
 (TCP/IP). *See* Transmission
 Control Protocol/Internet Protocol
 Transmission Control (TCP), 571–572, 801
 User Datagram (UDP), 571–572, 802
 X Display Manager Control (XDMCP), 373
 PS1 environment variable, 76
 ps -aux command, 411
 ps command, 78–79
 PS variables. *See* prompt variables
 PTR record, 667
 pty device, 205
 PubkeyAuthentication entry, 709
 public key authentication, 715–718
 public key encryption, 706
 public keys, importing, 146–147
 purge mode, 124
 purge option, 121, 150
 put file, 574
 pwd command, 74
 pwd file, 574
 pwd. *See* present working directory

Q

q command, 23, 209
 :q! command, 517
 q key, 79, 130
 -Q key, 279
 -q option
 depmod command, 489
 head utility, 176
 insmod command, 483
 modprobe command, 486
 ping utility, 576
 scp tool, 715
 shopt command, 540
 ssh tool, 713
 ssh-keygen tool, 716
 sshd command, 712
 tail utility, 177
 -Q option, 463, 712
 -qa option, 113
 -qf option, 113
 -qi option, 114
 -ql option, 114
 -qq option, 136
 query classes, dig tool, 673
 query types, dig tool, 673
 querying
 package databases, 122–123
 RPM databases, 112–115
 questions
 assessment. *See* assessment questions
 Frequently Asked (FAQ), 749, 794
 queues, print, 457–459, 465–466, 473
 –quiet option, 483, 486, 489
 quit command, 454
 Quit menu option, 131
 quitting
 dselect tool, 132
 selection screen, 131
 vi text editor, 516–517
 quota utility, 263–264
 quotaoff utility, 265–266
 quotaon utility, 265–266
 quotas, managing, 263–266
 quoting environment variables, 528–530

R

r <file option, 241
 r comands, 520, 703–705
 R command, 520
 -r function, 430

- r option
 - cp command, 242
 - cpio utility, 432
 - cron daemon, 424
 - depmod command, 489
 - fsck utility, 213
 - gzip utility, 433
 - insmod command, 483
 - lpr command, 462
 - modprobe command, 486
 - mount command, 218
 - netstat utility, 584
 - rm command, 246
 - rmmod command, 484
 - scp tool, 715
 - sort utility, 168
 - syslogd daemon, 414
 - tac utility, 182
 - umount command, 219
- R option, 242, 236, 405, 714, 716
- R <remote account> option, 463
- R value, 586
- RAID. *See* Redundant Array of Inexpensive Disks
- RAM. *See* random access memory
- Ramdac “RAMDAC Type” entry, 347
- ramdisk0 rescue option, 42
- ramdisk1 rescue option, 42
- ramdisks, 499
- random access memory (RAM), 800
- Random Access Memory. *See* memory
- ranges
 - characters, matching, 525
 - numbers, time and date fields, 424
- Raw Write tool, 29, 41
- rcp utility, 704
- reading user input, 548–549
- Readline Library, changing commands, 70–71
- README files, 96, 98, 100
- rebuilddb option, 107
- rebuilding, RPM databases, 107
- records, 664–666, 791
- Red Hat, 7, 10, 12
 - checking ownership of files, 145
 - Disk Druid, 25–26
 - importing public keys, 146–147
 - installing Linux, 28–41
 - labs, 145–147
 - listing installed packages, 145
 - retrieving files, 145–146
 - sndconfig tool, 17
 - starting SYS V, 311–315
- Red Hat Package Manager (RPM)
 - described, 800
 - installing X Window System, 336
 - package files, 105–106
 - RPM database, 105–107
 - rpm tool. *See* rpm tool
- redirect, 800
- redirect command, 455
- Redirect directive, 630
- redirection, text, 161–164
- redo command, 455
- Redundant Array of Inexpensive Disks (RAID), 426
- refreshing, available package lists from Debian site, 148
- regular expressions, 800
 - enhancing searches with, 188–190
- reinstallation required value, 124
- Reiser file system, 800
- relative path, 799
- release command, 455
- remote clients, running, 371–374
- remote logins
 - configuring, 373–374
 - configuring XDM for, 381
- remote sites, installing packages from, 148
- remote X applications, displaying across networks, 380–381
- removable media, support of by Linux, 15
- Remove menu option, 131
- remove mode, 124
- remove option, 486
- remove-essential option, 121
- removing
 - characters, 175
 - comment lines from source lines, 147
 - files, 246
 - groups, 397
 - hosts, 372
 - modules, 484
 - packages, 112, 150
 - apt-get tool, 135
 - dpkg tool, 121–122
 - dselect tool, 131–132
 - partitions, 26–27
 - PCMCIA packages, 51
 - text, 395, 520–521, 720–721
- renice command, 82
- repair disks, creating, 320–321
- repeating searches, 526
- replace option, 721

