

Matthew Helmke

Ubuntu

UNLEASHED

SAMS

2013 Edition
Covering 12.10 and 13.04

FREE SAMPLE CHAPTER



SHARE WITH OTHERS

Matthew Helmke
with Andrew Hudson
and Paul Hudson

Ubuntu

UNLEASHED

2013 Edition

SAMS

800 East 96th Street, Indianapolis, Indiana 46240 USA

Ubuntu Unleashed 2013 Edition

Copyright © 2013 by Pearson Education, Inc.

All rights reserved. No part of this book shall be reproduced, stored in a retrieval system, or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without written permission from the publisher. No patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this book, the publisher and author assume no responsibility for errors or omissions. Nor is any liability assumed for damages resulting from the use of the information contained herein.

ISBN-13: 978-0-672-33624-9

ISBN-10: 0-672-33624-3

The Library of Congress cataloging-in-publication data is on file.

Printed in the United States of America

First Printing December 2012

Trademarks

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Sams Publishing cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

Warning and Disclaimer

Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied. The information provided is on an "as is" basis. The author and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book or programs accompanying it.

Bulk Sales

Sams Publishing offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales. For more information, please contact

U.S. Corporate and Government Sales
1-800-382-3419
corpsales@pearsontechgroup.com

For sales outside of the U.S., please contact

International Sales
international@pearsoned.com

Editor-in-Chief

Mark Taub

Acquisitions Editor

Debra Williams
Cauley

Development Editor

Michael Thurston

Managing Editor

Kristy Hart

Project Editor

Jovana Shirley

Copy Editor

Charlotte Kughen

Indexer

Angie Martin

Proofreader

Language Logistics

Technical Editors

Chris Johnston
Shannon Oliver

Editorial Assistant

Kim Boedigheimer

Media Producer

Dan Scherf

Cover Designer

Anne Jones

Compositor

Nonie Ratcliff

Contents at a Glance

Introduction	1
Part I Installation and Configuration	
1 Installing Ubuntu and Post-Installation Configuration	7
Part II Desktop Ubuntu	
2 Working with Unity	33
3 On the Internet	51
4 Productivity Applications	65
5 Multimedia Applications	75
6 Other Ubuntu Interfaces	99
7 Games	107
Part III System Administration	
8 Managing Software	119
9 Command-Line Quickstart	133
10 Command-Line Master Class	169
11 Managing Users	209
12 Automating Tasks and Shell Scripting	237
13 The Boot Process	281
14 System-Monitoring Tools	291
15 Backing Up	307
16 Networking	335
17 Remote Access with SSH and Telnet	381
18 Securing Your Machines	391
19 Performance Tuning	405
20 Kernel and Module Management	417
Part IV Ubuntu as a Server	
21 Sharing Files and Printers	439
22 Apache Web Server Management	461
23 Other Http Servers	491
24 Remote File Serving with FTP	497
25 Handling Email	507

26	Proxying, Reverse Proxying, and Virtual Private Networks (VPN).....	527
27	Administering Relational Database Services.....	543
28	NoSQL Databases.....	569
29	Lightweight Directory Access Protocol (LDAP).....	581
30	Linux Terminal Server Project (LTSP).....	589
31	Virtualization on Ubuntu.....	597
32	Ubuntu in the Cloud.....	607
33	Managing Sets of Servers.....	641
Part V	Programming Linux	
34	Opportunistic Development.....	645
35	Helping with Ubuntu Development.....	665
36	Helping with Ubuntu Testing and QA.....	675
37	Using Perl.....	681
38	Using Python.....	703
39	Using PHP.....	723
40	C/C++ Programming Tools for Ubuntu.....	755
41	Using Other Popular Programming Languages.....	767
42	Beginning Mobile Development for Android.....	779
Part VI	Appendices	
A	Ubuntu Under the Hood.....	787
B	Ubuntu and Linux Internet Resources.....	797
	 Index.....	 807

Table of Contents

Introduction	1
Licensing	2
Who This Book Is For	3
Those Wanting to Become Intermediate or Advanced Users	3
Sysadmins, Programmers, and DevOps	4
What This Book Contains	5
Conventions Used in This Book	5
Part I Installation and Configuration	
1 Installing Ubuntu and Post-Installation Configuration	7
Before You Begin the Installation	7
Researching Your Hardware Specifications	8
Installation Options	8
Planning Partition Strategies	10
The Boot Loader	10
Installing from DVD or USB Drive	11
Step-by-Step Installation	11
Installing	12
First Update	16
Wubi: The Easy Installer for Windows	16
Shutting Down	18
Finding Programs and Files	19
Software Updater	19
The <code>sudo</code> Command	22
Configuring Software Repositories	23
System Settings	26
Detecting and Configuring a Printer	26
Configuring Power Management in Ubuntu	27
Setting the Time and Date	27
Configuring Wireless Networks	29
Troubleshooting Post-Installation Configuration Problems	31
References	32

Part II Desktop Ubuntu

2	Working with Unity	33
	Foundations and the X Server	33
	Basic X Concepts	34
	Using X	35
	Elements of the <code>xorg.conf</code> File	36
	Starting X	41
	Using a Display Manager	41
	Changing Window Managers	42
	Using Unity, a Primer	42
	The Desktop	43
	Customizing and Configuring Unity	48
	Power Shortcuts	49
	References	50
3	On the Internet	51
	Getting Started with Firefox	52
	Checking Out Google Chrome and Chromium	53
	Choosing an Email Client	55
	Mozilla Thunderbird	56
	Evolution	56
	Other Mail Clients	57
	RSS Readers	58
	Firefox	58
	Liferea	58
	Instant Messaging and Video Conferencing with Empathy	59
	Internet Relay Chat	60
	Usenet Newsgroups	62
	Ubuntu One Cloud Storage	64
	References	64
4	Productivity Applications	65
	Introducing LibreOffice	67
	Other Office Suites for Ubuntu	69
	Working with GNOME Office	69
	Working with KOffice	70
	Other Useful Productivity Software	71
	Working with PDF	71
	Working with XML and DocBook	71
	Working with LaTeX	73

Productivity Applications Written for Microsoft Windows	73
References	74
5 Multimedia Applications	75
Sound and Music	75
Sound Cards	76
Adjusting Volume	77
Sound Formats	78
Listening to Music	79
Buying Music in the Ubuntu One Music Store	81
Graphics Manipulation	83
The GNU Image Manipulation Program	83
Using Scanners in Ubuntu	85
Working with Graphics Formats	85
Capturing Screen Images	87
Using Digital Cameras with Ubuntu	88
Handheld Digital Cameras	88
Using Shotwell Photo Manager	88
Burning CDs and DVDs in Ubuntu	89
Creating CDs and DVDs with Brasero	89
Creating CDs from the Command Line	89
Creating DVDs from the Command Line	91
Viewing Video	94
TV and Video Hardware	94
Video Formats	95
Viewing Video in Linux	96
Personal Video Recorders	97
Video Editing	97
References	98
6 Other Ubuntu Interfaces	99
Desktop Environment	100
KDE and Kubuntu	101
Xfce and Xubuntu	102
LXDE and Lubuntu	103
GNOME3 and Gnobuntu	104
References	105

7	Games	107
	Ubuntu Gaming	107
	Installing Proprietary Video Drivers	108
	Installing Games in Ubuntu	109
	Warsow	110
	Scorched 3D	110
	Frozen Bubble	111
	SuperTux	112
	Battle for Wesnoth	112
	Frets on Fire	114
	FlightGear	114
	Speed Dreams	114
	Games for Kids	114
	Commercial Games	115
	Playing Windows Games	116
	References	116
Part III	System Administration	
8	Managing Software	119
	Ubuntu Software Center	119
	Using Synaptic for Software Management	120
	Staying Up-to-Date	122
	Working on the Command Line	123
	Day-to-Day Usage	124
	Finding Software	127
	Compiling Software from Source	128
	Compiling from a Tarball	128
	Compiling from Source from the Ubuntu Repositories	129
	Configuration Management	130
	dotdee	130
	OneConf	131
	References	131
9	Command-Line Quickstart	133
	What Is the Command Line?	134
	Accessing the Command Line	135
	Text-Based Console Login	136
	Logging Out	137
	Logging In and Out from a Remote Computer	137
	User Accounts	138

Reading Documentation	140
Using Man Pages	140
Using <code>apropos</code>	140
Using <code>whereis</code>	141
Understanding the Linux File System Hierarchy	141
Essential Commands in <code>/bin</code> and <code>/sbin</code>	142
Configuration Files in <code>/etc</code>	143
User Directories: <code>/home</code>	143
Using the Contents of the <code>/proc</code> Directory to Interact with the Kernel	144
Working with Shared Data in the <code>/usr</code> Directory	145
Temporary File Storage in the <code>/tmp</code> Directory	146
Accessing Variable Data Files in the <code>/var</code> Directory	146
Navigating the Linux File System	146
Listing the Contents of a Directory with <code>ls</code>	146
Changing Directories with <code>cd</code>	148
Finding Your Current Directory with <code>pwd</code>	149
Working with Permissions	149
Assigning Permissions	150
Directory Permissions	151
Altering File Permissions with <code>chmod</code>	152
File Permissions with <code>chgrp</code>	153
Changing File Permissions with <code>chown</code>	153
Understanding Set User ID and Set Group ID Permissions	153
Working with Files	155
Creating a File with <code>touch</code>	155
Creating a Directory with <code>mkdir</code>	155
Deleting a Directory with <code>rmdir</code>	156
Deleting a File or Directory with <code>rm</code>	157
Moving or Renaming a File with <code>mv</code>	157
Copying a File with <code>cp</code>	158
Displaying the Contents of a File with <code>cat</code>	159
Displaying the Contents of a File with <code>less</code>	159
Using Wildcards and Regular Expressions	159
Working as Root	160
Understanding and Fixing <code>sudo</code>	160
Creating Users	164
Deleting Users	164
Shutting Down the System	165
Rebooting the System	166
Commonly Used Commands and Programs	166
References	167

10	Command-Line Master Class	169
	Why Use the Command Line?	170
	Using Basic Commands	171
	Printing the Contents of a File with <code>cat</code>	172
	Changing Directories with <code>cd</code>	173
	Changing File Access Permissions with <code>chmod</code>	175
	Copying Files with <code>cp</code>	175
	Printing Disk Usage with <code>du</code>	176
	Finding Files by Searching with <code>find</code>	177
	Searches for a String in Input with <code>grep</code>	179
	Paging Through Output with <code>less</code>	180
	Creating Links Between Files with <code>ln</code>	182
	Finding Files from an Index with <code>locate</code>	184
	Listing Files in the Current Directory with <code>ls</code>	184
	Reading Manual Pages with <code>man</code>	186
	Making Directories with <code>mkdir</code>	187
	Moving Files with <code>mv</code>	187
	Listing Processes with <code>ps</code>	188
	Deleting Files and Directories with <code>rm</code>	188
	Printing the Last Lines of a File with <code>tail</code>	189
	Printing Resource Usage with <code>top</code>	189
	Printing the Location of a Command with <code>which</code>	191
	Redirecting Output and Input	191
	<code>stdin</code> , <code>stdout</code> , <code>stderr</code> , and Redirection	193
	Comparing Files	194
	Finding Differences in Files with <code>diff</code>	194
	Finding Similarities in Files with <code>comm</code>	194
	Combining Commands	195
	Using Environment Variables	197
	Using Common Text Editors	200
	Working with <code>nano</code>	201
	Working with <code>vi</code>	202
	Working with <code>emacs</code>	203
	Working with Compressed Files	204
	Using Multiple Terminals with <code>byobu</code>	205
	References	207
11	Managing Users	209
	User Accounts	209
	The Super User/Root User	210
	User IDs and Group IDs	212
	File Permissions	212

Managing Groups	213
Group Listing	213
Group Management Tools	214
Managing Users	216
User Management Tools	216
Adding New Users	218
Monitoring User Activity on the System	222
Managing Passwords	222
System Password Policy	222
The Password File	223
Shadow Passwords	224
Managing Password Security for Users	226
Changing Passwords in a Batch	227
Granting System Administrator Privileges to Regular Users	227
Temporarily Changing User Identity with the <code>su</code> Command	227
Granting Root Privileges on Occasion: The <code>sudo</code> Command	229
Disk Quotas	232
Implementing Quotas	233
Manually Configuring Quotas	233
Related Ubuntu Commands	234
References	235
12 Automating Tasks and Shell Scripting	237
Scheduling Tasks	237
Using <code>at</code> and <code>batch</code> to Schedule Tasks for Later	237
Using <code>cron</code> to Run Jobs Repeatedly	240
Basic Shell Control	242
The Shell Command Line	243
Shell Pattern-Matching Support	245
Redirecting Input and Output	246
Piping Data	247
Background Processing	247
Writing and Executing a Shell Script	248
Running the New Shell Program	249
Storing Shell Scripts for System-wide Access	250
Interpreting Shell Scripts Through Specific Shells	250
Using Variables in Shell Scripts	252
Assigning a Value to a Variable	252
Accessing Variable Values	253
Positional Parameters	253

- A Simple Example of a Positional Parameter 253
- Using Positional Parameters to Access and Retrieve
 - Variables from the Command Line 254
- Using a Simple Script to Automate Tasks 255
- Built-In Variables 257
- Special Characters 257
- Using Double Quotes to Resolve Variables in Strings
 - with Embedded Spaces 258
- Using Single Quotes to Maintain Unexpanded Variables 259
- Using the Backslash as an Escape Character 260
- Using the Backtick to Replace a String with Output 260
- Comparison of Expressions in `pdksh` and `bash` 261
- Comparing Expressions with `tcsh` 266
- The `for` Statement 270
- The `while` Statement 271
- The `until` Statement 273
- The `repeat` Statement (`tcsh`) 274
- The `select` Statement (`pdksh`) 274
- The `shift` Statement 275
- The `if` Statement 275
- The `case` Statement 276
- The `break` and `exit` Statements 278
- Using Functions in Shell Scripts 279
- References 280

- 13 The Boot Process 281**
- Running Services at Boot 281
- Beginning the Boot Loading Process 282
 - Loading the Linux Kernel 283
 - System Services and Runlevels 284
 - Runlevel Definitions 284
 - Booting into the Default Runlevel 285
 - Understanding `init` Scripts and the Final Stage
 - of Initialization 285
 - Controlling Services at Boot with Administrative Tools 286
 - Changing Runlevels 286
 - Troubleshooting Runlevel Problems 287
- Starting and Stopping Services Manually 288
- Using `Upstart` 289
- References 290

14	System-Monitoring Tools	291
	Console-Based Monitoring	291
	Using the <code>kill</code> Command to Control Processes	293
	Using Priority Scheduling and Control	294
	Displaying Free and Used Memory with <code>free</code>	296
	Disk Space	297
	Disk Quotas	298
	Graphical Process and System Management Tools	298
	System Monitor	298
	Conky	300
	Other	305
	KDE Process- and System-Monitoring Tools	305
	Enterprise Server Monitoring	305
	Landscape	306
	Other	306
	References	306
15	Backing Up	307
	Choosing a Backup Strategy	307
	Why Data Loss Occurs	308
	Assessing Your Backup Needs and Resources	309
	Evaluating Backup Strategies	311
	Making the Choice	314
	Choosing Backup Hardware and Media	314
	Removable Storage Media	314
	CD-RW and DVD+RW/-RW Drives	315
	Network Storage	315
	Tape Drive Backup	315
	Cloud Storage	316
	Using Backup Software	316
	<code>tar</code> : The Most Basic Backup Tool	317
	The GNOME File Roller	319
	The KDE <code>ark</code> Archiving Tool	320
	Déjà Dup	320
	Back In Time	322
	Unison	324
	Using the Amanda Backup Application	324
	Alternative Backup Software	325

Copying Files	326
Copying Files Using <code>tar</code>	326
Compressing, Encrypting, and Sending <code>tar</code> Streams	327
Copying Files Using <code>cp</code>	327
Copying Files Using <code>mc</code>	328
Using <code>rsync</code>	328
Version Control for Configuration Files	330
System Rescue	332
The Ubuntu Rescue Disc	333
Restoring the GRUB2 Boot Loader	333
Saving Files from a Nonbooting Hard Drive	333
References	334
16 Networking	335
Laying the Foundation: The <code>localhost</code> Interface	336
Checking for the Availability of the Loopback Interface	336
Configuring the Loopback Interface Manually	336
Checking Connections with <code>ping</code> , <code>traceroute</code> , and <code>mtr</code>	338
Networking with TCP/IP	340
TCP/IP Addressing	341
Using IP Masquerading in Ubuntu	343
Ports	344
IPv6 Basics	344
Network Organization	347
Subnetting	347
Subnet Masks	348
Broadcast, Unicast, and Multicast Addressing	348
Hardware Devices for Networking	349
Network Interface Cards	349
Network Cable	351
Hubs and Switches	352
Routers and Bridges	353
Initializing New Network Hardware	353
Using Network Configuration Tools	355
Command-Line Network Interface Configuration	356
Network Configuration Files	360
Using Graphical Configuration Tools	363
Dynamic Host Configuration Protocol	365
How DHCP Works	365
Activating DHCP at Installation and Boot Time	366
DHCP Software Installation and Configuration	367

Using DHCP to Configure Network Hosts	369
Other Uses for DHCP	371
Wireless Networking	371
Support for Wireless Networking in Ubuntu	371
Advantages of Wireless Networking	373
Choosing from Among Available Wireless Protocols	373
Beyond the Network and onto the Internet	374
Common Configuration Information	374
Configuring Digital Subscriber Line Access	376
Understanding PPP over Ethernet	376
Configuring a PPPoE Connection Manually	377
Configuring Dial-Up Internet Access	378
Troubleshooting Connection Problems	379
References	380
17 Remote Access with SSH and Telnet	381
Setting Up a Telnet Server	381
Telnet Versus SSH	383
Setting Up an SSH Server	383
SSH Tools	383
Using <code>scp</code> to Copy Individual Files Between Machines	384
Using <code>sftp</code> to Copy Many Files Between Machines	385
Using <code>ssh-keygen</code> to Enable Key-Based Logins	385
Virtual Network Computing	387
References	389
18 Securing Your Machines	391
Understanding Computer Attacks	391
Assessing Your Vulnerability	393
Protecting Your Machine	394
Securing a Wireless Network	395
Passwords and Physical Security	395
Configuring and Using Tripwire	396
Devices	397
Viruses	397
Configuring Your Firewall	398
AppArmor	401
Forming a Disaster Recovery Plan	403
References	404

19	Performance Tuning	405
	Hard Disk	405
	Using the BIOS and Kernel to Tune the Disk Drives	406
	The <code>hdparm</code> Command	407
	File System Tuning	408
	The <code>tune2fs</code> Command	408
	The <code>e2fsck</code> Command	409
	The <code>badblocks</code> Command	409
	Disabling File Access Time	409
	Kernel	410
	Apache	411
	MySQL	412
	Measuring Key Buffer Usage	412
	Using the Query Cache	414
	Miscellaneous Tweaks	415
	Query Optimization	416
	References	416
20	Kernel and Module Management	417
	The Linux Kernel	418
	The Linux Source Tree	419
	Types of Kernels	421
	Managing Modules	422
	When to Recompile	424
	Kernel Versions	425
	Obtaining the Kernel Sources	426
	Patching the Kernel	426
	Compiling the Kernel	428
	Using <code>xconfig</code> to Configure the Kernel	431
	Creating an Initial RAM Disk Image	434
	When Something Goes Wrong	435
	Errors During Compile	435
	Runtime Errors, Boot Loader Problems, and Kernel Oops	436
	References	436
Part IV	Ubuntu as a Server	
21	Sharing Files and Printers	439
	Using the Network File System	440
	Installing and Starting or Stopping NFS	440
	NFS Server Configuration	440
	NFS Client Configuration	442

Putting Samba to Work	443
Manually Configuring Samba with <code>/etc/samba/smb.conf</code>	444
Testing Samba with the <code>testparm</code> Command	447
Starting, Stopping, and Restarting the <code>smbd</code> Daemon	448
Mounting Samba Shares	449
Configuring Samba Using SWAT	450
Network and Remote Printing with Ubuntu	453
Creating Network Printers	454
Using the Common UNIX Printing System GUI	456
Avoiding Printer Support Problems	458
References	460
22 Apache Web Server Management	461
About the Apache Web Server	461
Installing the Apache Server	462
Installing from the Ubuntu Repositories	463
Building the Source Yourself	464
Starting and Stopping Apache	467
Starting the Apache Server Manually	467
Using <code>/etc/init.d/apache2</code>	468
Runtime Server Configuration Settings	469
Runtime Configuration Directives	470
Editing <code>apache2.conf</code>	470
Apache Multiprocessing Modules	473
Using <code>.htaccess</code> Configuration Files	473
File System Authentication and Access Control	475
Restricting Access with <code>allow</code> and <code>deny</code>	476
Authentication	477
Final Words on Access Control	479
Apache Modules	480
<code>mod_access</code>	481
<code>mod_alias</code>	481
<code>mod_asis</code>	481
<code>mod_auth</code>	482
<code>mod_auth_anon</code>	482
<code>mod_auth_dbm</code>	482
<code>mod_auth_digest</code>	482
<code>mod_autoindex</code>	483
<code>mod_cgi</code>	483
<code>mod_dir</code> and <code>mod_env</code>	483
<code>mod_expires</code>	483
<code>mod_headers</code>	483

- mod_include..... 484
 - mod_info and mod_log_config..... 484
 - mod_mime and mod_mime_magic..... 484
 - mod_negotiation..... 484
 - mod_proxy..... 484
 - mod_rewrite..... 484
 - mod_setenvif..... 485
 - mod_speling..... 485
 - mod_status..... 485
 - mod_ssl..... 485
 - mod_unique_id..... 485
 - mod_userdir..... 485
 - mod_usertrack..... 485
 - mod_vhost_alias..... 485
 - Virtual Hosting..... 486
 - Address-Based Virtual Hosts..... 486
 - Name-Based Virtual Hosts..... 486
 - Logging..... 488
 - References..... 490
- 23 Other HTTP Servers 491**
 - Nginx..... 491
 - lighttpd..... 493
 - Yaws..... 494
 - Cherokee..... 494
 - Jetty..... 495
 - thttpd..... 495
 - Apache Tomcat..... 496
 - References..... 496
- 24 Remote File Serving with FTP 497**
 - Choosing an FTP Server..... 497
 - Choosing an Authenticated or Anonymous Server..... 498
 - Ubuntu FTP Server Packages..... 498
 - Other FTP Servers..... 498
 - Installing FTP Software..... 499
 - The FTP User..... 500
 - Configuring the Very Secure FTP Server..... 502
 - Controlling Anonymous Access..... 503
 - Other vsftpd Server Configuration Files..... 504
 - Using the ftpchosts File to Allow or Deny FTP Server Connection..... 505
 - References..... 506

25	Handling Email	507
	How Email Is Sent and Received	507
	The Mail Transport Agent	508
	Choosing an MTA	510
	The Mail Delivery Agent	510
	The Mail User Agent	511
	Basic Postfix Configuration and Operation	512
	Configuring Masquerading	514
	Using Smart Hosts	515
	Setting Message Delivery Intervals	515
	Mail Relaying	516
	Forwarding Email with Aliases	516
	Using Fetchmail to Retrieve Mail	517
	Installing Fetchmail	517
	Configuring Fetchmail	517
	Choosing a Mail Delivery Agent	521
	Procmail	521
	Spamassassin	521
	Squirrelmail	522
	Virus Scanners	522
	Autoresponders	522
	Alternatives to Microsoft Exchange Server	522
	Microsoft Exchange Server/Outlook Client	523
	CommuniGate Pro	523
	Oracle Beehive	524
	Bynari	524
	Open-Xchange	524
	phpgroupware	524
	PHProjekt	524
	Horde	524
	References	525
26	Proxying, Reverse Proxying, and Virtual Private Networks (VPN)	527
	What Is a Proxy Server?	527
	Installing Squid	528
	Configuring Clients	528
	Access Control Lists	529
	Specifying Client IP Addresses	533
	Sample Configurations	534
	Virtual Private Networks (VPN)	536
	Setting Up a VPN Client	537
	Setting Up a VPN Server	539
	References	541

27	Administering Relational Database Services	543
	A Brief Review of Database Basics	544
	How Relational Databases Work	545
	Understanding SQL Basics	547
	Creating Tables	548
	Inserting Data into Tables	549
	Retrieving Data from a Database	550
	Choosing a Database: MySQL Versus PostgreSQL	552
	Speed	552
	Data Locking	552
	ACID Compliance in Transaction Processing to Protect Data Integrity	553
	SQL Subqueries	554
	Procedural Languages and Triggers	554
	Configuring MySQL	554
	Setting a Password for the MySQL Root User	555
	Creating a Database in MySQL	556
	Configuring PostgreSQL	558
	Initializing the Data Directory in PostgreSQL	558
	Creating a Database in PostgreSQL	559
	Creating Database Users in PostgreSQL	559
	Deleting Database Users in PostgreSQL	560
	Granting and Revoking Privileges in PostgreSQL	561
	Database Clients	561
	SSH Access to a Database	562
	Local GUI Client Access to a Database	563
	Web Access to a Database	563
	The MySQL Command-Line Client	564
	The PostgreSQL Command-Line Client	566
	Graphical Clients	566
	References	567
28	NoSQL Databases	569
	Key/Value Stores	571
	Berkeley DB	572
	Cassandra	572
	Memcached and MemcacheDB	573
	Redis	573
	Riak	574
	Document Stores	574
	CouchDB	575
	MongoDB	575
	BaseX	576

Wide Column Stores	576
BigTable	577
HBase	577
Graph Stores	577
Neo4j	578
OrientDB	578
HyperGraphDB	578
FlockDB	578
References	579
29 Lightweight Directory Access Protocol (LDAP)	581
Configuring the Server	582
Creating Your Schema	582
Populating Your Directory	584
Configuring Clients	586
Evolution	586
Thunderbird	587
Administration	587
References	588
30 Linux Terminal Server Project (LTSP)	589
Requirements	590
Installation	593
Using LTSP	594
References	595
31 Virtualization on Ubuntu	597
KVM	599
VirtualBox	603
VMware	605
Xen	605
References	605
32 Ubuntu in the Cloud	607
Why a Cloud?	608
Software as a Service (SaaS)	609
Platform as a Service (PaaS)	609
Infrastructure as a Service (IaaS)	609
Metal as a Service (MaaS)	609
Before You Do Anything	610

- Ubuntu Cloud and Eucalyptus 610
 - Deploy/Install Basics: Public or Private? 612
 - Public 612
 - Private 613
 - A euca2ools Primer 616
- Ubuntu Cloud and OpenStack 618
 - Compute Infrastructure (Nova) 618
 - Storage Infrastructure (Swift) 619
 - Imaging Service (Glance) 619
 - Installation 619
 - Creating an Image 629
 - Instance Management 632
 - Storage Management 633
 - Network Management 633
 - An OpenStack Commands Primer 634
 - Learning More 634
- juju 634
 - Getting Started 635
 - Charms 638
- Landscape 640
- References 640

33 Managing Sets of Servers 641

- juju 641
- Puppet 642
- Chef 642
- CFEngine 643
- Landscape 643
- References 643

Part V Programming Linux

34 Opportunistic Development 645

- Version Control Systems 646
 - Managing Software Projects with Subversion 646
 - Managing Software Projects with Bazaar 647
 - Managing Software Projects with Mercurial 648
 - Managing Software Projects with Git 649
- Introduction to Opportunistic Development 650
- Launchpad 651
- Quickly 653

Ground Control	657
Bikeshed and Other Tools	661
References	663
35 Helping with Ubuntu Development	665
Introduction to Ubuntu Development	666
Setting Up Your Development System	667
Install Basic Packages and Configure	667
Create a Launchpad Account	668
Set Up Your Environment to Work with Launchpad	668
Fixing Bugs and Packaging	670
Finding Bugs to Fix with Harvest	673
Masters of the Universe	673
References	673
36 Helping with Ubuntu Testing and QA	675
Community Teams	675
Ubuntu Testing Team	676
QA Team	676
Bug Squad	677
Test Drive	677
References	680
37 Using Perl	681
Using Perl with Linux	681
Perl Versions	682
A Simple Perl Program	682
Perl Variables and Data Structures	684
Perl Variable Types	685
Special Variables	685
Operators	686
Comparison Operators	686
Compound Operators	687
Arithmetic Operators	687
Other Operators	688
Special String Constants	688
Conditional Statements: <code>if/else</code> and <code>unless</code>	689
<code>if</code>	689
<code>unless</code>	690
Looping	690
<code>for</code>	690
<code>foreach</code>	691

while.....	691
until.....	692
last and next.....	692
do ... while and do ... until.....	692
Regular Expressions.....	693
Access to the Shell.....	694
Modules and CPAN.....	695
Code Examples.....	695
Sending Mail.....	695
Purging Logs.....	697
Posting to Usenet.....	698
One-Liners.....	699
Command-Line Processing.....	700
References.....	700
38 Using Python.....	703
Python on Linux.....	704
The Basics of Python.....	705
Numbers.....	705
More on Strings.....	707
Lists.....	710
Dictionaries.....	712
Conditionals and Looping.....	713
Functions.....	715
Object Orientation.....	716
Class and Object Variables.....	717
Constructors and Destructors.....	718
Class Inheritance.....	719
The Standard Library and the Python Package Index.....	721
References.....	721
39 Using PHP.....	723
Introduction to PHP.....	724
Entering and Exiting PHP Mode.....	724
Variables.....	724
Arrays.....	726
Constants.....	728
References.....	728
Comments.....	729
Escape Sequences.....	729
Variable Substitution.....	730
Operators.....	731

Conditional Statements	733
Special Operators	734
Switching	735
Loops	737
Including Other Files	739
Basic Functions	740
Strings	740
Arrays	743
Files	745
Miscellaneous	747
Handling HTML Forms	751
Databases	751
References	754
40 C/C++ Programming Tools for Ubuntu	755
Programming in C with Linux	756
Using the C Programming Project Management	
Tools Provided with Ubuntu	757
Building Programs with <code>make</code>	757
Using Makefiles	757
Using the <code>autoconf</code> Utility to Configure Code	759
Debugging Tools	760
Using the GNU C Compiler	761
Graphical Development Tools	762
Using the KDevelop Client	762
The Glade Client for Developing in GNOME	763
References	764
41 Using Other Popular Programming Languages	767
Ada	768
Clojure	768
COBOL	769
Erlang	770
Forth	770
Go	771
Fortran	771
Groovy	771
Haskell	772
Java	772
JavaScript	772
Lisp	773
Lua	773

Mono	774
Ruby	774
Rust	775
Scala	775
Scratch	776
Vala	776
References	776
42 Beginning Mobile Development for Android	779
Introduction to Android	780
Hardware	780
Linux Kernel	780
Libraries	780
Android Runtime	780
Application Framework	780
Applications	781
Installing the Android SDK	781
Install Java	781
Install Eclipse	781
Install the SDK	781
Install the ADT Eclipse Plug-In	782
Install Other Components	782
Install Virtual Devices	783
Create Your First Application	784
References	785
Part VI Appendices	
A Ubuntu Under the Hood	787
What Is Linux?	787
Why Use Linux?	788
What Is Ubuntu?	790
Ubuntu for Business	791
Ubuntu in Your Home	792
64-Bit Ubuntu	793
Getting the Most from Ubuntu and Linux Documentation	793
Ubuntu Developers and Documentation	795
References	795

B	Ubuntu and Linux Internet Resources	797
	Websites and Search Engines	798
	Web Search Tips	798
	Google Is Your Friend	799
	Ubuntu Package Listings	799
	Certification	799
	Commercial Support	800
	Documentation	800
	Linux Guides	800
	Ubuntu	801
	Mini-CD Linux Distributions	801
	Various Intel-Based Linux Distributions	802
	PowerPC-Based Linux Distributions	802
	Linux on Laptops and PDAs	802
	The X Window System	803
	Usenet Newsgroups	803
	Mailing Lists	804
	Ubuntu Project Mailing Lists	805
	Internet Relay Chat	805
	Index	807

About the Authors

Matthew Helmke is an active member of the Ubuntu community. He served from 2006 to 2011 on the Ubuntu Forum Council, providing leadership and oversight of the Ubuntu Forums (www.ubuntuforums.org), and spent two years on the Ubuntu regional membership approval board for Europe, the Middle East, and Africa. He has written about Ubuntu for several magazines and websites, is a lead author of *The Official Ubuntu Book*, and coauthored *The VMware Cookbook*. He works as a senior technical writer for Pearson North America's Assessment and Information division, documenting assessment software. Matthew first used Unix in 1987 while studying LISP on a Vax at the university. He has run a business using only free and open source software, has consulted, and has a master's degree in Information Resources and Library Science from the University of Arizona. You can find out more about Matthew at matthewhelmke.com or drop him a line with errata or suggestions at matthew@matthewhelmke.com.

Andrew Hudson is a freelance journalist who specializes in writing about Linux. He has significant experience in Red Hat and Debian-based Linux distributions and deployments and can often be found sitting at his keyboard tweaking various settings and config files just for the hell of it. He lives in Wiltshire, which is a county of England, along with his wife, Bernice, and their son, John. Andrew does not like Emacs. He can be reached at andy.hudson@gmail.com.

Paul Hudson is a recognized expert in open-source technologies. He is also a professional developer and full-time journalist for Future Publishing. His articles have appeared in *MacFormat*, *PC Answers*, *PC Format*, *PC Plus*, and *Linux Format*. Paul is passionate about free software in all its forms and uses a mix of Linux and BSD to power his desktops and servers. Paul likes Emacs. Paul can be contacted through <http://hudzilla.org>.

Dedication

To Heather, Saralyn, Sedona, and Philip—the most amazing family a guy could hope for; to my grandfather for always believing in me and teaching me to believe in myself; and to my friends in the Ubuntu, developer, sysadmin, cloud computing, and DevOps communities.

Acknowledgments

I am solely responsible for this edition of Ubuntu Unleashed, but I freely acknowledge that I am standing on the shoulders of giants. I want to express my gratitude to Andrew and Paul Hudson for the solid foundation that past editions of the book (up to Ubuntu Unleashed, 2008 Edition) provided to this update. Thanks to Ryan Troy for helping with the 2010 edition. Thank you to the many people who helped with technical edits and both formal and informal advice, especially Shannon Oliver, Chris Johnston, and Daniel Holbach for their help with this edition. I owe a huge debt of gratitude to the Ubuntu community, Canonical, and Mark Shuttleworth for inviting me to participate in the community, including my role in the forums, a turn on the EMEA membership board, and two Ubuntu Developer Summits. Thanks to the Ubuntu All Stars for the chance to jam with you on guitar. Thank you to the entire Ubuntu community for your labor of love to create this wonderful operating system. Finally, thanks to my colleagues at Pearson, especially Debra Williams Cauley, for the trust placed in me and the opportunity to collaborate on projects like this one.

We Want to Hear from You!

As the reader of this book, *you* are our most important critic and commentator. We value your opinion and want to know what we're doing right, what we could do better, what areas you'd like to see us publish in, and any other words of wisdom you're willing to pass our way.

We welcome your comments. You can email or write to let us know what you did or didn't like about this book—as well as what we can do to make our books better.

Please note that we cannot help you with technical problems related to the topic of this book.

When you write, please be sure to include this book's title and author as well as your name and email address. We will carefully review your comments and share them with the author and editors who worked on the book.

Email: errata@informit.com

Mail: Addison-Wesley/Prentice Hall Publishing
ATTN: Reader Feedback
1330 Avenue of the Americas
35th Floor
New York, New York, 10019

Reader Services

Visit our website and register this book at informit.com/register for convenient access to any updates, downloads, or errata that might be available for this book.

Introduction

We are pleased to present the 2013 edition of *Ubuntu Unleashed*. Ubuntu is a Linux-based computer operating system that has taken the world by storm. From its humble beginning in 2004, Ubuntu has risen to be the vanguard of desktop Linux, as well as a popular choice for servers.