- replacefiles option, 110
- replacing
 - characters, 175
 - commands, 71
 - text, 519, 526
- reread command, 455
- rescue option, 42
- Resolution <count> entry, 345
- resolv.conf, 671
- resolving, conflicts, plug-and-play hardware, 18–21
- restart command, 455
- restart option, 412
- restarting, inetd daemon, 604–605
- retrieving. *See* downloading
- retry option, 177
- reverse option, 368
- rhosts file, 703
- RhostsAuthentication entry, 711
- RhostsRSAAuthentication entry, 711
- rlogin utility, 704
- rm printer definition, 446
- rmmmod command, 484
- root accounts, 36, 389–390, 800
- root domains, 652–653
- root option, 483, 489
- root passwords, setting, 36
- rotate n command, 418
- rotating system logs, 416–418
- route utility, 585–586
- routers, Linux support of, 9
- routing, Classless Inter-Domain (CIDR), 792
- rp printer definition, 446
- rpm command
 - creating binary packages from source packages, 116–118
 - installing packages, 109–111
 - querying RPM database, 112–115
 - rebuilding databases, 107
 - removing packages, 112
 - upgrading packages, 111
 - validating package integrity, 107–108
 - verifying package files, 115–116
- RPM databases, 105–107, 112–115
- RPM files, downloading, 145–146, 150–151
- RPM. *See* Red Hat Package Manager
- rsh utility, 704
- rss limit, 732
- rules, IP chains, 723–725
- runlevels, 305–311, 800
- running
 - clients remotely, 371–374
 - configuration scripts for new packages, 132

- configure scripts, 98–100
- processes, 81
- X Window System, 359–366
- XF86Setup configuration tool, 350
- runtime utility, 705
- Rute User's Tutorial and Exposition, 749
- rv option, 368
- rwho utility, 705
- rxvt terminal emulator, 369

S

- s command, 24, 186, 520
- s <limit> option, 733
- s [N] option, 180
- s option
 - alien tool, 138
 - apt-get tool, 136
 - cp command, 242
 - depmod command, 489
 - du utility, 216
 - fmt utility, 174
 - insmod command, 483
 - ip chains tool, 722
 - lpq command, 458
 - lpr command, 462–463
 - ls command, 237
 - modprobe command, 486
 - paste utility, 171
 - rmmmod command, 484
 - shopt command, 540
 - ssh-agent tool, 717
 - tar utility, 431
 - tr utility, 175
 - whereis command, 251
- S option
 - whereis command, 251
- S package verification code, 116
- S section-list option, 280
- s SEPARATOR option, 182
- S<server> option, 456
- \s setting, 77
- s size option, 576
- s STRING option, 184
- s switch, 78
- S switch, 78
- SAINT tool, 735
- Samba, 643–650, 799–800
- SampleRate <rate> entry, 345
- saving files, 57, 516–517, 727–728
- Sawfish window manager, 335
- scenarios
 - boot process, 326
 - documentation, 292–293

- installing Linux, 61–62
- installing software, 144, 155
- kernels, 505, 511
- managing files, 271
- managing network services, 682
- managing security, 742, 746
- managing users and groups, 405, 407
- networking, 595, 597
- partitions and file systems, 225
- printing, 472, 475
- scripts, 554, 556–557
- shells, 89, 92, 554, 556–557
- system administration, 439, 442
- X Window System, 380, 383–384
- scheduling, jobs, 421–425
- Scheifler, Robert, 333
- schemes, partitioning, 21–28
- scp tool, 714–715
- Screen section, XF86Config file, 348–349
- ScreenNo <Screen Number> entry, 348
- screens
 - About to Install, Red Hat 7.0, 39
 - Access, 128
 - Account Configuration, Red Hat 7.0, 36
 - Automatic Partitioning, Red Hat 7.0, 33–34
 - color depths, 353–354
 - Individual Package Selection, Red Hat 7.0, 37–38
 - introduction, 129–130
 - Keyboard Configuration, Red Hat 7.0, 30–31
 - Language Selection, Red Hat 7.0, 30–31
 - Mouse Configuration, Red Hat 7.0, 32
 - Network Configuration, Red Hat 7.0, 34–35
 - Package Group Selection, Red Hat 7.0, 37
 - selection, closing, 131
 - Time Zone Selection, Red Hat 7.0, 35–36
 - Unresolved Dependencies, Red Hat 7.0, 37–38
 - Welcome, 29–30, 41–42
 - X Configuration, Red Hat 7.0, 39–40
- ScriptAlias directive, 631
- scripts
 - chat, 588–589
 - configuration, running for new packages, 132
 - configure, 98–100
 - etc/rc.d, 411–412
 - flow control, 545–548
 - package, printing, 115
 - reading user input, 548–549
 - security, 549
 - testing, 543–544
 - writing, 541–549
- SCSI controllers, 13–14
- SCSI devices, configuring, 12
- scsi directory, 482
- SCSI. *See* Small Computer System Interface
- SCSI support menu, 508
- sd device, 205
- sd printer definition, 446
- searches, enhancing with regular expressions, 188–190
- searching
 - files, 248–251
 - information about packages, 150
 - list of installed files, 148–150
 - man pages, 279–283
 - module files, 482–483
 - packages that installed files, 113, 126
 - text, 524–527
- Secondary DNS option, 34
- sections
 - Device, 345–348
 - Files, 341–342
 - man pages, 281–283
 - Monitor, 345–346
 - Pointer, 344–345
 - Screen, 345–348
 - ServerFlags, 342–343
 - spec file, 117
 - XF86Config file, 341–348
- Secure Shell (SSH)
 - authentication methods, 705–712
 - described, 801
- security
 - checking system for, 734–737
 - configuring, X Window System, 371–372
 - Kerberos, 797
 - NFS servers, 642
 - Pretty Good Privacy (PGP), 799
 - root accounts, 389
 - scripts, 549
- security administration
 - configuring TCP wrappers, 697–701
 - IP chains, 721–728
 - limiting users, 732–734
 - managing packages, 701–702
 - monitoring security lists and sites, 730–732
 - Pluggable Authentication Modules (PAM), 728–730
 - r commands, 703–705
 - removing unused services, 720–721
 - shadow passwords, 719–720
 - SUID root issues, 701
- sed utility, 185–187, 196
- seek=BLOCKS option, 244