Ubuntu descends from one of the oldest and most revered Linux distributions, Debian. Debian is assembled by a team of talented volunteers, is one of the most stable and customizable distributions of Linux, and is well respected for its quality and technological prowess. It is, however, an operating system for geeks; the bar for entry into the Debian realm is set high, and its user base tends to be highly proficient and expects new users to learn the ropes before joining in. That is both appropriate and okay.

What Ubuntu has done is leverage the quality of Debian to create an operating system that ordinary people can use. That doesn't mean that Ubuntu users are not technologically proficient, just that they do not have to be. In fact, many talented and respected software developers love Ubuntu because it enables them to concentrate on their specific interests instead of the details of the operating system. This book is for these people and for those who aspire to join their ranks.

If you are new to Linux, you have made a great decision by choosing this book. Sams Publishing's *Unleashed* books offer an in-depth look at their subjects, taking in both beginner and advanced users and moving them to a new level of knowledge and expertise. Ubuntu is a fast-changing distribution that has an updated release twice a year. We have tracked the development of Ubuntu from early on to make sure that the information in this book mirrors closely

IN THIS INTRODUCTION

- ▶ Licensing
- ▶ Who This Book Is For
- ▶ What This Book Contains
- ▶ Conventions Used in This Book

the development of the distribution. A full copy of Ubuntu is included on the enclosed disc, and it is possible for you to install Ubuntu from that disc in less than an hour!

A QUICK WORD ABOUT MARKETING

Almost all of the content in this book applies regardless of what Ubuntu release version you are using, so long as it is reasonably current. The book has been written to try to focus on information that is useful for the longest amount of time possible. Some chapters, like those covering installation or the basics of the default Ubuntu graphical user interface, will have their information change frequently. Those chapters are the exception. The blurb on the cover of the book about which editions this book covers was added to account for these chapters and to denote clearly when the book was most recently revised. The note about a free upgrade to 13.04 is true in that we will send an upgrade kit to anyone who has purchased and registered the book. The kit includes install media for the 13.04 release and a supplemental guide to the changes from 12.10 to 13.04.

Do not let the reputation of Linux discourage you, however. Many people who have heard of Linux think that it is found only on servers, looking after websites and email. Nothing could be further from the truth. Distributions like Ubuntu are making huge inroads in to the desktop market. Corporations are realizing the benefits of running a stable and powerful operating system that is easy to maintain and easy to secure. The best part is that as Linux distributions make improvements, the majority of those improvements are shared freely, allowing you to benefit from the additions and refinements made by one distribution, such as Red Hat, while continuing to use a different distribution, such as Ubuntu, which in turn shares its improvements. You can put Ubuntu to work today and be assured of a great user experience. Feel free to make as many copies of the software as you want; Ubuntu is freely and legally distributable all over the world—no copyright lawyers are going to pound on your door.

Licensing

Software licensing is an important issue for all computer users and can entail moral, legal, and financial considerations. Many consumers think that purchasing a copy of a commercial or proprietary operating system, productivity application, utility, or game conveys ownership, but this is not true. In the majority of cases, the *end user license agreement (EULA)* included with a commercial software package states that you have paid only for the right to use the software according to specific terms. This generally means you may not examine, make copies, share, resell, or transfer ownership of the software package. More onerous software licenses enforce terms that preclude you from distributing or publishing comparative performance reviews of the software. Even more insidious licensing schemes (and supporting legislation, especially in the United States) contain provisions allowing onsite auditing of the software's use!

This is not the case with the software included with this book. You are entirely free to make copies, share copies, and install the software on as many computers as you want—we encourage you to purchase additional copies of this book to give as gifts, however. Be sure to read the README file on the disc included with this book for important

information regarding the included software and disk contents. After you install Ubuntu, go to www.gnu.org/licenses/gpl.html to find a copy of the GNU GPL. You will see that the GPL provides unrestricted freedom to use, duplicate, share, study, modify, improve, and even sell the software.

You can put your copy of Ubuntu to work right away in your home or at your place of business without worrying about software licensing, per-seat workstation or client licenses, software auditing, royalty payments, or any other type of payments to third parties. However, be aware that although much of the software included with Ubuntu is licensed under the GPL, some packages on this book's disc are licensed under other terms. There is a variety of related software licenses, and many software packages fall under a broad definition known as *open source*. Some of these include the Artistic License, the BSD License, the Mozilla Public License, and the Q Public License.

For additional information about the various GNU software licenses, browse to www.gnu.org/. For a definition of open-source and licensing guidelines, along with links to the terms of nearly three dozen open-source licenses, browse to www.opensource.org/.

Who This Book Is For

Those Wanting to Become Intermediate or Advanced Users

Ubuntu Unleashed is intended for intermediate and advanced users or those who want to become one. Our goal is to give you a nudge in the right direction, to help you enter the higher stages by exposing you to as many different tools and ideas as possible; we want to give you some thoughts and methods to consider and spur you on to seek out more. Although the contents are aimed at intermediate to advanced users, new users who pay attention will benefit from the advice, tips, tricks, traps, and techniques presented in each chapter. Pointers to more detailed or related information are also provided at the end of each chapter.

If you are new to Linux, you might need to learn some new computer skills, such as how to research your computer's hardware, how to partition a hard drive, and (occasionally) how to use a command line. This book helps you learn these skills and shows you how to learn more about your computer, Linux, and the software included with Ubuntu. Most important, it helps you overcome your fear of the system by telling you more about what it is and how it works.

We would like to take a moment to introduce a concept called "The Three Levels of Listening" from Alistair Cockburn's *Agile Software Development*, published by Addison Wesley. These describe how a person learns and masters a technique. We all start at the first stage and progress from there. Few reach the last stage, but those who do are incredibly effective and efficient. People aiming for this stage are the very ones for whom we intend this book.

- ▶ **Following**—The stage where the learner looks for one very detailed process that works and sticks to it to accomplish a task.

- ▶ **Detaching**—The stage where the learner feels comfortable with one method and begins to learn other ways to accomplish the same task.
- ▶ **Fluent**—The stage where the learner has experience with or understanding of many methods and doesn't think of any of them in particular while doing a task.

Myriad books focus on the first set of users. This is not one of them. It is our goal in *Ubuntu Unleashed* to write just enough to be sufficient to get you from where you are to where you want or need to be. This is not a book for newcomers who want or need every step outlined in detail, although we do that occasionally. This is a book for people who want help learning about what can be done and a way to get started doing it. The Internet is an amazing reference tool, so this is not a comprehensive reference book. This book is a tool to help you see the landscape; to learn enough about what you seek to get you started in the right direction with a quality foundational understanding.

Sysadmins, Programmers, and DevOps

Systems administrators, or Sysadmins, are the people who keep servers and networks up and running. Their role is sometimes called *operations*. They deal with software installation and configuration, security, and do all the amazing things behind the scenes that let others use these systems for their work. They are often given less respect than they deserve, but the pay is good and it is a ton of fun to wield the ultimate power over a computer system. It is also a great responsibility, and these amazing guys and gals work hard to make sure they do their jobs well, striving for incredible system uptime and availability. Ubuntu is an excellent operating system for servers and networks, and in this book you can find much of the knowledge needed to get started in this role.

Programmers are the people who write software. They are sometimes called *developers*. Programmers work with others to create the applications that run on top of those systems. Ubuntu is a great platform for writing and testing software. This is true whether you are doing web application development or writing software for desktop or server systems. It also makes a great platform for learning new programming languages and trying out new ideas. This book can help you get started.

DevOps is a portmanteau of *developer* and *operations*. It signifies a blending of the two roles already described. The information technology (IT) world is changing, and roles are becoming less clear cut and isolated from one another. In the past, it was common to witness battles between programmers excited about new technology and sysadmins in love with stability. DevOps realizes that neither goal is healthy in isolation, but that seeking a balance between the two can yield great results by removing the barriers to communication and understanding that sometimes cause conflict within a team. Because of the rise of cloud computing and virtualization, which are also covered in this book, and more agile forms of development, DevOps is a useful perspective that enables people working in IT to do an even better job of serving their ultimate clients: end users. This book is a great foundation for those wanting to learn knowledge that will help with both roles, hopefully presented in a way that balances them nicely.

What This Book Contains

Ubuntu Unleashed is organized into six parts, described here. A disc containing the entire distribution is included so that you have everything you need to get started.

Part I, “Installation and Configuration” takes you through installing Ubuntu on your computer in the place of any other operating system you might be running, such as Windows.

Part II, “Desktop Ubuntu,” is aimed at users who want to use Ubuntu on desktop systems.

Part III, “System Administration,” covers both elementary and sophisticated details of setting up a system for specific tasks and maintaining that system.

Part IV, “Ubuntu as a Server,” gives you the information you need to start building your own file, web, and other servers for use in your home or office.

Part V, “Programming Linux,” provides a great introduction to how you can extend Ubuntu capabilities even further using the development tools supplied with it.

Part VII, “Appendices,” includes references that give you scope to explore in even more depth some of the topics covered in this book as well as historical context to Ubuntu and installation resources.

In addition to what has already been mentioned, after the spring release of Ubuntu, a bonus chapter will be available online at www.informit.com/title/9780672336249.

If you have the print copy of this book, follow the instructions on the inside back cover page to register your product, and you will receive a DVD of Ubuntu 13.04 and an upgrade kit with information about changes from 12.10.

The ebook edition does not provide a DVD of Ubuntu or the upgrade kit.

If you purchased the ebook edition, or if you don’t want to wait for a DVD, you can download the most current release of Ubuntu from www.ubuntu.com/download.

Conventions Used in This Book

It is impossible to cover every option of every command included in Ubuntu. Besides, with the rise of the Internet and high-speed connections, reference materials are far less valuable than they used to be because most of these details are only a quick Google search away. Instead, we focus on teaching you how to find information you need while giving a quality overview worthy of the intermediate or advanced user. Sometimes this book offers tables of various options, commands, and keystrokes to help condense, organize, and present information about a variety of subjects.

To help you better understand code listing examples and sample command lines, several formatting techniques are used to show input and ownership. For example, if the command or code listing example shows typed input, the input is formatted in boldface after the sample command prompt, as follows:

```
matthew@seymour:~$ ls
```

If typed input is required, as in response to a prompt, the sample typed input also is in boldface, like so:

```
Delete files? [Y/n] y
```

All statements, variables, and text that should appear on your display use the same boldface formatting. In addition, command lines that require root or super-user access are prefaced with the `sudo` command, as follows:

```
matthew@seymour:~$ sudo printtool &
```

The following elements provide you with useful tidbits of information that relate to the discussion of the text:

NOTE

A note provides additional information you might find useful as you are working. Notes augment a discussion with ancillary details or point you to an article, a whitepaper, or another online reference for more information about a specific topic.

TIP

A tip contains a special insight or a timesaving technique, as well as information about items of particular interest to you that you might not find elsewhere.

CAUTION

A caution warns you about pitfalls or problems before you run a command, edit a configuration file, or choose a setting when administering your system.

SIDEBARS CAN BE GOLDMINES

Just because it is in a sidebar does not mean that you will not find something new here. Be sure to watch for these elements that bring in outside content that is an aside to the discussion in the text. You will read about other technologies, Linux-based hardware, and special procedures to make your system more robust and efficient.

Other formatting techniques include the use of italic for placeholders in computer command syntax. Computer terms or concepts are also italicized upon first introduction in text.

Finally, you should know that all text, sample code, and screenshots in *Ubuntu Unleashed* were developed using Ubuntu and open-source tools.

Read on to start learning about and using the latest version of Ubuntu.

CHAPTER 9

Command-Line Quickstart

The Linux command line is one of the most powerful tools available for computer system administration and maintenance. The command line is also known as the terminal, shell, the console, the command prompt, and the *command-line interface (CLI)*. For the purposes of this chapter and the next, these terms are interchangeable, although fine-grained differences do exist between them.

The command line is an efficient way to perform complex tasks accurately and much more easily than it would seem at a first glance. Knowledge of the commands available to you and also how to string them together makes using Ubuntu easier for many tasks. Many of the commands were created by the GNU Project as free software analogs to previously existing proprietary UNIX commands. If you are interested, you can learn more about the GNU Project at www.gnu.org/gnu/thegnuproject.html.

This chapter covers some of the basic commands that you need to know to be productive at the command line. You find out how to get to the command line and discover some of the commands used to navigate the file system and perform basic operations with files, directories, and users. This chapter does not give comprehensive coverage of all the commands discussed, but it does give you enough to get started. Chapter 10, “Command-Line Master Class,” advances the subject further while expanding on some of the commands from this chapter. The skills you discover in this chapter help you get started using the command line with confidence.

IN THIS CHAPTER

- ▶ What Is the Command Line?
- ▶ Accessing the Command Line
- ▶ User Accounts
- ▶ Reading Documentation
- ▶ Understanding the Linux File System Hierarchy
- ▶ Navigating the Linux File System
- ▶ Working with Permissions
- ▶ Working with Files
- ▶ Working as Root
- ▶ Commonly Used Commands and Programs
- ▶ References

What Is the Command Line?

If you spend any amount of time with experienced Linux users, you have heard them mention the command line. Some, especially those who have begun their journey in the Linux world using distributions that make it easy to complete many tasks using a *graphical user interface (GUI)*, such as Ubuntu, might speak with trepidation about the mysteries of the text interface. Others either praise its power or comment about doing something via the command line as if it is the most natural and obvious way to complete a task.

It is not necessary for you to embrace either extreme. You might develop an affinity for the command line when performing some tasks and prefer the GUI for others. This is where most users end up today. Some might say that you will never need to access the command line because Ubuntu offers a slew of graphical tools that enable you to configure most things on your system. Although the premise might be true most of the time, there are some good reasons to acquire a fundamental level of comfort with the command line that you should consider before embracing that view.

Sometimes things go wrong, and you might not have the luxury of a graphical interface to work with. In these situations, a fundamental understanding of the command line and its uses can be a real lifesaver. Also, some tasks end up being far easier and faster to accomplish from the command line. More important, though, you will be able to make your way around a command-line-based system, which you will encounter if you ever work with a Linux server because most Linux servers have no GUI, and all administration is done using a command-line interface.

NOTE

Don't be tempted to skip over this chapter as irrelevant. You should take the time to work through the chapter and ensure that you are comfortable with the command line before moving on. Doing so will benefit you greatly for years to come.

Initially, you might be tempted to think of the command line as the product of some sort of black and arcane art, and in some ways it can appear to be extremely difficult and complicated to use. However, with a little perseverance, by the end of this chapter you will start to feel comfortable using the command line, and you'll be ready to move on to Chapter 10, "Command-Line Master Class."

This chapter introduces you to commands that enable you to perform the following:

- ▶ **Routine tasks**—Logging in and out, changing passwords, listing and navigating file directories
- ▶ **Basic file management**—Creating files and folders, copying or moving them around the file system, renaming and deleting them
- ▶ **Basic system management**—Shutting down or rebooting, changing file permissions, and reading man pages, which are entries for commands included as files already on your computer in a standardized manual format

The information in this chapter is valuable for individual users or system administrators who are new to Linux and are learning to use the command line for the first time.

TIP

Those of you who have used a computer for many years will probably have come into contact with MS-DOS, in which case being presented with a black screen will fill you with a sense of nostalgia. Don't get too comfy; the command line in Linux is different from (and actually more powerful than) its distant MS-DOS cousin. Even cooler is that whereas MS-DOS skills are transferable only to other MS-DOS environments, the skills that you learn at the Linux command line can be transferred easily to other UNIX and UNIX-like operating systems, such as Solaris, OpenBSD, FreeBSD, and even Mac OS X, which provides access to the terminal.

Accessing the Command Line

You can quickly access the terminal using the desktop menu option Terminal. This opens `gnome-terminal`, from which you can access the terminal while remaining in a GUI environment. This time, the terminal appears as white text on an aubergine (dark purple) background. This is the most common method for accessing the terminal for most desktop users.

NOTE

Finding and running programs, such as Terminal, from a GUI is covered in Chapter 2, "Working with Unity," as is logging it to a Linux system using a graphical interface. This chapter focuses on text-based logins and the use of Linux.

The second most common way for graphical desktop users to access the command line is to press the key combination `Ctrl+Alt+F1`, after which Ubuntu switches to a black screen and a login prompt like this:

```
Ubuntu 12.10 oneric seymour tty1
seymour login:
```

TIP

This is `tty1`, one of six virtual consoles that Ubuntu provides. After you have accessed a virtual console, you can use `Ctrl+Alt` + any of `F1` through `F6` to switch to a different console, `tty1` through `tty6`. If you want to get back to the graphical interface, press `Ctrl+Alt+F7`. You can also switch between consoles by holding the `Alt` key and pressing either the left or the right cursor key to move down or up a console, such as `tty1` to `tty2`.

Regardless of which way you access the terminal, using the virtual `tty` consoles accessible at `Ctrl+Alt+F1-6` or via the windowed version atop your GUI desktop, you will find the rest of the usage details that we cover work the same. As you continue to learn and

experiment beyond the contents of this book, you might start to discover some subtle differences between the two and develop a preference. For our purposes, either method works quite well.

There are many other ways to access and use the command line. You could use a traditional console with a monitor, keyboard, and mouse attached to the PC, but which boots into a command-line interface instead of a GUI. You can also connect to your system through a wired or wireless network using the `telnet` or `ssh` commands, as covered in Chapter 17, “Remote Access with SSH and Telnet.”

With that, let’s begin.

Text-Based Console Login

However you connect to a command-line interface, you start with a prompt similar to this one:

```
Ubuntu 12.10 oneric seymour ttyl
seymour login:
```

Your prompt might vary, depending on the version of Ubuntu you are using and the method you are using to connect. In any event, at this prompt, type in your username and press Enter. When you are prompted for your password, type it in and press Enter.

NOTE

Your password is not echoed back to you, which is a good idea. Why is it a good idea? Well, people are prevented from looking over your shoulder and seeing your screen input. It is not difficult to guess that a five-letter password might correspond to the user’s spouse’s first name. After typing your username and pressing the Enter key, you are asked for your password, which you type. Note that Ubuntu does not show any characters while you are typing your password in. This is a good thing because it prevents any shoulder surfers from seeing what you’ve typed or the length of the password.