- Select menu option, 129–131
- selecting
 - absolute paths, 75
 - desktop environments, 367
 - group ids, 392
 - languages, Red Hat 7.0, 30–31
 - monitors, 39
 - package groups, 37–38
 - packages to install, 52–54
 - time zones, 14, 35–36
 - user ids (uid), 392
 - video cards, 39
 - window managers, 367
- selection screen, closing, 131
- selection status modes, dselect tool, 124
- semicolon (;), 69, 527
- sending
 - commands, 67
 - data. *See* processing text
 - jobs to print queues, 473
- sendmail, 618–625, 801
- sendmail.cf, configuring, 619–623
- separating, directories by volume, 21–22, 44
- Serial Line Transport Protocol (SLIP), 801
- serial ports, 18
- server command, 455
- Server Message Block (SMB), 801
- server statement, 658–659
- Server System option, 32
- ServerFlags section, XF86Config file, 342–343
- ServerKeyBits entry, 711
- servers
 - Apache. *See* Apache servers
 - DNS. *See* DNS servers
 - FTP, 605–617
 - Internet Super, 602–605
 - Linux support of, 9
 - NFS, 637–643, 685–688
 - Secure Shell (SSH) configuring, 708–712
 - XFree v3, 339
- service-name field, 730
- services
 - boot, 326–328
 - network. *See* network services
 - NMB, 799
 - printer, 453–461
 - runlevels, 800
 - unused, removing from hosts, 720–721
 - Windows Internet Naming (WINS), 802
- session module, 730
- set command, 535, 539–540
- set option, 722
- setgid command, 702
- setting. *See* configuring
- settings
 - \$(date), 77
 - \#, 77
 - \\$, 77
 - \!, 77
 - \[, 77
 - \], 77
 - BIOS, 14
 - clockchip, 56
 - configuring prompts, 77
 - current, displaying, 76
 - d \d, 77
 - d \h, 77
 - environment variables, 72–77
 - global, configuring, 400–402
 - n \n, 77
 - nnn \nnn, 77
 - s \s, 77
 - SUID, 801
 - \t, 77
 - time, 14, 35
 - \u, 77
 - user, configuring, 400–402
 - \W, 77
 - \w, 77
- setups, multiboot, 28
- SGID file setting, 258
- sh printer definition, 446
- sh shell, 66
- shadow passwords, 391, 398–400, 719–720
 - described, 801
- shared directories, creating, 742–743
- shareware, 7
- SHELL environment variable, 65
- shell prompts, 67, 76–77
- SHELL variable, 423
- shells
 - aliases, 536
 - ash, 66
 - bash2, 66
 - bash, 65–66, 792
 - bsh, 66
 - command lines as, 67–72
 - csh, 66, 793
 - described, 65–66, 801
 - environment variables, 528–535
 - Korn, 65
 - login files, 536–537
 - options, 539–541
 - Secure (SSH), 801

- sh, 66
 - standardizing users on, 66–67
 - tcsh, 66
- shells option, 66
- shopt command, 540–541
- show option, 486, 489
- showconfig option, 486
- SIGHUP signal, 412
- SIGKILL signal, 412
- signal levels, viewing, 80–81
- signals, kill command, 412
- SIGTERM signal, 412
- Simple installation method, 53–54
- Simple Mail Transport Protocol (SMTP), 801
- single-user mode, 319
- sites
 - remote, installing packages from, 148
 - Web. *See* Web sites
- sites. *See* Web sites
- size n command, 418
- sizes, maximum print jobs, configuring, 473
- SkeyAuthentication entry, 711
- skip=BLOCKS option, 244
- Slackware, 11
- slash (/), 68
- SLIP. *See* Serial Line Transport Protocol
- Small Computer System Interface (SCSI), 800
- SMB. *See* Server Message Block
- smbclient command, 647–650
- smf.conf, 645–646
- SMTP. *See* Simple Mail Transport Protocol
- sndconfig tool, 17
- SOA records, 664–665
- software
 - availability of for Linux, 9
 - compiling, 101–102
 - installing. *See* installing software
- software. *See* applications
- sort utility, 167–170, 196
- sorting, lines of files, 167–170
- sound cards
 - configuring, 17
 - support of by Linux, 17
- Sound menu, 508
- source code
 - installing software from, 95–102
 - kernels, 490–491, 493
- source command, 537
- source lines, removing comment lines from, 147
- source option, 722
- source packages, creating binary packages from, 116–118
- source-port option, 722
- sources.list, changing, 133–134
- space
 - hard disk requirements, installing Linux, 14
 - kernel, 9–10
 - spool files, 467
 - user, 10
 - volumes, consumption of, 22
- spacebar, 279
- spaces
 - converting tabs to, 172–173
 - swap, 801
- SPARC processor, 801
- spec file, 117–118
- special users, 389–390
- specifying, absolute paths, 75
- speed, Linux, 9
- split utility, 179–180
- splitting files, 179–180
- spool directories, 448–449, 473
- spool files, 467
- sport option, 722
- sr device, 205
- src option, 722
- SRV record, 666
- SSH. *See* Secure Shell
- ssh tool, 712–714
- ssh-add tool, 717–718
- ssh-agent tool, 717
- ssh-keygen tool, 715–716
- sshd command, 711
- sshd_config command, 709–711
- st device, 205
- stability of Linux, 8
- stable branch, 480
- stable distribution, 133
- stack limit, 732
- stacks option, 484
- standard error, 161
- standard input, 161
- standard output, 161
- standardizing, users on one shell, 66–67
- StandByTime <Time> entry, 348
- start command, 455
- start option, 412
- starting
 - daemons, 411–412
 - httpd daemon, 626–627
 - printers, 457
 - processes, 81
 - X Window System, 359–366
 - XF86Setup configuration tool, 350