Pressing the Enter key drops you to a shell prompt, signified by the dollar sign:

```
matthew@seymour:~$
```

This particular prompt tells me that I am logged in as the user `matthew` on the system `seymour` and I am currently in my home directory; Linux uses the tilde (`~`) as shorthand for the home directory, which would usually be something like `/home/matthew`.

TIP

Navigating through the system at the command line can get confusing at times, especially when a directory name occurs in several places. Fortunately, Linux includes a simple command that tells you exactly where you are in the file system. It’s easy to remember because the command is just an abbreviation of the present working directory, so type `pwd` at any point to get the full path of your location. For example, typing `pwd` after

following these instructions shows `/home/yourusername`, meaning that you are currently in your home directory.

Using the `pwd` command can save you a lot of frustration when you have changed directory half a dozen times and have lost track.

Logging Out

Use the `exit` or `logout` command or `Ctrl+D` to exit your session. You are then returned to the login prompt. If you use virtual consoles, remember to exit each console before leaving your PC. (Otherwise, someone could easily sit down and use your account.)

Logging In and Out from a Remote Computer

Although you can happily log in on your computer, an act known as a *local* login, you can also log in to your computer via a network connection from a remote computer. Linux-based operating systems provide a number of remote access commands you can use to log in to other computers on your *local area network (LAN)*, *wide area network (WAN)*, or the Internet. Note that you must have an account on the remote computer, *and* the remote computer must be configured to support remote logins; otherwise, you won't be able to log in.

NOTE

See Chapter 16, “Networking,” to see how to set up network interfaces with Linux to support remote network logins and Chapter 17, “Remote Access with SSH and Telnet,” to see how to start remote access services (such as `sshd`).

The best and most secure way to log in to a remote Linux computer is to use `ssh`, the Secure Shell client. Your login and session are encrypted while you work on the remote computer. The `ssh` client features many command-line options but can be simply used with the name or IP address of the remote computer, as follows:

```
matthew@seymour:~$ ssh 192.168.0.41
The authenticity of host '192.168.0.41 (192.168.0.41)' can't be established.
RSA key fingerprint is e1:db:6c:da:3f:fc:56:1b:52:f9:94:e0:d1:1d:31:50.
Are you sure you want to continue connecting (yes/no)?
```

yes

The first time you connect with a remote computer using `ssh`, Linux displays the remote computer's encrypted identity key and asks you to verify the connection. After you type `yes` and press `Enter`, you are warned that the remote computer's identity (key) has been entered in a file named `known_hosts` under the `.ssh` directory in your home directory. You are also prompted to enter your password:

```
Warning: Permanently added '192.168.0.41' (RSA) \
to the list of known hosts.
matthew@192.168.0.41's password:
matthew@babbage~$
```

After entering your password, you can work on the remote computer, which you can confirm by noticing the changed prompt that now uses the name of the remote computer on which you are working. Again, because you are using `ssh`, everything you enter on the keyboard in communication with the remote computer is encrypted. When you log out, you return to the shell on your computer:

```
matthew@babbage~$ logout
matthew@seymour:~$
```

User Accounts

A good place to start is with the concept of user-based security. For the most part, only two types of people access the system as users. (Although there are other accounts that run programs and processes, here we are talking about accounts that represent human beings rather than something like an account created for a web server process.) Most people have a regular user account. These users can change anything that is specific to their accounts, such as the wallpaper on the desktop, their personal preferences, and the configuration for a program when it is run by them using their account. Note that the emphasis is on anything that is *specific to their accounts*. This type of user cannot make systemwide changes that could affect other users.

To make systemwide changes, you need to use super user privileges, such as can be done using the account you created when you started Ubuntu for the first time (see Chapter 1, “Installing Ubuntu and Post-Installation Configuration”). With super user privileges you have access to the entire system and can carry out any task, even destructive ones. To help prevent this from happening, this user does not run with these powers enabled at all times, but instead spends most of the time as a regular user.

To use super user privileges from the command line, you need to preface the command you want to execute with another command, `sudo`, followed by a space and the command you want to run. As a mnemonic device, some think of this as “super user do.” When you press Enter (after typing the remaining command), you are prompted for your password, which you should type and then press the Enter key. As usual on any UNIX-based system, the password does not appear on the screen while you are typing it as a security measure, in case someone is watching over your shoulder. Ubuntu then carries out the command, but with super user privileges.

An example of the destructive nature of working as the super user is the age-old example `sudo rm -rf /`, which erases everything on your hard drive. If you enter a command using `sudo` as a regular user who does not have an account with super user privileges, an error message appears and nothing happens because the command will not run. We

recommend that you don't try this particular command as a test, though. If you enter this command using an account with super user privileges, you will soon find yourself starting over with a fresh installation and hoping you have a current backup of all your data. You need to be especially careful when using your super user privileges; otherwise, you might do irreparable damage to your system.

However, the ability to work as the super user is fundamental to a healthy Linux system and should not be feared, but rather respected, even while used only with focused attention. Without this ability, you could not install new software, edit system configuration files, or do a large number of important administration tasks. By the way, you have already been performing operations with super user privileges from the GUI if you have ever been asked to enter your password to complete a specific task, such as installing software updates. The difference is that most graphical interfaces limit the options that users have and make it a little more difficult to do some of the big, disruptive tasks, even the ones that are incredibly useful.

Ubuntu works slightly differently from many other Linux distributions. If you study some other Linux distros, especially older or more traditional ones, you will hear about a specific user account called `root`, which is a super user account. In those distros, instead of typing in `sudo` before a command while using a regular user account with super user privileges, you log in to the root account and simply issue the command without entering a password (at least by default; in almost all cases, `sudo` can be installed and configured in these distros). In those cases, you can tell when you are using the root account at the command line because you will see a pound sign (`#`) in the command line prompt in the place of the dollar sign (`$`).

For example: `matthew@seymour:~#` versus the usual `matthew@seymour:~$`

In Ubuntu, the root account is disabled by default because forcing regular users with super user privileges to type a specific command every time they want to execute a command as a super user should have the benefit of making them carefully consider what they are doing when they use that power. It is easy to forget to log out of a root account, and entering a powerful command while logged in to root can be catastrophic. However, if you are more experienced and comfortable with the more traditional method of using super user privileges and want to enable the root account, you can use the command `sudo passwd`. When prompted, enter your user password to confirm that your user account has super user privileges. You are then asked for a new UNIX password, which will be the password for the root account, so make sure to remember it. You are also prompted to repeat the password, in case you've made any mistakes. After you've typed it in and pressed Enter, the root account is active. You find out how to switch to root later on.

An alternative way of getting a root prompt, without having to enable the root account, is to issue the command `sudo -i`. After entering your password, you find yourself at a root prompt (`#`). Do what you need to do, and when you are finished, type `exit`, and press Enter to return to your usual prompt. You can learn more about `sudo` and root from an Ubuntu perspective at <https://help.ubuntu.com/community/RootSudo>.

Reading Documentation

Although you learn the basics of using Ubuntu in this book, you need time and practice to master and troubleshoot more complex aspects of the Linux operating system and your distribution. As with any operating system, you can expect to encounter some problems or perplexing questions as you continue to work with Linux. The first place to turn for help with these issues is the documentation included with your system; if you cannot find the information you need there, check Ubuntu's website.

Using Man Pages

To learn more about a command or program, use the `man` command followed by the name of the command. Man pages are stored in places like `/usr/share/man` and `/usr/local/share/man`, but you don't need to know that. To read a man page, such as the one for the `rm` command, use the `man` command like this:

```
matthew@seymour:~$ man rm
```

After you press Enter, the `less` command (a Linux command known as a *pager*) displays the man page. The `less` command is a text browser you can use to scroll forward and backward (even sideways) through the document to learn more about the command. Type the letter `h` to get help, use the forward slash (`/`) to enter a search string, or press `q` to quit.

No one can remember everything. Even the best and most experienced systems administrators use man pages regularly. Looking up complicated information is easy because this frees you from having to recall it all, enabling you to focus on your task rather than punishing you for not remembering syntax.

NOTE

Nearly all the hundreds of commands included with Linux each have a man page; however, some do not or may only have simple pages. You may also use the `info` command to read more detailed information about some commands or as a replacement for others. For example, to learn even more about `info` (which has a rather extensive manual page), use the `info` command like this:

```
matthew@seymour:~$ info info
```

Use the arrow keys to navigate through the document and press `q` to quit reading.

Using apropos

Linux, like UNIX, is a self-documenting system, with man pages accessible through the `man` command. Linux offers many other helpful commands for accessing its documentation. You can use the `apropos` command (for example, with a keyword such as `partition`) to find commands related to partitioning, like this:

```
matthew@seymour:~$ apropos partition
addpart      (8)    - Simple wrapper around the "add partition" ioctl
all-swaps    (7)    - Event signaling that all swap partitions have been ac...
cfdisk       (8)    - Curses/slang based disk partition table manipulator fo...
delpart      (8)    - Simple wrapper around the "del partition" ioctl
fdisk        (8)    - Partition table manipulator for Linux
gparted      (8)    - Gnome partition editor for manipulating disk partitions.
Mpartition   (1)    - Partition an MSDOS hard disk
Partprobe    (8)    - Inform the OS of partition table changes
Partx        (8)    - Telling the kernel about presence and numbering of on-...
Pvcreate     (8)    - Initialize a disk or partition for use by LVM
Pvresize     (8)    - Resize a disk or partition in use by LVM2
Sfdisk       (8)    - Partition table manipulator for Linux
```

Using whereis

To find a command and its documentation, you can use the `whereis` command. For example, if you are looking for the `fdisk` command, you can do this:

```
matthew@seymour:~$ whereis fdisk
fdisk: /sbin/fdisk /usr/share/man/man8/fdisk.8.gz
```

Understanding the Linux File System Hierarchy

Linux has inherited from UNIX a well-planned hierarchy for organizing things. It isn't perfect, but it is generally logical and mostly consistent, although distributions do tend to make some modifications that force some thinking and adaptation when moving between, say, Fedora, Slackware, and Ubuntu. Table 4.1 shows some of the top-level directories that are part of a standard Linux distro.

TABLE 10.1 Basic Linux Directories

Name	Description
/	The root directory
/bin	Essential commands
/boot	Boot loader files, Linux kernel
/dev	Device files
/etc	System configuration files
/home	User home directories
/lib	Shared libraries, kernel modules
/lost+found	Directory for recovered files (if found after a file system check)
/media	Mount point for removable media, such as DVDs and floppy disks
/mnt	Usual mount point for local, remote file systems
/opt	Add-on software packages

Name	Description
<code>/proc</code>	Kernel information, process control
<code>/root</code>	Super user (root) home
<code>/sbin</code>	System commands (mostly root only)
<code>/srv</code>	Holds information relating to services that run on your system
<code>/sys</code>	Real-time information on devices used by the kernel
<code>/tmp</code>	Temporary files
<code>/usr</code>	Software not essential for system operation, such as applications
<code>/var</code>	Variable data (such as logs); spooled files

Knowing these directories can help you find files when you need them. This knowledge can even help you partition hard drives when you install new systems by letting you choose to put certain directories on their own distinct partition, which can be useful for things like isolating directories from one another, such as a server security case like putting a directory like `/boot` that doesn't change often on its own partition and making it read-only and unchangeable without specific operations being done by a super user during a maintenance cycle. Desktop users probably won't need to think about that, but the directory tree is still quite useful to know when you want to find the configuration file for a specific program and set some program options systemwide to affect all users.

NOTE

This is a lot to remember, especially at first. For reference, there is a man page for the Linux filesystem hierarchy:

```
matthew@seymour:~$ man hier
```

This returns a detailed listing, with descriptions of each part.

Some of the important directories in Table 4.1, such as those containing user and root commands or system configuration files, are discussed in the following sections. You may use and edit files under these directories when you use Ubuntu.

Essential Commands in `/bin` and `/sbin`

The `/bin` directory contains essential commands used by the system for running and booting the system. In general, only the root operator uses the commands in the `/sbin` directory. The software in both locations is essential to the system; they make the system what it is, and if they are changed or removed, it could cause instability or a complete system failure. Often, the commands in these two directories are *statically* linked, which means that the commands do not depend on software libraries residing under the `/lib` or `/usr/lib` directories. Nearly all the other applications on your system are *dynamically* linked, meaning that they require the use of external software libraries (also known as *shared* libraries) to run. This is a feature for both sets of software.

The commands in `/bin` and `/sbin` are kept stable to maintain foundational system integrity and do not need to be updated often, if at all. For the security of the system, these commands are kept in a separate location and isolated where changes are more difficult and where it will be more obvious to the system administrator if unauthorized changes are attempted or made.

Application software changes more frequently, and applications often use the same functions that other pieces of application software use. This was the genesis of shared libraries. When a security update is needed for something that is used by more than one program, it has to be updated in only one location, a specific software library. This enables easy and quick security updates that will affect several pieces of non-system-essential software at the same time by updating one shared library, contained in one file on the computer.

Configuration Files in `/etc`

System configuration files and directories reside under the `/etc` directory. Some major software packages, such as Apache, OpenSSH, and `xinetd`, have their own subdirectories in `/etc` filled with configuration files. Others like `crontab` or `fstab` use one file. Examples of system-related configuration files in `/etc` include the following:

- ▶ **`fstab`**—The file system table is a text file listing each hard drive, CD-ROM, floppy, or other storage device attached to your PC. The table indexes each device's partition information with a place in your Linux file system (directory layout) and lists other options for each device when used with Linux (see Chapter 20, "Kernel and Module Management"). Nearly all entries in `fstab` can be manipulated by root using the `mount` command.
- ▶ **`modprobe.d/`**—This folder holds all the instructions to load kernel modules that are required as part of the system startup.
- ▶ **`passwd`**—The list of users for the system, including special-purpose nonhuman users like `syslog` and `CouchDB`, along with user account information.
- ▶ **`sudoers`**—A list of users or user groups with super user access.

User Directories: `/home`

The most important data on a non-server Linux system often resides in the user's directories, found under the `/home` directory. User directories are named by default according to account usernames, so on a computer where I have an account named `matthew`, my home directory would generally be found in `/home/matthew`. This can be changed, and if you're curious you can read more about it in Chapter 10, "Command-Line Master Class."

Segregating the system and user data can be helpful in preventing data loss and making the process of backing up easier. For example, having user data reside on a separate file system or mounted from a remote computer on the network might help shield users from data loss in the event of a system hardware failure. For a laptop or desktop computer at home, you might place `/home` on a separate partition from the rest of the file system, so

that if the operating system is upgraded, damaged, or reinstalled, `/home` would be more likely to survive the event intact.

Using the Contents of the `/proc` Directory to Interact with the Kernel

The content of the `/proc` directory is created from memory and exists only while Linux is running. This directory contains special files that either extract information from or send information to the kernel. Many Linux utilities extract information from dynamically created directories and files under this directory, also known as a *virtual file system*. For example, the `free` command obtains its information from a file named `meminfo`:

```
matthew@seymour:~$ free
```

	total	used	free	shared	buffers	cached
Mem:	4055680	2725684	1329996	0	188996	1551464
-/+ buffers/cache:		985224	3070456			
Swap:	8787512	0	8787512			

This information constantly changes as the system is used. You can get the same information by using the `cat` command to see the contents of the `meminfo` file:

```
matthew@seymour:~$ cat /proc/meminfo
```

```
MemTotal:          4055680 KB
MemFree:           1329692 KB
Buffers:           189208 KB
Cached:            1551488 KB
SwapCached:        0 KB
Active:            1222172 KB
Inactive:          1192244 KB
Active(anon):      684092 KB
Inactive(anon):    16 KB
Active(file):      538080 KB
Inactive(file):    1192228 KB
Unevictable:       48 KB
Mlocked:           48 KB
SwapTotal:         8787512 KB
SwapFree:          8787512 KB
Dirty:             136 KB
Writeback:         0 KB
AnonPages:         673760 KB
Mapped:            202308 KB
Shmem:             10396 KB
Slab:              129248 KB
SReclaimable:     107356 KB
SUnreclaim:       21892 KB
KernelStack:      2592 KB
PageTables:       30108 KB
```

```

NFS_Unstable:          0 KB
Bounce:                0 KB
WritebackTmp:         0 KB
CommitLimit:          10815352 KB
Committed_AS:         1553172 KB
VmallocTotal:         34359738367 KB
VmallocUsed:          342300 KB
VmallocChunk:         34359387644 KB
HardwareCorrupted:    0 KB
HugePages_Total:      0
HugePages_Free:       0
HugePages_Rsvd:       0
HugePages_Surp:       0
Hugepagesize:         2048 KB
DirectMap4k:          38912 KB
DirectMap2M:          4153344 KB

```

The `/proc` directory can also be used to dynamically alter the behavior of a running Linux kernel by “echoing” numerical values to specific files under the `/proc/sys` directory. For example, to “turn on” kernel protection against one type of denial-of-service (DoS) attack known as *SYN flooding*, use the `echo` command to send the number 1 to the following `/proc` path:

```
matthew@seymour:~$ sudo echo 1 >/proc/sys/net/ipv4/tcp_syncookies
```

Other ways to use the `/proc` directory include the following:

- ▶ Getting CPU information, such as the family, type, and speed from `/proc/cpuinfo`.
- ▶ Viewing important networking information under `/proc/net`, such as active interfaces information under `/proc/net/dev`, routing information in `/proc/net/route`, and network statistics in `/proc/net/netstat`.
- ▶ Retrieving file system information.
- ▶ Reporting media mount point information via USB; for example, the Linux kernel reports what device to use to access files (such as `/dev/sda`) if a USB camera or hard drive is detected on the system. You can use the `dmesg` command to see this information.
- ▶ Getting the kernel version in `/proc/version`, performance information such as uptime in `/proc/uptime`, or other statistics such as CPU load, swap file usage, and processes in `/proc/stat`.

Working with Shared Data in the `/usr` Directory

The `/usr` directory contains software applications, libraries, and other types of shared data for use by anyone on the system. Many Linux system administrators give `/usr` its own partition. A number of subdirectories under `/usr` contain manual pages (`/usr/share/man`),

software package shared files (`/usr/share/name_of_package`, such as `/usr/share/emacs`), additional application or software package documentation (`/usr/share/doc`), and an entire subdirectory tree of locally built and installed software, `/usr/local`.

Temporary File Storage in the `/tmp` Directory

As its name implies, the `/tmp` directory is used for temporary file storage; as you use Linux, various programs create files in this directory.

Accessing Variable Data Files in the `/var` Directory

The `/var` directory contains subdirectories used by various system services for spooling and logging. Many of these variable data files, such as print spooler queues, are temporary, whereas others, such as system and kernel logs, are renamed and rotated in use. Incoming email is usually directed to files under `/var/spool/mail`.

Linux also uses `/var` for other important system services. These include the topmost *File Transfer Protocol (FTP)* directory under `/var/ftp` (see Chapter 24, “Remote File Serving with FTP”), and the Apache web server’s initial home page directory for the system, `/var/www/html`. (See Chapter 22, “Apache Web Server Management,” for more information about using Apache.)

NOTE

There is a recent trend to move data that is served from `/var/www` and `/var/ftp` to `/srv`, but this is not universal.

Navigating the Linux File System

In the Linux file system, as with its predecessor UNIX, everything is a file: data files, binary files, executable programs, even input and output devices. These files are placed in a series of directories that act like file folders. A directory is nothing more than a special type of file that contains a list of other files/directories. These files and directories are used to create a hierarchical structure that enables logical placement of specific types of files. Later this chapter discusses the standard hierarchy of the Linux file system. First, you learn how to navigate and interact with the file system.

NOTE

A directory with contents is called a *parent*, and its contents are called *children*, as in “`/home/matthew/Documents` is a child directory of `/home/matthew`, its parent.”

Listing the Contents of a Directory with `ls`

The `ls` command lists the contents of the current directory. It is commonly used by itself, but a number of options (also known as switches) are available for `ls` and give you more

information. If you have just logged in as described earlier, this command lists the files and directories in your user's home directory:

```
matthew@seymour:~$ ls
Documents  Music      file.txt  Pictures  Music
```

NOTE

All directory listings in this chapter are abbreviated to save space.

By itself, the `ls` command shows just a list of names. Some are files, some are directories. This is useful if I know what I am looking for but cannot remember the exact name. However, using `ls` in this manner has some limitations. First, it does not show hidden files. Hidden files use filenames that start with a period (.) as the first character. They are often used for configuration of specific programs and are not accessed frequently. For this reason, they are not included in a basic directory listing. You can see all the hidden files by adding a switch to the command like this:

```
matthew@seymour:~$ ls -a
.          .bash_logout  Documents     Music
..         .bashrc       file.txt      Pictures
.bash_history .config       .local        .profile
```

There is still more information available about each item in a directory. To include details such as the file/directory permissions, owner and group (all of which are discussed later in this chapter), as well as the size, and the date and time it was last modified, enter the following:

```
matthew@seymour:~$ ls -al
total 608
drwxr-xr-x 38 matthew matthew 4096 2011-06-04 08:20 .
drwxr-xr-x  3 root    root    4096 2011-05-16 16:48 ..
-rw-----  1 matthew matthew  421 2011-06-04 10:27 .bash_history
-rw-r--r--  1 matthew matthew   220 2011-05-16 16:48 .bash_logout
-rw-r--r--  1 matthew matthew 3353 2011-05-16 16:48 .bashrc
drwxr-xr-x 13 matthew matthew 4096 2011-05-21 10:42 .config
drwxr-xr-x  2 matthew matthew 4096 2011-05-16 17:07 Documents
-rw-r--r--  1 matthew matthew  335 2011-05-16 16:48 file.txt
drwxr-xr-x  3 matthew matthew 4096 2011-05-16 17:07 .local
drwxr-xr-x  2 matthew matthew 4096 2011-05-16 17:07 Music
drwxr-xr-x  3 matthew matthew 4096 2011-05-16 18:07 Pictures
-rw-r--r--  1 matthew matthew  675 2011-05-16 16:48 .profile
```

The listing (abbreviated here) is now given with one item per line, but with multiple columns. The listing starts with the number of items in the directory. (Both files and sub-directories are included; remember that the listing here is abbreviated.) Then, the details are as shown in Figure 9.1.

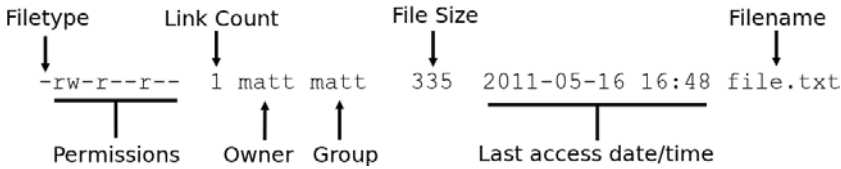


FIGURE 9.1 Decoding the output of a detailed directory listing.

These details are discussed more completely later in the chapter in the “Working with Permissions” section.

Another useful switch is this:

```
matthew@seymour:~$ ls -R
```

This command scans and lists all the contents of the subdirectories of the current directory. This is likely to be a lot of information, so you might want to redirect the output to a text file so that you can browse through it at your leisure by using the following:

```
matthew@seymour:~$ ls -laR > listing.txt
```

TIP

The previous command sends the output of `ls -laR` to a file called `listing.txt` and demonstrates part of the power of the Linux command line. At the command line, you can use files as inputs to commands, or you can generate files as outputs as shown. For more information about redirects and combining commands, see Chapter 12, “Automating Tasks and Shell Scripting.” In the meantime, note that you can read the contents of that text file using the command `less listing.txt`, which lets you read the file bit by bit using the arrow keys to navigate in the file (or Enter to move to the next line), the spacebar to move to the next page, and `q` to exit when done.

Changing Directories with `cd`

Use the `cd` command to move within the file system from one directory to another. It might help you remember this command to think of it meaning “change directory.” The most basic usage of `cd` is this:

```
matthew@seymour:~$ cd somedir
```

That looks in the current directory for the `somedir` subdirectory, and then moves you into it. You can also specify an exact location for a directory, like this:

```
matthew@seymour:~$ cd /home/matthew/stuff/somedir
```

You can also use the `cd` command with several shortcuts. For example, to quickly move up to the *parent* directory, the one above the one you are currently in, use the `cd` command like this:

```
matthew@seymour:~$ cd ..
```

To return to your home directory from anywhere in the Linux file system, use the `cd` command like this:

```
matthew@seymour:~$ cd
```

You can also use the `$HOME` shell environment variable to accomplish the same thing. Environment variables are discussed in greater detail in Chapter 10, “Command-Line Master Class.” Type this command and press Enter to return to your home directory:

```
matthew@seymour:~$ cd $HOME
```

You can accomplish the same thing by using the tilde (`~`) like this:

```
matthew@seymour:~$ cd ~
```

Finding Your Current Directory with `pwd`

Use `pwd` to remind you where you are within the file system.

Working with Permissions

Under Linux (and UNIX), everything in the file system, including directories and devices, is a file. And every file on your system has an accompanying set of permissions based on ownership. These permissions provide data security by giving specific permission settings to every single item denoting who may read, write, or execute the file. These permissions are set individually for the file’s owner, for members of the group the file belongs to, and for all others on the system.

You can examine the default permissions for a file you create by using the `umask` command, which lists default permissions using the number system explained next, or by using the `touch` command and then the `ls` command’s long-format listing like this:

```
matthew@seymour:~$ touch file
matthew@seymour:~$ ls -l file
-rw-r--r-- 1 matthew matthew 0 2010-06-30 13:06 file
```

In this example, the `touch` command is used to quickly create a file. The `ls` command then reports on the file, displaying the following (from left to right):

- **The type of file created**—Common indicators of the type of file are in the leading letter in the output. A blank (which is represented by a dash, as in the preceding example) designates a plain file, `d` designates a directory, `c` designates a character device (such as `/dev/ttyS0`), and `b` is used for a block device (such as `/dev/sda`).

- ▶ **Permissions**—Read, write, and execute permissions for the owner, group, and all others on the system. (You learn more about these permissions later in this section.)
- ▶ **Number of links to the file**—The number `1` designates that there is only one file, whereas any other number indicates that there might be one or more hard-linked files. Links are created with the `ln` command. A hard-linked file is an exact copy of the file, but it might be located elsewhere on the system. Symbolic links of directories can also be created, but only the root operator can create a hard link of a directory.
- ▶ **The owner**—The account that owns the file; this is originally the file creator, but you can change this designation using the `chown` command.
- ▶ **The group**—The group of users allowed to access the file; this is originally the file creator’s main group, but you can change this designation using the `chgrp` command.
- ▶ **File size and creation/modification date**—The last two elements indicate the size of the file in bytes and the date the file was created or last modified.

Assigning Permissions

Under Linux, permissions are grouped by owner, group, and others, with read, write, and execute permission assigned to each, as follows:

Owner	Group	Others
<code>rwX</code>	<code>rwX</code>	<code>rxw</code>

Permissions can be indicated by mnemonic or octal characters. Mnemonic characters are listed here:

- ▶ `r` indicates permission for an owner, member of the owner’s group, or others to open and read the file.
- ▶ `w` indicates permission for an owner, member of the owner’s group, or others to open and write to the file.
- ▶ `x` indicates permission for an owner, member of the owner’s group, or others to execute the file (or read a directory).

In the previous example for the file named `file`, the owner, `matthew`, has read and write permission. Any member of the group named `matthew` may only read the file. All other users may only read the file. Also note that default permissions for files created by the root operator (while using `sudo` or a root account) will differ because of `umask` settings assigned by the shell.

Many users prefer to use numeric codes, based on octal (base 8) values, to represent permissions. Here’s what these values mean:

- ▶ 4 indicates read permission.
- ▶ 2 indicates write permission.
- ▶ 1 indicates execute permission.

In octal notation, the previous example file has a permission setting of `644` (read + write or $4 + 2$, read-only or 4 , read-only or 4). Although you can use either form of permissions notation, octal is easy to use quickly after you visualize and understand how permissions are numbered.

NOTE

In Linux, you can create groups to assign a number of users access to common directories and files, based on permissions. You might assign everyone in accounting to a group named `accounting` and allow that group access to accounts payable files while disallowing access by other departments. Defined groups are maintained by the root operator, but you can use the `newgrp` command to temporarily join other groups to access files (as long as the root operator has added you to the other groups). You can also allow or deny other groups' access to your files by modifying the group permissions of your files.

Directory Permissions

Directories are also files under Linux. For example, again use the `ls` command to show permissions like this:

```
matthew@seymour:~$ mkdir directory
matthew@seymour:~$ ls -ld directory
drwxr-xr-x 2 matthew matthew 4096 2010-06-30 13:23 directory
```

In this example, the `mkdir` command is used to create a directory. The `ls` command, and its `-ld` option, is used to show the permissions and other information about the directory (not its contents). Here you can see that the directory has permission values of `755` (read + write + execute or $4 + 2 + 1$, read + execute or $4 + 1$, and read + execute or $4 + 1$).

This shows that the owner can read and write to the directory and, because of execute permission, also list the directory's contents. Group members and all other users can list only the directory contents. Note that directories require execute permission for anyone to be able to view their contents.

You should also notice that the `ls` command's output shows a leading `d` in the permissions field. This letter specifies that this file is a directory; normal files have a blank field in its place. Other files, such as those specifying a block or character device, have a different letter.

For example, if you examine the device file for a Linux serial port, you will see the following:

```
matthew@seymour:~$ ls -l /dev/ttyS0
crw-rw---- 1 root dialout 4, 64 2010-06-30 08:13 /dev/ttyS0
```

Here, `/dev/ttyS0` is a character device (such as a serial communications port and designated by a `c`) owned by `root` and available to anyone in the `dialout` group. The device has permissions of `660` (read + write, read + write, no permission).

On the other hand, if you examine the device file for an IDE hard drive, you see this:

```
matthew@seymour:~$ ls -l /dev/sda
brw-rw-- -- 1 root disk 8, 0 2010-06-30 08:13 /dev/sda
```

In this example, `b` designates a block device (a device that transfers and caches data in blocks) with similar permissions. Other device entries you will run across on your Linux system include symbolic links, designated by `s`.

Altering File Permissions with `chmod`

You can use the `chmod` command to alter a file's permissions. This command uses various forms of command syntax, including octal or a mnemonic form (such as `u`, `g`, `o`, or `a` and `rwx`, and so on) to specify a desired change. You can use the `chmod` command to add, remove, or modify file or directory permissions to protect, hide, or open up access to a file by other users (except for the root account or a user with super user permission and using `sudo`, either of which can access any file or directory on a Linux system).

The mnemonic forms of `chmod`'s options are (when used with a plus character, `+`, to add, or a minus sign, `-`, to remove):

- ▶ **u**—Adds or removes user (owner) read, write, or execute permission
- ▶ **g**—Adds or removes group read, write, or execute permission
- ▶ **o**—Adds or removes read, write, or execute permission for others not in a file's group
- ▶ **a**—Adds or removes read, write, or execute permission for all users
- ▶ **r**—Adds or removes read permission
- ▶ **w**—Adds or removes write permission
- ▶ **x**—Adds or removes execution permission

For example, if you create a file, such as a `readme.txt`, the file has the following default permissions (set by the `umask` setting in `/etc/bashrc`):

```
-rw-r--r-- 1 matthew matthew 0 2010-06-30 13:33 readme.txt
```

As you can see, you can read and write the file. Anyone else can only read the file (and only if it is outside your home directory, which will have read, write, and execute

permission set only for you, the owner). You can remove all write permission for anyone by using `chmod`, the minus sign (-), and `aw`, as follows:

```
matthew@seymour:~$ chmod a-w readme.txt
matthew@seymour:~$ ls -l readme.txt
-r--r--r-- 1 matthew matthew 0 2010-06-30 13:33 readme.txt
```

Now, no one can write to the file (except you, if the file is in your `/home` or `/tmp` directory because of directory permissions). To restore read and write permission for only you as the owner, use the plus sign (+) and the `u` and `rw` options like so:

```
matthew@seymour:~$ chmod u+rw readme.txt
matthew@seymour:~$ ls -l readme.txt
-rw-r--r-- 1 matthew matthew 0 2010-06-30 13:33 readme.txt
```

You can also use the octal form of the `chmod` command (for example, to modify a file's permissions so that only you, the owner, can read and write a file). Use the `chmod` command and a file permission of `600`, like this:

```
matthew@seymour:~$ chmod 600 readme.txt
matthew@seymour:~$ ls -l readme.txt
-rw----- 1 matthew matthew 0 2010-06-30 13:33 readme.txt
```

If you take away execution permission for a directory, files might be hidden inside and may not be listed or accessed by anyone else (except the root operator, of course, who has access to any file on your system). By using various combinations of permission settings, you can quickly and easily set up a more secure environment, even as a normal user in your `/home` directory.

File Permissions with `chgrp`

You can use the `chgrp` command to change the group to which a file belongs:

```
matthew@seymour:~$ chgrp wheel filename
```

Changing File Permissions with `chown`

You can use the `chown` command to change the owner of a file:

```
matthew@seymour:~$ chown matthew filename
```

You can also use the `chown` command to change the group of a file at the same time:

```
matthew@seymour:~$ chown matthew:wheel filename
```

Understanding Set User ID and Set Group ID Permissions

Two more types of permission are “set user ID,” known as *suid*, and “set group ID,” or *sgid*. These settings, when used in a program, enable any user running that program to have

program owner or group owner permissions for that program. These settings enable the program to be run effectively by anyone, without requiring that each user's permissions be altered to include specific permissions for that program.

One commonly used program with `suid` permissions is the `passwd` command:

```
matthew@seymour:~$ ls -l /usr/bin/passwd
-rwsr-xr-x 1 root root 42856 2010-01-26 10:09 /usr/bin/passwd
```

This setting allows normal users to execute the command (as root) to make changes to a root-only-accessible file `/etc/passwd`.

You also can assign similar permission with the `chfn` command. This command allows users to update or change `finger` information in `/etc/passwd`. You accomplish this permission modification by using a leading `4` (or the mnemonic `s`) in front of the three octal values.

NOTE

Other files that might have `suid` or `guid` permissions include `at`, `rcp`, `rlogin`, `rsh`, `chage`, `chsh`, `ssh`, `crontab`, `sudo`, `sendmail`, `ping`, `mount`, and several UNIX-to-UNIX Copy (UUCP) utilities. Many programs (such as games) might also have this type of permission to access a sound device.

Files or programs that have `suid` or `guid` permissions can sometimes present security holes because they bypass normal permissions. This problem is compounded if the permission extends to an executable binary (a command) with an inherent security flaw because it could lead to any system user or intruder gaining root access. In past exploits, this typically happened when a user fed a vulnerable command with unexpected input (such as a long pathname or option); the command would fail, and the user would be presented a root prompt. Although Linux developers are constantly on the lookout for poor programming practices, new exploits are found all the time, and can crop up unexpectedly, especially in newer software packages that haven't had the benefit of peer developer review.

Savvy Linux system administrators keep the number of `suid` or `guid` files present on a system to a minimum. The `find` command can be used to display all such files on your system:

```
matthew@seymour:~$ sudo find / -type f -perm /6000 -exec ls -l {} \;
```

NOTE

The `find` command is quite helpful and can be used for many purposes, such as before or during backup operations.

Note that the programs do not necessarily have to be removed from your system. If your users really do not need to use the program, you can remove the programs execute permission for anyone. You have to decide, as the root operator, whether your users are

allowed, for example, to mount and unmount CD-ROMs or other media on your system. Although Linux-based operating systems can be set up to accommodate ease of use and convenience, allowing programs such as `mount` to be `suid` might not be the best security policy. Other candidates for `suid` permission change could include the `chsh`, `at`, or `chage` commands.

Working with Files

Managing files in your home directory involves using one or more easily remembered commands.

Creating a File with `touch`

To create an empty file called `filename` within your current directory, use the following command:

```
matthew@seymour:~$ touch filename
```

To edit this file, you must use a text editor. Several are discussed in Chapter 12, “Automating Tasks and Shell Scripting.” However, it is sometimes useful to create an empty file as this creates an access record because of the time and date information that is connected to the file. You can also use `touch` to update this information, called a timestamp, without otherwise accessing or modifying a file.