- StartServers directive, 629
 - startx command, 359–361
 - statements
 - case, 546–547
 - if/else, 545–546
 - named.conf, 656–663
 - order, 630
 - until, 547–548
 - while, 547–548
 - status
 - packages, viewing, 124–125
 - print queues, 458–459
 - printers, checking, 473
 - status command, 455
 - status file, 127
 - status option, 411
 - stderr command, 161–165
 - stdin command, 161–162, 165
 - stdout command, 161–165
 - stick bit, 258
 - sticky bits, 801
 - stop command, 455
 - stop option, 412
 - stopping
 - daemons, 411–412
 - httpd daemon, 626–627
 - printers, 456
 - processes, 81
 - storing user and group information, 391–392
 - stream editor. *See* sed utility
 - streams, 161–167
 - StrictModes entry, 711
 - ?string command, 527
 - /string command, 527
 - strings, prompt customization, 534
 - study guides. *See also* LPIC certification exam
 - boot process, 323–330
 - documentation, 290–293
 - installing Linux, 59–62
 - installing software, 141–155
 - kernels, 502–511
 - managing files, 268–273
 - managing network services, 679–693
 - managing security, 739–746
 - managing users and groups, 403–407
 - networking, 592–597
 - partitions and file systems, 222–229
 - printing, 469–475
 - processing text, 192–197
 - scripts, 551–557
 - shells, 85–92, 551–557
 - system administration, 436–442
 - X Window System, 377–384
 - su command, 390, 575
 - subdirectories, /usr/src/redhat tree, 117
 - subnet masks, 566–569
 - SubSection “Display” entry, 348
 - subsections, Display, 349
 - Subsystem entry, 711
 - SUID file setting, 258, 801
 - superblocks, 206, 801
 - SuperProbe tool, 356–357
 - superuser. *See* root account
 - support
 - CD drives, 15
 - DVD drives, 15
 - mice, 15
 - printers, 15
 - technical, 287–288
 - SuSe, 7, 11–12
 - SuspendTime <Time> entry, 348
 - swap partitions, 22, 208
 - configuring, 43–46
 - swap space, 801
 - swapping, 13
 - switches, 78
 - switching
 - consoles, Red Hat installations, 28
 - directories, 233–235
 - shells, 65
 - symbolic links, 252, 801
 - symbolic permissions, 253
 - SYS V startup, 311–315
 - syslog facility, 415
 - syslog option, 483–484, 486, 489
 - syslogd daemon, 413–414
 - SysLogFacility entry, 711
 - system administration
 - backups, 425–434
 - documentation and, 287
 - limiting core dump files, 434
 - scheduling jobs, 421–425
 - starting and stopping daemons, 411–412
 - system logging, 413–421
 - technical support, 287–288
 - system directories, 248
 - system logs, 287, 413–421
 - systems, configuring, Red Hat installations, 32–34
- T**
- t CHAR option, 178
 - T COLS option, 238
 - t command, 24
 - T FILE option, 431
 - t FILESYSTEM option, 216

- t FILESYSTEM_TYPE option, 218
- t fstype option, 210
- t function, 430
- t <n seconds> option, 733
- t option
 - expand utility, 172
 - fmt utility, 174
 - gzip utility, 433
 - lpr command, 462
 - ls command, 237
 - modprobe command, 487
 - netstat utility, 584
 - nl utility, 184
- T package verification code, 116
- \t setting, 77
- t<sleep time> option, 458
- t <TIME> option, 241
- T <title> option, 462–463
- t <type> option, 716
- t TYPE option, 180
- ta command, 527
- Ta command, 527
- Tab key, 70
- tabs
 - Card, 352
 - converting to spaces, 172–173
 - Keyboard, 351
 - Modeselection, 353–354
 - Monitor, 352–353
 - Mouse, 350–351
 - Other, 354
- tac utility, 182
- tail utility, 176–177
- tape drives, 15, 428
- tar files, 337
- tar utility, 259–260, 429–431
- tar zxvf command, 98
- tarballs, decompressing, 97–98
- task package, 119
- TCP. *See* Transmission Control Protocol
- TCP wrappers, configuring, 697–701
- tcpd daemon, 697–701
- tcpdchk command, 700
- tcsh shell, 66
- technical support, 287–288
- tee command, 196
- tee utility, 165–166
- Telephony support menu, 508
- Telnet, 575, 617, 802
- temporary files, 497
- terminal emulators, 368–369
- Tertiary DNS option, 34
- test command, 543–544
- test engines, 748–749
- testing
 - kernels, 500, 509
 - scripts, 543–544
- tests, arithmetic, 548. *See also* exam objectives; LPIC certification exam; study guides
- text
 - adding line numbers to files, 183–185
 - appending, 519
 - ASCII, 792
 - changing, 519, 527
 - converting files for printing, 181–182
 - converting tabs to spaces, 172–173
 - copying, 521–523
 - creating new lines, 519
 - cutting, 170–171
 - deleting, 520–521
 - deleting and substituting characters, 175
 - displaying files backwards, 182
 - displaying files in other formats, 180–181
 - displaying numeric details of files, 183
 - dividing files, 179–180
 - enhancing searches with regular expressions, 188–190
 - formatting paragraphs, 173–175
 - grep utility, 187–188
 - inserting, 519
 - joining files, 177–178
 - moving, 521–523
 - pastings, 171–172
 - pipes(), 164–165
 - redirection, 161–164
 - replacing, 519
 - searching for, 524–527
 - sorting lines of files, 167–170
 - stream editor, 185–187
 - viewing beginning and end of files, 176–177
- text <ENTER> option, 30
- text editors. *See specific text editors*
- themes, 334
- three-button emulation, 55
- time fields, 424
- time specifications, at command, 422
- time stamps, changing, 240–241
- Time Zone Selection screen, Red Hat 7.0, 35–36
- time zones, selecting, 14, 35–36
- tkrat mail agent, 371
- TLD. *See* top-level domains
- to-slp option, 137
- tools. *See* packages; *specific tools*
- top command, 79–80

- top-level domains (TLD), 651
 - topq command, 455
 - Torvalds, Linus, 5–6, 479
 - touch command, 240–241, 743
 - tr utility, 175
 - traceroute utility, 577–578, 584–585
 - translation, Network Address (NAT), 798
 - Transmission Control Protocol (TCP), 571–572, 801
 - Transmission Control Protocol/Internet Protocol (TCP/IP)
 - addresses, 564–565
 - applications used by, 573–579
 - classes of networks, 565–566
 - port assignments, 571–572
 - protocols used by, 569–571
 - subnet masks, 566–569
 - trees, /usr/src/redhat, 117
 - Tripwire, 736–737
 - Trolltech, 335
 - troubleshooting
 - boot process, 316–321
 - CD, 752
 - conflicts, plug-and-play hardware, 18–21
 - log files, 418–421
 - networks, 580–590
 - printing problems, 464–467
 - trusted-key statement, 663
 - ttl field, 663
 - tty device, 205
 - ttyS device, 205
 - Turbolinux, 12
 - turning off. *See* disabling
 - turning on. *See* enabling
 - TXT record, 666
 - txx switch, 78
 - type moduletype option, 487
 - TypesConfig directive, 631
- U**
- u column, 123
 - u command, 24, 527
 - U command, 527
 - u function, 430
 - u <limit> option, 733
 - u <n> option, 712
 - u option
 - cpio utility, 432
 - fmt utility, 174
 - locate command, 250
 - ls command, 237
 - mv command, 245
 - netstat utility, 584
 - shopt command, 540
 - whereis command, 251
 - U option 111, 396
 - U package verification code, 116
 - U /PATH option, 249–250
 - \u setting, 77
 - u switch, 78
 - u -uid option, 394, 396
 - u user option, 424
 - U<user> option, 461–462
 - U<username> option, 456, 463
 - U value, 586
 - UDP. *See* User Datagram Protocol
 - uid. *See* user id
 - ulimit command, 733
 - ulimit tool, 802
 - Ultrastor, 13
 - umask command, 257–258
 - umask utility, 801
 - umount command, 218–219
 - undefined macro, 620
 - undo, vi text editor, 527
 - Uniform Resource Identifier (URI), 133
 - uninstall option, 112
 - University of Berkeley, 515
 - University of Helsinki, 5
 - UNIX operating system, accessing partitions, 25
 - unknown mode, 124
 - unmounting, file systems, 217–220
 - unpacked value, 124
 - Unresolved Dependencies screen, Red Hat 7.0, 37–38
 - unstable distribution, 133
 - until statement, 547–548
 - up arrow key, 72
 - up command, 455
 - up option, 582
 - Update menu option, 129
 - updating
 - apt-get tool distributions, 135–136
 - Linux LOader (LILO), 303–304
 - package lists, 148
 - packages, 134–135
 - system logs, 287
 - upgrading
 - apt-get tool distributions, 135–136
 - packages, 111, 134–135
 - URI. *See* Uniform Resource Identifier
 - USB support menu, 508
 - Use Graphical Login option, 39
 - UseLogin entry, 711