You can create a file in a different location by changing what is after `touch`. To create a new file in `/home/matthew/randomdirectory`, if I am already in my home directory, I can use the following:

```
matthew@seymour:~$ touch randomdirectory/newfile
```

Or from anywhere using an absolute path, I use this:

```
matthew@seymour:~$ touch /home/matthew/randomdirectory/newfile
```

Or from anywhere using a path shortcut, I use the following command:

```
matthew@seymour:~$ touch ~/randomdirectory/newfile
```

Creating a Directory with `mkdir`

To create an empty directory called `newdirectory` within your current directory, use this command:

```
matthew@seymour:~$ mkdir newdirectory
```

You can create a directory in a different location by changing what is after `mkdir`. To create a new directory in `/home/matthew/music`, if I am already in my `/home` directory, I can use the following:

```
matthew@seymour:~$ mkdir music/newdirectory
```

Or from anywhere using an absolute path, I can use this:

```
matthew@seymour:~$ mkdir /home/matthew/music/newdirectory
```

Or from anywhere using a path shortcut, I can use the following command:

```
matthew@seymour:~$ mkdir ~/music/newdirectory
```

The `-p` option is valuable. It enables you to create a directory and its parent directories at the same time, if they do not already exist. This can be a real time saver. If the parent directories exist, the command works normally. For example, suppose I want to make a new directory with two layers of subdirectories. In this example, `music` and `newdirectory` already exist, but `subdir1` and `subdir2` are to be created:

```
matthew@seymour:~$ mkdir -p ~/music/newdirectory/subdir1/subdir2
```

Deleting a Directory with `rmdir`

To delete an empty directory named `directoryname`, use the following command:

```
matthew@seymour:~$ rmdir directoryname
```

You can remove a directory in a different location by changing what is after `rmdir`. To remove a directory in `/home/matthew/music`, if I am already in my `/home` directory, I can use the following:

```
matthew@seymour:~$ rmdir music/directoryname
```

Or from anywhere using an absolute path, I can use this:

```
matthew@seymour:~$ rmdir /home/matthew/music/directoryname
```

Or from anywhere using a path shortcut, I can use the following command:

```
matthew@seymour:~$ rmdir ~/music/directoryname
```

The directory must be empty to be removed using `rmdir`. However, there is a way to remove a directory with its contents using `rm`.

CAUTION

You cannot easily recover anything that has been deleted using `rmdir` or `rm`, so proceed carefully. Be absolutely certain you will never need what you are about to delete before you do so. Only a professional data recovery service is likely to be able to recover the files, and even then at great expense.

Deleting a File or Directory with `rm`

To delete a file named `filename`, use this command:

```
matthew@seymour:~$ rm filename
```

You can remove a directory in a different location by changing what is after `rm`. To remove a directory in `/home/matthew/randomdirectory`, if I am already in my `/home` directory, I can use the following:

```
matthew@seymour:~$ rm randomdirectory/filename
```

Or from anywhere using an absolute path, I can use this:

```
matthew@seymour:~$ rm /home/matthew/randomdirectory/filename
```

Or from anywhere using a path shortcut, I can use the following command:

```
matthew@seymour:~$ rm ~/randomdirectory/filename
```

If you try to use `rm` to remove an empty directory, you will receive an error message: `rm: cannot remove `random/': Is a directory`. In this case, you must use `rmdir`. However, you can remove a directory and its contents using `rm`.

CAUTION

Be sure that all the contents of a directory are known and unwanted if you choose to delete them. There is no way to recover them later. Also, be careful that you don't have extra spaces, mistype the name of the directory, or use `sudo` to delete something that you shouldn't be deleting. Linux gives you great power, and it will let you use that power without questioning you about it; that's the human's job.

To delete a directory and all its contents, use the `-R` recursive switch. This switch works with many commands, not only `rm`:

```
matthew@seymour:~$ rm -R /home/matthew/randomdirectory/
```

Everything in `randomdirectory` as well as in the directory itself will be deleted.

Moving or Renaming a File with `mv`

In Linux land, moving and renaming a file are the same thing. It doesn't matter whether you are moving the directory to another or from one filename to another filename in the same directory, there is only one command to remember. To move a file named `filename` from `~/documents` to `~/archive`, use this command:

```
matthew@seymour:~$ mv documents/filename archive
```

Notice that the filename is not included in the destination. The destination here must be an existing directory. If it is not, the file is renamed to the term used. Some examples will make this clear.

To rename a file that is in my current directory, I could use the following:

```
matthew@seymour:~$ mv oldfilename newfilename
```

To rename a file as I move it from `~/documents` to `~/archive`, I could use this:

```
matthew@seymour:~$ mv documents/oldfilename archive/newfilename
```

Or from anywhere using an absolute path, I could use the following command:

```
matthew@seymour:~$ mv /home/matthew/documents/oldfilename
➤/home/matthew/archive/newfilename
```

Or from anywhere using a path shortcut, I could use this:

```
matthew@seymour:~$ rm ~/documents/oldfilename ~/archive/newfilename
```

Copying a File with `cp`

Copying works similarly to moving, but retains the original in the original location. To copy a file named `filename` from `~/documents` to `~/archive`, use this command:

```
matthew@seymour:~$ cp documents/filename archive
```

Notice that the filename is not included in the destination. The destination here must be an existing directory. If it is not, the file is renamed to the term used. Some examples will make this clear.

To copy a file that is in my current directory I could use the following, and it will work exactly the same as `mv`, except that both files will exist afterward:

```
matthew@seymour:~$ cp oldfilename newfilename
```

To rename a file as I copy it from `~/documents` to `~/archive`, I could use this:

```
matthew@seymour:~$ cp documents/oldfilename archive/newfilename
```

Or from anywhere using an absolute path, I could use the following command:

```
matthew@seymour:~$ cp /home/matthew/documents/oldfilename
➤/home/matthew/archive/newfilename
```

Or from anywhere using a path shortcut, I can use this:

```
matthew@seymour:~$ cp ~/documents/oldfilename ~/archive/newfilename
```

Displaying the Contents of a File with `cat`

To view the contents of a text file named `filename` on your screen, use this command:

```
matthew@seymour:~$ cat filename
```

Notice that the text is displayed on your screen but that you cannot edit or work with the text in any way. This command is convenient when you want to know the contents of a file but don't need to make any changes. Text editors for the terminal are covered in Chapter 10, "Command-Line Master Class." This command works best with short files because the contents of longer files will scroll off of the screen too quickly to be read.

Displaying the Contents of a File with `less`

When you need to view the contents of a longer text file from the command line, you can use `less`. This produces a paged output, meaning that output stops each time your screen is full. You can then use your up- and down-arrow keys and page-up and page-down keys to scroll through the contents of the file. Then, use `q` to quit and return to the command line:

```
matthew@seymour:~$ less filename
```

There was a program that did give paged output in the early days of UNIX called `more`. It was the first paged output program but did not include the ability to scroll up and down. `less` was written to add that capability and was named as a bit of hacker humor because "less is more." You can also use `more`, but today it is merely an alias for `less`.

Using Wildcards and Regular Expressions

Each of these commands can be used with pattern-matching strings known as *wildcards* or *regular expressions*. For example, to delete all files in the current directory beginning with the letters `abc`, you can use an expression beginning with the first three letters of the desired filenames. An asterisk (*) is then appended to match all these files. Use a command line with the `rm` command like this:

```
matthew@seymour:~$ rm abc*
```

Linux shells recognize many types of file naming wildcards, but this is different from the capabilities of Linux commands supporting the use of more complex expressions. You learn more about using wildcards in Chapter 10, "Command-Line Master Class," and in Chapter 12, "Automating Tasks and Shell Scripting."

NOTE

You can also learn more about using expressions by reading the `grep` manual pages (`man grep`), but because both `man` and `grep` are covered in Chapter 10, "Command-Line Master Class," consider this mention as included only to whet your appetite.

Working as Root

The root, or super user account, is a special account and user on UNIX and Linux systems. Super user permissions are required in part because of the restrictive file permissions assigned to important system configuration files. You must have root permission to edit these files or to access or modify certain devices (such as hard drives). When logged in as root, you have total control over your system, which can be dangerous.

When you work in root, you can destroy a running system with a simple invocation of the `rm` command like this:

```
matthew@seymour:~$ sudo rm -rf /
```

This command line not only deletes files and directories but also could wipe out file systems on other partitions and even remote computers. This alone is reason enough to take precautions when using root access.

The only time you should run Linux as the super user is when you are configuring the file system, for example, or to repair or maintain the system. Logging in and using Linux as the root operator isn't a good idea because it defeats the entire concept of file permissions.

Knowing how to run commands as the super user (root) without logging in as root can help avoid serious missteps when configuring your system. In Ubuntu, you can use `sudo` to allow you to execute single commands as root and then quickly return to normal user status. For example, if you would like to edit your system's file system table (a text file that describes local or remote storage devices, their type, and location), you can use `sudo` like this:

```
matthew@seymour:~$ sudo nano -w /etc/fstab
[sudo] password for matthew:
```

After you press Enter, you are prompted for a password that gives you access to root. This extra step can also help you “think before you leap” into the command. Enter the root password, and you are then editing `/etc/fstab`, using the `nano` editor with line wrapping disabled (thanks to the `-w`).

CAUTION

Before editing any important system or software service configuration file, make a backup copy. Then make sure to launch your text editor with line wrapping disabled. If you edit a configuration file without disabling line wrapping, you could insert spurious carriage returns and line feeds into its contents, causing the configured service to fail when restarting. By convention, nearly all configuration files are formatted for 80-character text width, but this is not always the case. By default, the `vi` and `emacs` editors don't use line wrap.

Understanding and Fixing `sudo`

Most Ubuntu users never have a problem here, but sometimes, people who like to experiment break things, especially while learning. This section exists to help you first

understand more completely how `sudo` works and also how to restore `sudo` access to a specific user when, for some reason, it has ceased to function for that user.

NOTE

You will usually know a problem has occurred because an error message like this will appear when a user tries to issue a command using `sudo`:

```
matthew@seymour:~$ sudo shutdown -h now
[sudo] password for matthew:
matthew is not in the sudoers file. This incident will be reported.
```

Sometimes, you might not even receive an error message, but the command issued simply does nothing. Either way, you can fix the problem using the following knowledge and procedure.

In order for a user to use `sudo`, the user account must belong to the `admin` group and also be listed in the `/etc/sudoers` file. If both conditions are met, the user will be permitted to temporarily use root powers for specific commands that are issued at the command line by that user account by prefacing the command with the word `sudo`.

A problem can occur for a specific user with `sudo` when

- ▶ A user is taken out of the `admin` group but should not have been. The permissions for the `/etc/sudoers` file has been changed to anything other than 440. The `/etc/sudoers` file has been changed in a way that does not allow members of the `admin` group to use root powers.

TIP

Generally, these things are the result of a user doing something they should not have done, such as changing the permissions on all files rather than taking the time to figure out a specific file that is causing problems due to permissions issues. Take heed, it is better to spend a bit more time learning than it is to take a shortcut that causes bigger problems.

Fixing any of these problems requires the use of root powers. This is an obvious problem because if `sudo` is not working, then the account does not have access to root. To fix it, we must gain root access. You can do this by booting into *recovery mode* using the following steps:

1. Hold down the Shift key while the computer is booting.
2. When the GRUB menu page appears, use the arrow keys on your keyboard to scroll to the entry that ends with (recovery mode) and press Enter to select it.
3. When the boot process finishes, you have several options. Select the menu entry for root, which is described as Drop to Root Shell Prompt. You are now at the command line with full root access to the computer.

4. Ubuntu mounts filesystems as read-only by default in recovery mode, so you need to remount the root filesystem, /, as read-write so that you can fix the problem. Enter the following:

```
root@seymour:~# mount -o rw,remount /
```

NOTE

You now have complete root access and read-write privileges on the machine. This is an example of why security of a physical machine is important. If someone has physical access to your computer, he can easily and quickly gain full control over the machine and all it contains if he knows what he is doing.

If the problem exists because the user account was removed from the admin group, enter

```
root@seymour:~# adduser username admin
```

If the problem exists because the permissions for */etc/sudoers* are wrong, enter

```
root@seymour:~# chmod 440 /etc/sudoers
```

If the problem exists because of an internal problem in */etc/sudoers*, make a backup of the existing file and use `visudo` to edit it (this is a special use of the `vi` editor, covered in Chapter 10, “Command-Line Master Class,” that runs a check on the file after editing to be certain that it is correct—this particular problem usually occurs when someone edits the file using another editor that does not make this check). The contents of the file should be the following:

```
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults env_reset
Defaults secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
```

```

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root ALL=(ALL:ALL) ALL

# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:

#include_dir /etc/sudoers.d

```

After your fix is complete, exit the root command line:

```
root@seymour:~# exit
```

You return to the recovery mode menu. Select resume, described as Resume Normal Boot, to finish and return to a normal boot. When the boot completes, you should be able to use `sudo` correctly again.

Creating Users

When a Linux system administrator creates a user, an entry in `/etc/passwd` for the user is created. The system also creates a directory, labeled with the user's username, in the `/home` directory. For example, if you create a user named `heather`, the user's home directory is `/home/heather`.

NOTE

In this chapter, you learn how to manage users from the command line. See Chapter 11, "Managing Users," for more information on user administration including doing so using graphical administration utilities.

Use the `adduser` command, along with a user's name, to quickly create a user:

```
matthew@seymour:~$ sudo adduser heather
```

After creating the user, you must also create the user's initial password with the `passwd` command:

```
matthew@seymour:~$ sudo passwd heather
```

```
Changing password for user heather.
```

```
New password:
```

```
Retype new password:
```

```
passwd: all authentication tokens updated successfully.
```

Enter the new password twice. If you do not create an initial password for a new user, the user cannot log in.

The `adduser` command has many command-line options. The command can be used to set policies and dates for the new user's password, assign a login shell, assign group membership, and other aspects of a user's account. See `man adduser` as well as Chapter 11, "Managing Users," for more info.

Deleting Users

Use the `deluser` command to delete users from your system. This command removes a user's entry in the system's `/etc/passwd` file. You should also use the command's `--remove-all-files` and `--remove-home` option to remove all the user's files and directories (such as the user's mail spool file under `/var/spool/mail`):

```
matthew@seymour:~$ sudo deluser --remove-all-files --remove-home andrew
```

If you do not use the `-r` option, you have to manually delete the user's directory under `/home`, along with the user's `/var/spool/mail` queue.

Shutting Down the System

Use the `shutdown` command to shut down your system. The `shutdown` command has a number of different command-line options (such as shutting down at a predetermined time), but the fastest way to cleanly shut down Linux is to use the `-h` or `halt` option, followed by the word `now` or the numeral zero (0), like this:

```
matthew@seymour:~$ sudo shutdown -h now
```

or

```
matthew@seymour:~$ sudo shutdown -h 0
```

To incorporate a timed shutdown and a pertinent message to all active users, use `shutdown`'s time and message options, as follows:

```
matthew@seymour:~$ sudo shutdown -h 18:30 "System is going down for maintenance this evening at 6:30 p.m. Please make sure you have saved your work and logged out by then or you may lose data."
```

This example shuts down your system and provides a warning to all active users 15 minutes before the shutdown (or reboot). Shutting down a running server can be considered drastic, especially if there are active users or exchanges of important data occurring (such as a backup in progress). One good approach is to warn users ahead of time. This can be done by editing the system *Message of the Day (MOTD)* `motd` file, which displays a message to users when they log in using the command-line interface, as is common on multi-user systems.

It used to be that to create a custom MOTD you only had to use a text editor and change the contents of `/etc/motd`. However, this has changed in Ubuntu as the developers have added a way to automatically and regularly update some useful information contained in MOTD using `cron`. To modify how the MOTD is updated, you should install `update-motd` and read the man page.

You can also make downtimes part of a regular schedule, perhaps to coincide with security audits, software updates, or hardware maintenance.

You should shut down Ubuntu for only a few very specific reasons:

- ▶ You are not using the computer, no other users are logged in or expected to need or use the system, such as your personal desktop or laptop computer, and you want to conserve electrical power.
- ▶ You need to perform system maintenance that requires any or all system services to be stopped.
- ▶ You want to replace integral hardware.

TIP

Do not shut down your computer if you suspect that intruders have infiltrated your system; instead, disconnect the machine from any or all networks and make a backup copy of your hard drives. You might want to also keep the machine running to examine the contents of memory and to examine system logs. Exceptions to this are when the system contains only trivial data files and nonessential services, such as a personal computer that is only used to run a web browser, and when you have no intention of trying to track down what an intruder might have changed, either to repair the damage or to try to catch them using computer forensics, but rather plan to merely wipe everything clean and rebuild or reinstall the system from scratch.

Rebooting the System

You should also use the `shutdown` command to reboot your system. The fastest way to cleanly reboot Linux is to use the `-r` option, and the word `now` or the numeral zero (0):

```
matthew@seymour:~$ sudo shutdown -r now
```

Or

```
matthew@seymour:~$ sudo shutdown -r 0
```

Both rebooting and shutting down can have dire consequences if performed at the wrong time (such as during backups or critical file transfers, which arouses the ire of your system's users). However, Linux-based operating systems are designed to properly stop active system services in an orderly fashion. Other commands you can use to shut down and reboot Linux are the `halt` and `reboot` commands, but the `shutdown` command is more flexible.

Commonly Used Commands and Programs

The following programs and built-in shell commands are commonly used when working at the command line. These commands are organized by category to help you understand the command's purpose. If you need to find full information for using the command, you can find that information under the command's man page.