User Datagram Protocol (UDP), 571–572, 802
 user facility, 415
 user id (uid), 390, 392
 user names, 390
 user option, 249
 user permissions, 254–258
 user processes, 10
 user space, 10
 useradd command, 390–391, 394
 UserDir directive, 631
 usermod command, 395–396
 users
 adding, 390–394, 405–406
 configuring settings, 400–402
 deleting, 395
 lp, 448
 managing, 393–400
 modifying accounts, 395–396
 reading input by, 548–549
 special, 389–390
 standardizing on one shell, 66–67
 /usr/doc/usr/doc, 284–285
 /usr/src/redhat directory, 117
 UTC. *See* Coordinated Universal Time
 UTC/GMC time setting, 14
 utilities. *See* packages; specific utilities
 uucp facility, 415

V

-V <client protocol ID> option, 712
 v command, 24, 209
 -v FILE# option, 178
 V key, 130
 -v <limit> option, 733
 -v NUMBER option, 184
 -v option
 at utility, 422
 cp command, 242
 cpio utility, 432
 depmod command, 489
 file command, 240
 fsck utility, 213
 gzip utility, 433
 head utility, 176
 insmod command, 483
 ipchains tool, 723
 lpr command, 462
 modprobe command, 487
 mount command, 218
 mv command, 245
 ping utility, 576
 rm command, 246

route utility, 586
 scp tool, 715
 ssh tool, 714
 tail utility, 177
 tar utility, 431
 tcpdchk command, 700
 traceroute utility, 578
 umound command, 219
 -V option
 depmod command, 489
 fsck utility, 213
 lpc command, 456
 lpq command, 457
 lpr command, 464
 mkfs utility, 210
 modinfo command, 486
 modprobe command, 487
 mount command, 218
 umount command, 219
 -v parameter, 111
 validating, package integrity, 107–108
 values
 config-files, 124
 current status, 124
 failed-config, 124
 flags, 586
 half-installed, 124
 hold, 124
 installed, 124
 listing, 535
 none, 124
 not installed, 124
 reinstallation required, 124
 unpacked, 124
 /var/lib/dpkg directory, 126
 /var/lib/rpm/var directory, 107, 113
 /var/log/auth.log, 415
 /var/log/kern.log, 415
 /var/log/lastlog file, 415–416
 /var/log/mail/mail.warn, 415
 /var/log/messages file, 415
 /var/log/utmp file, 415–416
 /var/log/wtmp file, 415–416
 variables
 CDPATH, 535
 DISPLAY, 372
 EDITOR, 401
 environment. *See* environment variables
 exporting, 535
 HISTCMD, 72, 530–531

Continued

variables (*continued*)

- HISTFILESIZE, 72, 531
- HISTIGNORE, 530–531
- HISTSIZ, 531
- HOME, 77, 423
- LOGNAME, 423
- MAIL, 531–532
- MAILCHECK, 531–532
- MAILPATH, 531–532
- MANPATH, 279, 284
- PATH, 73–76, 284, 535
- prompt, 532–534
- PS1, 76
- quoting, 528–530
- SHELL, 65, 423

VendorName “Vendor” entry, 346–347

vendors, 286

verbose file, 574

–verbose option 179, 483, 487, 489, 723

verifying

- available packages, 126–127
- bad blocks, 44–45
- boot services, 326–328
- dependencies, 37–38 122
- file ownership, 145
- file systems, 212–217
- package file changes, 115–116
- package integrity, 107–108
- packages that installed files, 113, 126
- paths, 74–75
- print queue status, 458–459
- security on systems, 734–737
- source code, kernels, 493
- status of printers, 473

–version option

- cut utility, 170
- depmod command, 489
- expand utility, 172
- head utility, 176
- join utility, 178
- modinfo command, 486
- modprobe command, 487
- od utility, 181
- sort utility, 168
- split utility, 179
- tail utility, 177
- tr utility, 175
- wc utility, 183

versions

- kernels, 479–480
- X Window System, 338–340

VertRefresh <vertical refresh range> entry, 346

vfat file system, 802

vi option, 540

vi text editor, 802

- adding text, 519–520
- copying and pasting text, 521–523
- deleting text, 520–521
- editing /etc/profile, 75
- exiting and saving files, 516–517
- moving cursor in, 517–518
- opening files for editing, 516
- searching for text, 524–527
- text editing practices, 515
- undoing changes in, 527

video, fine-tuning, 357–358

video cards, 39, 55–56, 346–347

video directory, 483

video requirements, installing Linux, 14

VideoRam <memory> entry, 347

viewing

- beginning of files, 176
- configuration addresses, 18–20
- current settings, 76
- end of files, 176–177
- environment variable settings, 73
- files backwards, 182
- files in other formats, 180–181
- members of groups, 397–398
- messages, 419
- module information, 485–490
- numeric details of files, 183
- package information, 114–115
- priority levels, 82
- processes, 78
- remote X applications across networks, 380–381
- signal levels, 80–81
- status of packages, 124–125

ViewPort <x0> <y0> entry, 349

Virtual <x dimensions> <y dimensions> entry, 349

virtual hosts, Apache servers, 633–636

virtual memory, 22

volumes

- configuring, 43–46
- initializing, 45
- separating directories by, 21–22, 44

W

–w BYTES option 181

–w COLS option, 238

w command, 24, 209

:w command, 516

w filename flag 186

w key, 58

- w<num> option, 462
- w NUMBER option, 174, 184
- w option, 183, 218, 280–281, 431, 578
- W option, 431
- \w setting, 77
- \W setting, 77
- w <width> option, 464
- w WIDTH option, 182
- W WIDTH option, 182
- “W” windows package, 333
- wc utility, 183
- Web browsers, Netscape Navigator, 371
- Web sites
 - AfterStep window manager, 334
 - ALSA project, 17
 - Apsfilter, 450
 - Blackbox window manager, 334
 - Caldera, 11
 - configuring printing in Linux, 445
 - Debian, 11
 - list of available packages, 148
 - Enlightenment window manager, 334
 - FVWM window manager, 34
 - GnuPG, 108
 - IceWM window manager, 335
 - kernels, 491
 - Linux benchmark example, 9
 - Linux Documentation Project, 285
 - Magicfilter, 451
 - Mandrake, 11
 - newest Hardware-HOWTO, 12
 - PGP, 108
 - Raw Write, 41
 - Red Hat, 10
 - Red Hat Package Manager (rpm)
 - packages, 106
 - Sawfish window manager, 335
 - security, 731
 - Slackware, 11
 - source code for applications, 96
 - SuSe, 11
 - tar files, 337
 - Turbolinux, 12
 - Window Maker window manager, 335
 - window manager themes, 334
 - window managers, 334–335
 - XFree86, 333
- Weight <RGB> entry, 349
- Welcome screen, 29–30, 41–42
- Western Digital, 13
- wget tool, installing packages from remote sites, 148
- whatis utility, 282

- whereis command, 251
- which command, 250
- while statement, 547–548
- whois utility, 578–579
- width, default, fmt utility, 173
- wildcards. *See* asterisk
- Window Maker, 335
- window managers, X Window System,
 - 333–334, 367
- WindowMaker, 55
- Windows, installing CD from, 747
- Windows Internet Naming Service (WINS), 802
- Winmodems, 16
- WINS. *See* Windows Internet Naming Service
- words, deleting, 520
- Workstation option, 32–33
- workstations, separating volumes on, 22
- :wq command, 517
- wrappers, TCP, 697–701
- Write option, 27
- writing, scripts, 541–549

X

- X11, 333
- X11DiaplsyOffset entry, 711
- X11Forwarding entry, 711
- x86, 802
- X clients, 368
- x command, 24, 521
- X command, 521
- X Configuration screen, Red Hat 7.0, 39–40
- X Display Manager Control Protocol (XDMCP), 373
- x FILESYSTEM option, 217
- x function, 430
- X key, 130
- x option, 216, 238, 484, 714
- X option, 431, 714, 716
- x switch, 78
- x term terminal emulator, 369
- X Toolkit, 368–370
- X Window System, 14
 - architecture, 333–334
 - configuring, 39–40, 54–57
 - configuring fonts, 358–359
 - configuring XF86Config file, 341–348
 - customizing applications in, 369–370
 - desktop environments, 335–336, 367
 - detecting video hardware, 356–357
 - fine-tuning video, 357–358
 - history of, 333

Continued

- X Window System (*continued*)
 - installing, 336–337
 - Main Menu options, 507–508
 - managing bad applications in, 370–371
 - running clients remotely, 371–374
 - special key combinations in, 370
 - starting, 359–366
 - terminal emulators, 368–369
 - versions of, 338–340
 - window managers, 333–334, 367
 - X clients, 368
 - xf86config configuration tool, 355–356
 - XF86Setup configuration tool, 349–354
 - xargs utility, 166–167
 - xauth tool, 360
 - XAuthLocation entry, 711
 - Xclient files, 361, 367
 - XDM, configuring for remote logins, 381
 - xdm package, 55, 361–366
 - xdm-config file, 362–363
 - XDMCP. *See* X Display Manager Control Protocol
 - xf86config configuration tool, 355–356
 - XF86Config file, configuring, 341–348
 - XF86Setup configuration tool, 349–354
 - xferlog file, 614–615
 - XFree86. *See* X Window System
 - xhost command, 183, 371–372
 - Ximian, 335–336
 - xinit package, 360–361
 - xinitrc file, 361, 367
 - Xresources file, 365–366
 - Xsession file, 363–365, 367
 - xvidtune tool, 357–358
- ## Y
- y\$ command, 523
 - Y command, 521, 523
 - y option, 136, 213, 484, 716
 - Y option, 484
 - yanking text, 521–523
 - ynw command, 523
 - yw command, 523
 - yy command, 521, 523
- ## Z
- ^Z command, 81
 - Z <options>, 464
 - z option, 431, 721
 - Z option, 240, 431
 - ZaxisMapping N M entry, 345
 - zero option, 721
 - Zip drives, 428
 - zone files, 654, 663–664
 - zone statement, 659–662
 - zones
 - described, 802
 - time, selecting, 14, 35–36
 - ZZ command, 517

Hungry Minds, Inc.

End-User License Agreement

READ THIS. You should carefully read these terms and conditions before opening the software packet(s) included with this book (“Book”). This is a license agreement (“Agreement”) between you and Hungry Minds, Inc. (“HMI”). By opening the accompanying software packet(s), you acknowledge that you have read and accept the following terms and conditions. If you do not agree and do not want to be bound by such terms and conditions, promptly return the Book and the unopened software packet(s) to the place you obtained them for a full refund.