- ▶ **Managing users and groups**—`chage`, `chfn`, `chsh`, `edquota`, `gpasswd`, `groupadd`, `groupdel`, `groupmod`, `groups`, `mkpasswd`, `newgrp`, `newusers`, `passwd`, `umask`, `useradd`, `userdel`, `usermod`
- ▶ **Managing files and file systems**—`cat`, `cd`, `chattr`, `chmod`, `chown`, `compress`, `cp`, `dd`, `fdisk`, `find`, `gzip`, `ln`, `mkdir`, `mksfs`, `mount`, `mv`, `rm`, `rmdir`, `rpm`, `sort`, `swapon`, `swapoff`, `tar`, `touch`, `umount`, `uncompress`, `uniq`, `unzip`, `zip`
- ▶ **Managing running programs**—`bg`, `fg`, `kill`, `killall`, `nice`, `ps`, `pstree`, `renice`, `top`, `watch`

- ▶ **Getting information**—`apropos`, `cal`, `cat`, `cmp`, `date`, `diff`, `df`, `dir`, `dmesg`, `du`, `env`, `file`, `free`, `grep`, `head`, `info`, `last`, `less`, `locate`, `ls`, `lsattr`, `man`, `more`, `pinfo`, `ps`, `pwd`, `stat`, `strings`, `tac`, `tail`, `top`, `uname`, `uptime`, `vdir`, `vmstat`, `w`, `wc`, `whatis`, `whereis`, `which`, `who`, `whoami`
- ▶ **Console text editors**—`ed`, `jed`, `joe`, `mcedit`, `nano`, `red`, `sed`, `vim`
- ▶ **Console Internet and network commands**—`bing`, `elm`, `ftp`, `host`, `hostname`, `ifconfig`, `links`, `lynx`, `mail`, `mutt`, `ncftp`, `netconfig`, `netstat`, `pine`, `ping`, `pump`, `rdate`, `route`, `scp`, `sftp`, `ssh`, `tcpdump`, `traceroute`, `whois`, `wire-test`

References

- ▶ <https://help.ubuntu.com/community/UsingTheTerminal>—The Ubuntu community help page for using the terminal.
- ▶ <https://help.ubuntu.com/community/LinuxFilesystemTreeOverview>—The Ubuntu community help page for and overview of the Linux file system tree.
- ▶ <https://help.ubuntu.com/community/RootSudo>—An Ubuntu community page explaining `sudo`, the philosophy behind using it by default, and how to use it.

This page intentionally left blank

Index

Symbols

- ./configure, building Apache, 465
- .htaccess configuration files (Apache web server configuration), 473-474
- /bin directory commands, 141-143
- /etc directory configuration files, 143
- /etc/host.conf file, 363
- /etc/hosts file, adding hosts, 361
- /etc/init.d/apache2, starting/stopping Apache web server, 468-469
- /etc/modprobe.conf file
 - editing, 354
 - manually loading kernel modules, 354-355
- /etc/nsswitch.conf file, 362
- /etc/resolv.conf file, 362-363
- /etc/samba/smb.conf file, 444-447
- /etc/services file, service settings, 361-362
- /home directory, user directories, 143-144
- /proc directory, interacting with kernel, 144-145
- /proc file system, 292
- /sbin directory commands, 141-143
- /tmp directory, temporary file storage, 146
- /usr directory
 - shared data, 145-146
 - subdirectories, 35
- /usr/bin subdirectory, 35
- /usr/include subdirectory, 35
- /usr/lib subdirectory, 35
- /usr/lib/modules subdirectory, 35
- /usr/src/linux-3.2 directory, 419
- /var directory data files, 146
- 10BASE-T, 349

64-Bit Ubuntu, 793
 100BASE-T, 349-350
 1000BASE-T, 350

A

AbiWord (GNOME Office component), 70
 ac command, 222, 234
 accept command, 460
 access, databases
 local GUI clients, 563-564
 SSH, 562
 access control, Apache web server, 474-480
 access control lists (ACLs), 529-533
 accessing
 command line, 135-138
 Perl shell, 694
 accounts, Launchpad, 668
 ACID compliance, MySQL *versus* PostgreSQL, 553-554
 ACLs (access control lists), 529-533
 activation, DHCP, 366-367
 Ada, 768
 Adblock Plus plug-in, 53
 adding users, 218-221
 Additional Drivers manager, 109
 address-based virtual hosts (Apache), 486
 addressing
 broadcast, 348
 IPv4, 343
 IPv6, 344-347
 multicasts, 348
 TCP/IP, 341-342
 Unicast, 348
 adduser command, 164
 adjusting volume, music and sound, 77
 administration, LDAP, 587-588
 administrative tools, BIOS (basic input/output system), 286-287
 Adobe Flash, 96
 Adobe Photoshop, 84
 ADT (Android Development Tools) plug-in, 781
 ADT Eclipse plug-in, installation, 782
 Advanced Linux Sound Architecture (ALSA), 76
 Advanced Package Tool (APT), 123-128, 603
 afio backup tool, 325
 aliases, forwarding email, 516-517
 Alien Arena, 110
 all-in-one devices (print/fax/scan), 459
 allow directive, Apache web server access control, 476-477
 AllowOverrides directive, Apache web server configuration, 474
 Alpine mail client, 58
 ALSA (Advanced Linux Sound Architecture), 76
 Amanda backup application, 324-325
 AMD, proprietary drivers, 108
 American Registry for Internet Numbers, 341
 Android, mobile development for, 779-785
 Android Runtime, 780
 Application Framework, 780-781
 core applications, 781
 creating application, 784-785
 installation of Android SDK, 781-783
 libraries, 780
 Linux kernel, 780
 Android Development Tools (ADT) plug-in, 781
 Android Runtime, mobile development for Android, 780
 Android SDK, installation, 781-783
 Android Virtual Device (AVD), 783
 anonymous servers (FTP), 498
 Apache, performance tuning, 411-412
 Apache Module Registry, 480

- Apache Software Foundation, 462
- Apache Tomcat, 496
- Apache Web Server, 461-490, 499
 - access control, 461-490
 - Apache package directories, 463-464
 - directives, 489
 - file system authentication, 474-480
 - installation, 462-466
 - building the source, 464-466
 - Ubuntu repositories, 463-464
 - logging, 488-490
 - modules, 480-486
 - disabling, 481
 - enabling, 481
 - quick guide setup, 466
 - runtime server configuration settings, 469-474
 - directives, 470
 - editing apache2.conf, 470-473
 - .htaccess configuration file, 473-474
 - multiprocessing modules, 473
 - starting/stopping, 467-469
 - user files, 478-479
 - virtual hosting, 486-488
 - wide use, 461
- apache2.conf, editing, 470-473
- AppArmor, 401-403
- Application Framework, mobile development for Android, 780-781
- applications
 - Internet, 51-64
 - Chromium, 53-54
 - email clients, 55-58
 - Firefox, 52-53
 - Google Chrome, 53-54
 - mobile development for Android, 781
 - multimedia, 75-97
 - burning CDs/DVDs, 89-93
 - digital cameras, 88
 - graphics manipulation, 83-87
 - sound and music, 75-81
 - viewing video, 94-97
 - productivity
 - GNOME Office, 69-70
 - KOffice, 69, 70-71
 - LaTeX, 73
 - LibreOffice, 66-68
 - Microsoft Windows, 73-74
 - PDF, 71-72
 - XML and DocBook, 71-72
- apply-patch tool (Bikeshed), 661
- apropros command, 140
- APT (Advanced Package Tool), 123-128
 - day-to-day usage, 124-127
 - finding software, 127-128
- apt-get autoclean command, 126
- apt-get clean command, 126
- apt-get dist-upgrade command, 125
- apt-get install command, 125
- apt-get remove command, 127
- apt-get update command, 124
- arithmetic operators (Perl), 686
- ark archiving tool (KDE), 320-322
- ARM processors, 780
- ARPANET, 51
- array_keys() function (PHP), 744
- array_unique() function (PHP), 743
- array_values() function (PHP), 744
- arrays
 - Perl, 685
 - PHP functions, 743-745
 - PHP programming, 725-727

Artistic License, 3
 asort() function (PHP), 744
 assessment
 backup needs, 309-310
 resources for backup, 309-310
 vulnerability, 393-394
 assigning permissions, 150-151
 at command, scheduling tasks, 237-240
 audio formats, 79
 authenticated servers (FTP), 498
 authentication, Apache web server, 477-479
 autoconf utility, 759-760
 autocracking scripts, 392
 automation of tasks, 237-242
 scheduling tasks, 237-242
 writing shell scripts, 255-256
 Autoresponders, 522
 AVD (Android Virtual Device), 783
 AVI (video format), 95

B

Back in Time backup tool, 322-323
 background processing, shell control, 247
 backing up data, 307-334
 choosing a strategy, 307-314
 copying files, 326-330
 hardware and media, 314-316
 software, 316-325
 system rescue, 332-334
 version control, 330-332
 backslash, writing shell scripts, 260
 backtick, writing shell scripts, 260
 backticks, accessing the shell in Perl, 694
 badblocks command, 409
 Bandwidth Meter, 53

Banshee, 81-82
 Base (LibreOffice component), 67
 BaseX, 576
 Bash shell, configuration, 669-670
 Basic Authentication, Apache web server, 477
 basic input/output system. See BIOS
 batch command, scheduling tasks, 237-240
 Battle for Wesnoth, 112
 Bazaar, 647-649
 bch tool (Bikeshed), 661
 Berkeley DB, 572
 Bernes-Lee, Tim, 51
 BigTable, 577
 Bikeshed, 661-663
 BIOS (basic input/output system), 282-288
 booting into default runlevel, 285
 controlling services, 286-287
 final stage of initialization, 285-286
 init scripts, 285-286
 loading Linux Kernel, 283
 runlevel definitions, 284
 system services and runlevels, 284
 troubleshooting runlevel problems, 287-288
 tuning disk drives, 406-407
 Bitbucket, 649
 BMP (graphics format), 86
 body of message (newsgroup articles), 63
 Boot Loader, Ubuntu installation, 10-11
 boot loader problems, compiling kernel, 436
 boot process, 281-290
 BIOS, 282-288
 manually starting/stopping services, 288-289
 running services, 281-282
 Upstart, 289-290
 booting into default runlevel, BIOS (basic input/output system), 285
 bootmail tool, 663

Brasero, burning CDs/DVDs, 89
 break statement, executing shell scripts, 278
 bridged networking, 601
 bridges, 353
 broadcast addressing, 348
 BSD License, 3
 buffer usage, MySQL, 412-413
 Bug Squad, 677
 bugs
 fixing, 670-672
 Harvest, 673
 built-in security, kernel, 391
 built-in variables, executing shell scripts, 257
 BulletProofX, 36
 bunzip2 command, 204
 burning CDs/DVDs, 89-93
 Brasero, 89
 command line, 89-93
 buying music, Ubuntu One Music Store, 81
 Bynari, 524
 Byobu, 60
 byobu command, 205-206
 bzip2 command, 204
 bzip tool (Bikeshed), 661

C

C programming tools, 755-763
 autoconf utility, 759-760
 debugging tools, 760-761
 GNU C compiler, 761-762
 graphical development, 762-763
 Linux, 755-757
 macros, 758-759
 make command, 757
 makefile targets, 759
 makefiles, 757-759
 C++ programming tools, 755-763
 autoconf utility, 759-760
 debugging tools, 760-761
 GNU C compiler, 761-762
 graphical development, 762-763
 Linux, 755-757
 macros, 758-759
 make command, 757
 makefile targets, 759
 makefiles, 757-759
 c10k problem, 492
 cable (network), 351-352
 Calc (LibreOffice component), 67
 cameras, digital, 88
 cancel command, 460
 captured screen images, 87
 case statement, executing shell scripts, 276-278
 Cassandra, 572-573
 cat command, 144-145, 159, 171-173
 cd command, 148-149, 171, 173-175
 Cd-RW drives, 315
 CDs, burning, 89-93
 Brasero, 89
 command line, 89-91
 Cedega, 116
 CFEngine, managing sets of servers, 643
 change command, 234
 changing
 passwords in a batch, 227
 runlevels, 286-287

characters

- shell pattern matching, 245

- writing shell scripts, 257

charms (juju), 638-640

Checkbox, 677

checking connections, networking, 338-340

Chef, managing sets of servers, 642

Cherokee, 494-495

chfn command, 234

chgrp command, 153, 212, 234

children's games, 114-115

Childsplay, 114

chmod command, 152-153, 171, 175, 212, 234

chown command, 212, 234

chpasswd command, 227, 234

Chromium, 53-54

chsh command, 218, 234

class inheritance, Python object orientation, 719-720

class variables, Python object orientation, 717-718

classes of networks, 341-342

Claws mail client, 57

CLI (command-line interface). *See* command line

client configuration

- NFS, 442

- proxy servers, 528-337

client/server system, relational database services, 561-567

clients

- graphical, 566-567

- IP addresses, 533-534

- LDAP configuration, 586-587

Clojure, 768-769

Cloud, 607-640

- benefits, 607-610

- Eucalyptus, 610-618

- deployment/installation, 612

- euca2ools primer, 616-618

IaaS (Infrastructure as a Service), 609

juju, 634-640

- charms, 638-640

- installation, 635-637

Landscape, 640

MaaS (Metal as a Service), 610

OpenStack, 618-634

- commands, 634

- Compute Infrastructure (Nova), 618-619

- creating an image, 629-632

- Imaging Service (Glance), 619

- installation, 619-629

- instance management, 632

- network management, 633-634

- Storage Infrastructure (Swift), 619

- storage management, 633

PaaS (Platform as a Service), 609

SaaS (Software as a Service), 609

cloud-sandbox tool (Bikeshed), 661

Cloud storage, 64, 316

COBOL (Common Business Oriented Language), 769

code

- configuring, 759-760

- packaging, 670-672

CodeWeavers, 74

coll tool (Bikeshed), 661

combining commands, 195-197

comm command, 194-195

command line, 133-206

- accessing, 135-138

- APT

- day-to-day usage, 124-127

- finding software, 127-128

- burning CDs/DVDs, 89-93
 - commands, 171-191
 - cat command, 172-173
 - cd command, 173-175
 - chmod command, 175
 - combining, 195-197
 - cp command, 175-177
 - du command, 176-177
 - find command, 177-180
 - grep command, 179-180
 - less command, 180-182
 - ln command, 182-183
 - locate command, 184
 - ls command, 184-186
 - man command, 186-187
 - mkdir command, 187
 - mv command, 187
 - ps command, 188
 - rm command, 188-189
 - tail command, 189
 - top command, 189-191
 - which command, 191
 - common commands and programs, 166-167
 - comparing files, 194-195
 - compressed files, 204-205
 - defined, 134-135
 - directories
 - changing with cd command, 148-149
 - listing contents with ls command, 146-148
 - pwd command, 149
 - environment variables, 197-200
 - errors in Perl, 683
 - files
 - copying files, 158
 - creating directories, 155-156
 - creating files with touch command, 155
 - deleting directories, 156
 - deleting files, 157
 - displaying file contents, 159
 - moving/rename files, 157-158
 - wildcards/regular expressions, 159
 - Linux file system hierarchy, 141-146
 - logging in/out from a remote computer, 137-138
 - logging out, 137
 - multiple terminals, 205-206
 - MySQLclient, 564-566
 - navigating Linux file system, 146-149
 - network interface configuration, 356-360
 - permissions, 149-155
 - assigning permissions, 150-151
 - chmod command, 152-153
 - directory permissions, 151-152
 - file permissions, 153
 - set UIDs/GID permissions, 153-155
 - PostgreSQL client, 566
 - reading documentation, 140-141
 - reasons for use, 170-171
 - redirecting input and output, 191-193
 - redirection of streams, 193
 - root users, 160-166
 - scripting. See Python
 - shell control, 243-244
 - text editors, 200-204
 - emacs, 203-204
 - nano, 201-202
 - vi, 202-203
 - user accounts, 138-139
 - working with files, 155-159
- command-line interface (CLI). See command line
- command prompt. See command line

commands

- ac, 222
- adduser, 164
- AppArmor, 403
- apropros, 140
- apt-get autoclean, 126
- apt-get clean, 126
- apt-get dist-upgrade, 125
- apt-get install, 125
- apt-get remove, 127
- apt-get update, 124
- at, 237-240
- badblocks, 409
- batch, 237-240
- /bin directory, 141-143
- byobu, 205-206
- cat, 144-145, 159
- cd, 148-149
- chgrp, 153, 212
- chmod, 152-153, 212
- chown, 212
- chpasswd, 227
- chsh, 218
- comm, 194-195
- command line, 166-167, 171-191
 - cat command, 172-173
 - cd command, 173-175
 - chmod command, 175
 - combining, 195-197
 - cp command, 175-177
 - du command, 176-177
 - find command, 177-180
 - grep command, 179-180
 - less command, 180-182
 - ln command, 182-183
 - locate command, 184
 - ls command, 184-186
 - man command, 186-187
 - mkdir command, 187
 - mv command, 187
 - ps command, 188
 - rm command, 188-189
 - tail command, 189
 - top command, 189-191
 - which command, 191
- compression of files, 204-205
- cp, 158, 327-328
- cron, 240-242
- date, 28-29
- deluser, 164
- diff, 194
- dmesg, 31
- dropuser (PostgreSQL), 560
- e2fsck, 409
- edquota, 233
- emacs text editor, 204
- env, 198
- Eucalyptus, 616-618
- exit, 137
- find, 154
- gpsswd, 215
- gprof, 761
- groupadd, 215
- groupdel, 215
- groupmod, 215
- grpck, 215
- hdparm, 407-408
- hdx=ide-scsi, 406
- hwclock, 29
- idebus=xx, 406
- idex=autotune, 406
- idex=dma, 406
- ifconfig, 341, 356-360, 397
- init, 284

- Internet connectivity, 380
- kernel module management, 422-423
- kill, 293-294
- less, 140, 159
- logout, 137
- ls, 146-148, 149-150
- make, 757
- man, 140
- mc, 328-330
- mkdir, 151, 155-156
- mv, 157-158
- mysql -u root, 555
- nice, 294
- OpenStack, 634
- passwd, 217
- patch, 428
- pci=biosirq, 407
- Perl, 700
- ping, 338-340
- postconf, 513
- printenv, 198, 229
- printing, 460
- pwd, 137, 149
- Quickly, 654
- quotacheck, 233
- quotaoff, 233
- quotaon, 233
- rcp, 384
- relational database services, 567
- renice, 294
- repquota, 233
- rm, 157
- rmdir, 156
- route, 356-360
- rsync, 328-330
- /sbin directory, 141-143
- scp, 384
- sftp, 383, 385
- shutdown, 18, 165-166
- smbclient, 449
- smbstatus, 448
- splint, 760-761
- ssh-keygen, 385-387
- su, 227-229
- sudo, 22-23, 160-163, 210
- tar, 326-327
- testparm, 447-448
- time, 294
- top, 295
- touch, 149, 155, 386
- tune2fs, 408-409
- UFW (Uncomplicated Firewall), 399
- umask, 149
- uptime, 297
- user accounts, 234
- useradd, 215-216
- usermod, 215-218
- vi text editor, 202-203
- whereis, 141
- comments, PHP programming, 729
- commercial games, 115-116
- Common Business Oriented Language (COBOL), 769
- Common UNIX Printing System (CUPS), 454
- CommuniGate Pro, 523
- community teams, testing, 675-677
- comparison operators (Perl), 686-687
- compiling applications from source, software management, 128-130
- compiling the kernel, 427-436
- Compiz, 100
- compound operators (Perl), 686
- Comprehensive Perl Archive Network (CPAN), 695
- compressed files, 204-205