- 1. License Grant.** HMI grants to you (either an individual or entity) a non-exclusive license to use one copy of the enclosed software program(s) (collectively, the “Software”) solely for your own personal or business purposes on a single computer (whether a standard computer or a workstation component of a multi-user network). The Software is in use on a computer when it is loaded into temporary memory (RAM) or installed into permanent memory (hard disk, CD-ROM, or other storage device). HMI reserves all rights not expressly granted herein.
- 2. Ownership.** HMI is the owner of all right, title, and interest, including copyright, in and to the compilation of the Software recorded on the disk(s) or CD-ROM (“Software Media”). Copyright to the individual programs recorded on the Software Media is owned by the author or other authorized copyright owner of each program. Ownership of the Software and all proprietary rights relating thereto remain with HMI and its licensors.
- 3. Restrictions On Use and Transfer.**
 - (a)** You may only (i) make one copy of the Software for backup or archival purposes, or (ii) transfer the Software to a single hard disk, provided that you keep the original for backup or archival purposes. You may not (i) rent or lease the Software, (ii) copy or reproduce the Software through a LAN or other network system or through any computer subscriber system or bulletin-board system, or (iii) modify, adapt, or create derivative works based on the Software.
 - (b)** You may not reverse engineer, decompile, or disassemble the Software. You may transfer the Software and user documentation on a permanent basis, provided that the transferee agrees to accept the terms and conditions of this Agreement and you retain no copies. If the Software is an update or has been updated, any transfer must include the most recent update and all prior versions.

4. Restrictions on Use of Individual Programs. You must follow the individual requirements and restrictions detailed for each individual program in Appendix A of this Book. These limitations are also contained in the individual license agreements recorded on the Software Media. These limitations may include a requirement that after using the program for a specified period of time, the user must pay a registration fee or discontinue use. By opening the Software packet(s), you will be agreeing to abide by the licenses and restrictions for these individual programs that are detailed in Appendix A and on the Software Media. None of the material on this Software Media or listed in this Book may ever be redistributed, in original or modified form, for commercial purposes.

5. Limited Warranty.

- (a) HMI warrants that the Software and Software Media are free from defects in materials and workmanship under normal use for a period of sixty (60) days from the date of purchase of this Book. If HMI receives notification within the warranty period of defects in materials or workmanship, HMI will replace the defective Software Media.
- (b) **HMI AND THE AUTHOR OF THE BOOK DISCLAIM ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WITH RESPECT TO THE SOFTWARE, THE PROGRAMS, THE SOURCE CODE CONTAINED THEREIN, AND/OR THE TECHNIQUES DESCRIBED IN THIS BOOK. HMI DOES NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE SOFTWARE WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE SOFTWARE WILL BE ERROR FREE.**
- (c) This limited warranty gives you specific legal rights, and you may have other rights that vary from jurisdiction to jurisdiction.

6. Remedies.

- (a) HMI's entire liability and your exclusive remedy for defects in materials and workmanship shall be limited to replacement of the Software Media, which may be returned to HMI with a copy of your receipt at the following address: Software Media Fulfillment Department, Attn.: *LPIC 1 Certification Bible*, Hungry Minds, Inc., 10475 Crosspoint Blvd., Indianapolis, IN 46256, or call 1-800-762-2974. Please allow four to six weeks for delivery. This Limited Warranty is void if failure of the Software Media has resulted from accident, abuse, or misapplication. Any replacement Software Media will be warranted for the remainder of the original warranty period or thirty (30) days, whichever is longer.

- (b) In no event shall HMI or the author be liable for any damages whatsoever (including without limitation damages for loss of business profits, business interruption, loss of business information, or any other pecuniary loss) arising from the use of or inability to use the Book or the Software, even if HMI has been advised of the possibility of such damages.
- (c) Because some jurisdictions do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitation or exclusion may not apply to you.

7. U.S. Government Restricted Rights. Use, duplication, or disclosure of the Software for or on behalf of the United States of America, its agencies and/or instrumentalities (the "U.S. Government") is subject to restrictions as stated in paragraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause of DFARS 252.227-7013, or subparagraphs (c) (1) and (2) of the Commercial Computer Software - Restricted Rights clause at FAR 52.227-19, and in similar clauses in the NASA FAR supplement, as applicable.

8. General. This Agreement constitutes the entire understanding of the parties and revokes and supersedes all prior agreements, oral or written, between them and may not be modified or amended except in a writing signed by both parties hereto that specifically refers to this Agreement. This Agreement shall take precedence over any other documents that may be in conflict herewith. If any one or more provisions contained in this Agreement are held by any court or tribunal to be invalid, illegal, or otherwise unenforceable, each and every other provision shall remain in full force and effect.

CD Installation Instructions

Each software item on the *LPIC 1 Certification Bible* CD-ROM is located in its own folder. To install a particular piece of software, open its folder with My Computer or Internet Explorer. What you do next depends on what you find in the software's folder:

1. First, look for a `ReadMe.txt` file or a `.doc` or `.htm` document. If this is present, it should contain installation instructions and other useful information.
2. If the folder contains an executable (`.exe`) file, this is usually an installation program. Often it will be called `Setup.exe` or `Install.exe`, but in some cases the filename reflects an abbreviated version of the software's name and version number. Run the `.exe` file to start the installation process.

The `ReadMe.txt` file in the CD-ROM's root directory may contain additional installation information, so be sure to check it.

For a listing of the software on the CD-ROM, see Appendix A.