- Compute Infrastructure (Nova) service, OpenStack, 618-619
- computer attacks, hacker versus cracker, 392
- conditional statements
 - Perl, 689-690
 - PHP programming, 733-734
 - Python, 713-715
- configuration
 - Apache web server, 469-474
 - .htaccess configuration files, 473-474
 - directives, 470
 - editing apache2.conf, 470-473
 - multiprocessing modules, 473
 - Bazaar, 669
 - clients, proxy servers, 528-337
 - code, 759-760
 - DHCP, 367-368
 - DHCP network hosts, 369-371
 - Dial-Up access, 378-379
 - Digital Subscriber Line (DSL) access, 376-378
 - Fetchmail, 517-521
 - FireFox, 528
 - firewalls, 398
 - graphical tools, 363-364
 - kernel, 427-435
 - LDAP clients, 586-587
 - LDAP server, 582-585
 - local Bash shell, 669-670
 - loopback interface, 336-337
 - management, 130-131
 - MySQL, 554-558
 - networking tools, 355-364
 - NFS client, 442
 - NFS server, 440-442
 - Postfix, 512-517
 - Postfix masquerading, 514
 - PostgreSQL, 558-561
 - PPPoE, 377-378
 - quotas, 233-234
 - Samba, 444-447, 450-453
 - software repositories, 23-26
 - system settings, 26-29
 - printers, 26-27
 - time and date, 27-29
 - Tripwire, 396-397
 - Unity, 48-49
 - Very Secure FTP server, 502-505
 - wireless networks, 29-30
- configuration files
 - /etc directory, 143
 - .htaccess, 473-474
 - networking, 360-361
 - version control, 330-332
- Conky, 300-305
- console. See command line
- console-based monitoring tools, 291-298
 - disk quotas, 298
 - disk space, 297-298
 - free and used memory, 296-297
 - kill command, 293-294
 - priority scheduling and control, 294-296
- constants
 - Perl string constants, 686
 - PHP programming, 728
- constructors, Python object orientation, 718-719
- control structures, Perl, 690-693
- controlling services, BIOS (basic input/output system), 286-287
- convert utility (ImageMagick), 86
- copying files, 326-330
 - cp command, 327-328
 - Midnight Commander, 328-330
 - rsync command, 328-330
 - tar command, 326-327

core applications, mobile development for
 Android, 781
 CouchDB, 575
 cp command, 158, 171, 175-177, 327-328
 CPAN (Comprehensive Perl Archive
 Network), 695
 CREATE DATABASE statement (MySQL), 556
 CREATE DATABASE statement
 (PostgreSQL), 559
 creating
 MySQL database, 556-558
 OpenStack images, 629-632
 PostgreSQL database, 559
 cron command, running jobs repeatedly,
 240-242
 Crossover Games, 116
 CrossOver Office, 74
 Cube 2: Sauerbraten, 110
 CUPS (Common UNIX Printing System), 454,
 456-458
 custom tools, Eucalyptus, 616-618
 customizing Unity, 48-49
 CustomLog directive (Apache), 489

D

Dalvik, 780
 Dash, 19
 Dash (Unity desktop), 44-47
 data
 mirroring, 314
 retrieval from databases, 550-552
 data backup, 307-334
 choosing a strategy, 307-314
 copying files, 326-330
 hardware and media, 314-316
 software, 316-325
 system rescue, 332-334
 version control, 330-332
 data directory (PostgreSQL), initializing,
 558-559
 data integrity, MySQL *versus* PostgreSQL,
 553-554
 data locking, MySQL *versus* PostgreSQL,
 552-553
 data loss, 308-309
 data structures (Perl), 684-686
 database administrators (DBAs), 543-544
 databases
 NoSQL, *See* NoSQL databases
 PHP programming, 751-754
 relational database services, 543-567
 client/server system, 561-567
 commands, 567
 comparison of MySQL and PostgreSQL,
 552-554
 creating tables, 548-549
 how they work, 545-547
 inserting data in tables, 549-550
 MySQL, 554-558
 PostgreSQL, 558-561
 retrieving data, 550-552
 SQL basics, 547-548
 date, configuring system settings, 27-29
 date command, 28-29
 day job crackers, 392
 day-to-day usage (APT), 124-127
 DBAs (database administrators), 543-544
 Debian, 23
 debugging tools, 760-761
 default runlevel, BIOS (basic input/output
 system), 285
 definitions
 Python functions, 715
 Runlevels, BIOS (basic input/output
 system), 284

Deja Dup backup tool, 320-322

deleting users, PostgreSQL, 560-561

deluser command, 164

deny directive, Apache web server access control, 476-477

deployment, Eucalyptus, 612

Desktop Couch, 653

Desktop DVD, 8

desktop environment, interfaces, 100

destructors, Python object orientation, 718-719

development, 665-673

 finding bugs with Harvest, 673

 fixing bugs and packaging, 670-672

 installation packages, 667-670

 MOTU (Masters of the Universe), 673

 opportunistic (See opportunistic development)

 six-month cycle, 666-667

Device section (xorg.conf file), 37, 39-40

devices, security, 397

DevOps *versus* SysAdmin, 608

DHCP (Dynamic Host Configuration Protocol), 365-371, 594

 configuring network hosts, 369-371

 how it works, 365-366

 installation and activation, 366-367

 server, 368

 software installation and configuration, 367-368

 uses, 371

Dia (LibreOffice component), 68

Diagnostics, 53

Dial-Up access, Internet connectivity, 378-379

dictionaries, Python, 712-713

diff command, 194

digital cameras, 88

Digital Subscriber Line (DSL) access, 376-378

directives

 Apache web server

 access control, 476-477

 configuration, 470

 CustomLog, 489

 satisfy, 479

directories

 Apache package, 463-464

 command line

 changing with cd command, 148-149

 listing contents with ls command, 146-148

 pwd command, 149

 Linux, 142

 Linux source tree, 419-421

Directory Information Tree (DIT), 582

directory permissions, 151-152

DirectoryIndex directive, Apache web server configuration, 473

disable command, 460

disabling

 Apache modules, 481

 file access time, 409

disaster recovery plan, 403-404

disk drives, tuning, 406-407

disk quotas, 232-234, 298

disk space, console-based monitoring tools, 297-298

display manager, X Server, 41-42

dist-upgrade option, 22

DIT (Directory Information Tree), 582

dman tool (Bikeshed), 662

dmesg command, 31

do...until looping construct (Perl), 692

do...while loop (PHP), 739

do...while looping construct (Perl), 692

DocBook, 71-72

document stores, NoSQL databases, 574-576
documentation, command line, 140-141
DocumentRoot directive, Apache web server configuration, 472
documents, kernel programmers, 419-420
dotdee, 130-131
Draw (LibreOffice component), 67
dropuser command (PostgreSQL), 560
DSL (Digital Subscriber Line) access, 376-378
du command, 171, 176-177
DVD installation jump start, 8
DVD+RW/-RW drives, 315
DVDs
 burning, 89-93
 Brasero, 89
 command line, 91-93
 formats, 91
Dynamic Host Configuration Protocol. See DHCP

E

e2fsck command, 409
Eclipse, 772
Eclipse Foundation project, Jetty, 495
editing
 /etc/modprobe.conf file, 354
 apache2.conf, 470-473
 video, 97
editing commands, emacs text editor, 200-203-204
edquota command, 233
Edubuntu, 590
elements, xorg.conf file, 36-41
 Device section, 37, 39-40
 Files section, 36-38
 InputDevice section, 36, 38-39
 Module section, 36, 38
 Monitor section, 36, 39
 Screen section, 37, 40-41
 ServerLayout section, 36-37
emacs command, 171
emacs text editor, 200, 203-204
email, 507-524
 alternatives to Microsoft Exchange Server, 522-524
 Fetchmail, 517-521
 how email is sent and received, 507-512
 mail delivery agents, 521-522
 Postfix configuration and operation, 512-517
email clients, 55-58
 Evolution, 56-57
 Mozilla Thunderbird, 56
Empathy, 59
EmulatorsEmulators, 107
enable command, 460
enabling Apache modules, 481
endless loops, shell programs, 272
Enlightenment, 100
entering PHP mode, 724
enterprise servers, monitoring tools, 305-306
env command, 198
environment, Launchpad, 668-670
environment variables, command line, 197-200
Erlang, 770
errors, compiling kernel, 435-433
escape sequences, PHP programming, 729-730
ethereal, 305
etiquette, IRC (Internet Relay Chat), 62
euca2ools primer, Eucalyptus, 616-618
Eucalyptus, 607, 610-618
 deployment/installation, 612
 euca2ools primer, 616-618

Evolution, 56-57, 70, 586

executing shell scripts, 248-279

- accessing variable values, 253
- assigning value to variables, 252-253
- automation of tasks, 255-256
- backslash, 260
- backtick, 260
- break statement, 278
- built-in variables, 257
- case statement, 276-278
- comparison of expressions, 261-266
- comparison of expressions with tcsch, 266-270
- exit statement, 279
- for statement, 270-271
- if statement, 275-276
- interpreting shell scripts, 250-252
- positional parameters, 253-254
- repeat statement, 274
- running shell program, 249-250
- select statement, 274-275
- shift statement, 275
- sorting scripts for access, 250
- special characters, 257
- strings with embedded spaces, 257-259
- unexpanded variables, 259-260
- until statement, 273-274
- variables, 252
- while statement, 271-273

execution operator, 735

Exim, 509

exit command, 137

exit statement, executing shell scripts, 278

exiting PHP mode, 724

expressions, Perl regular expressions, 693

extensions, Firefox, 53

external attacks, 392

extract() function (PHP), 745

Extraversion level (kernel), 425

F

fclose() function (PHP), 747

Fetchmail, 517-521

- configuration, 517-521
- installation, 517
- user accounts, 519-521

fiber optic cable, 352

fiber optics, 350

file access time, disabling, 409

file operators, writing shell scripts, 264-265, 268-269

file permissions, 153, 212-213

file system, Linux hierarchy, 141-146

file system authentication, Apache web server, 474-480

file system settings, tuning, 408

File Systems tab (System Monitor), 298

File Transfer Protocol. See FTP

file_get_contents() function (PHP), 745

file_put_contents() function (PHP), 745

files

- Apache file locations after install, 465-466
- command line, 155-159
 - copying files, 158
 - creating directories, 155-156
 - creating files with touch command, 155
 - deleting directories, 156
 - deleting files, 157
 - displaying file contents, 159
 - moving/renaming files, 157-158
 - wildcards/regular expressions, 159

- comparison, 194-195
- compressed, 204-205
- configuration, version control, 330-332
- copying, 326-330
 - cp command, 327-328
 - Midnight Commander, 328-330
 - rsync command, 328-330
 - tar command, 326-327
- /etc/host.conf, 363
- /etc/hosts, adding hosts, 361
- /etc/nsswitch.conf, 362
- /etc/resolv.conf, 362-363
- /etc/services, service settings, 361-362
- ftphosts, 505-506
- .htaccess configuration, Apache web server configuration, 473-474
- PHP functions, 745-747
- restoring from an archive, 318-319
- saving from nonbooting hard drive, 333-334
- sharing, 439-453
 - NFS (Network File System), 440-442
 - Samba, 450-453
- Ubuntu installation, 19
- Files section (xorg.conf file), 36-38
- filesize() function (PHP), 747
- find command, 154, 171, 177-180
- Firefox, 52-53
 - configuration, 528
 - RSS feeds, 58
- firewalls, configuring, 398
- first-person shooter (FPS) games, 110
- fixing bugs, 670-672
- FLAC (sound format), 78
- Flash, 53
- Flash (Adobe), 96
- flavors (Ubuntu), 8
- flexbackup tool, 325
- FlightGear, 114-115
- FlockDB, 578
- FLV (video format), 95
- fopen() function (PHP), 746
- for loop
 - Python, 714
 - PHP, 737
- for looping construct (Perl), 690
- for statement, writing shell scripts, 270-271
- foreach loop (PHP), 738
- foreach looping construct (Perl), 691
- ForecastFox, 53
- formats
 - DVDs, 91
 - graphics manipulation, 85-87
 - sound, 78-79
 - video, 95
- Forth, 770
- Fortran, 771
- forwarding email, 516-517
- FPS (first-person shooter) games, 110
- FQDN (fully qualified domain name), 582
- fread() function (PHP), 747
- free and used memory, console-based monitoring tools, 296-297
- Frets on Fire, 114
- Frozen Bubble, 111
- FTP (File Transfer Protocol), 497-506
 - configuring Very Secure FTP server, 502-505
 - ftphosts file, 505-506
 - server selection, 497-499
 - servers, 498-499
 - software installation, 499-500
 - users, 500-502

ftphosts file, 505-506

full backups

incremental backups, 313

periodic basis, 313

tar backup tool, 317-318

fully qualified domain name (FQDN), 582

functions

PHP programming, 740-751

arrays, 743-745

files, 745-747

miscellaneous functions, 747-750

strings, 740-743

Python, 715-716

shell scripts, 279

fwrite() function (PHP), 747

G

games, 107-116

Battle for Wesnoth, 112

children, 114-115

commercial, 115-116

FlightGear, 114-115

Frets on Fire, 114

Frozen Bubble, 111

installation, 109-116

proprietary video drivers, 108-109

Scorched 3D, 110-111

Speed Dreams, 114-115

SuperTux, 112-113

Warsow, 110

Windows, 116

gcc (GNU Compiler Collection), 755

gcc command, 171

gCompris, 114

gdb tool, 761

gecos field, 223

gedit, 653

gedit text editor, 201

Genprof, 402

GHC (Glorious Glasgow Haskell Compilation system), 772

GIDs (group IDs), 153-155, 212

GIF (graphics format), 86

Gigabit Ethernet, 350

Gilt, 649-650

GIMP (GNOME Office component), 70

GIMP (GNU Image Manipulation Program), 83-84

GitHub, 650

Glade, 653

Glade client, GNOME, 763-764

Glance (Imaging Service), OpenStack, 619

global behavior, Samba, 446

global options, configuring Fetchmail, 518

Glorious Glasgow Haskell Compilation system (GHC), 772

Gnat Programming System, 768

Gnobuntu, 104-105

GNOME, Glade client, 763

GNOME file roller, 319

gnome-nettool, 305

gnome-screenshot, 87

gnome-shell, 105

GNOME3, 104-105

GNU C compiler, 761-762

GNU Compiler Collection (gcc), 755

GNU Fortran 95 compiler, 771

GNU Image Manipulation Program (GIMP), 83-84

GNU Privacy Guard, 667

GNU Project, 133

GNU/Linux, 418

Gnumeric (GNOME Office component), 70

Go, 771
 Google Chrome, 53-54
 gpasswd command, 215
 GPG key, 667-668
 gprof command, 761
 Grand Unified Boot Loader (GRUB2), 10-11
 GRANT statement (PostgreSQL), 561
 granting privileges, PostgreSQL databases, 561
 graph stores, NoSQL databases, 577-578
 graphical clients, relational database services, 566-567
 graphical configuration tools, 363-364
 graphical development tools, 762-763
 graphical process, monitoring tools, 298-305
 graphical user interfaces (GUIs), Unity desktop, 33-49
 graphics manipulation, 83-87

- formats, 85-87
- GNU Image Manipulation Program (GIMP), 83-84
- Photoshop, 84
- scanners, 85
- screen images, 87

 grep command, 171, 179-180
 Groklaw, 59
 Groovy, 771
 Ground Control, 657-661
 Group directive, Apache web server configuration, 471
 group IDs (GIDs), 212
 group listing, 213-214
 group permissions, 212
 groupadd command, 215
 groupdel command, 215
 groupmod command, 215
 groups, management, 213-216
 groups command, 234

grpck command, 215
 GRUB2 (Grand Unified Boot Loader), 10-11
 GRUB2 boot loader, 333
 GStreamer, 653
 GTK, 653
 GTK widget set, 69-70
 GUPFW, 398
 GUIs (graphical user interfaces), Unity desktop, 33-49
 gunzip command, 204
 gzip command, 204

H

hackers, crackers *versus*, 392
 handheld digital cameras, 88
 hard disk, performance tuning, 405-396

- badblocks command, 409
- disabling file access time, 409
- e2fsck command, 409
- file system settings, 408
- hdparm command, 407-408
- tune2fs command, 408-409
- tuning disk drives, 406-407

 hardware

- data backup, 314-316
- networking, 349-355
 - hubs and switches, 352
 - initializing new hardware, 353-355
 - network cable, 351-352
 - NIC (Network Interface Cards), 349-351
 - routers and bridges, 353
- TV and video, 94-95

 hardware specifications, 8
 Harvest, 673

- hashes (Perl), 685
- Haskell, 772
- HBase, 577
- HDLC (high-level data link control), 378
- hdparm command, 407-408
- hdx=ide-scsi command, 406
- header lines (newsgroup articles), 63
- hibernate, 14
- hierarchy, Linux file system, 141-146
- high-level data link control (HDLC), 378
- history
 - Internet, 51
 - LibreOffice, 68
- home directories, 446-447
- home users, backup programs, 311
- Horde, 524
- hosts, 361
- HTML (Hypertext Markup Language), 51
- HTML forms, 751
- HTTP (Hypertext Transfer Protocol), 51
- HTTP servers, 491-496
 - Apache Tomcat, 496
 - Cherokee, 494-495
 - Jetty, 495
 - lighttpd, 493
 - Nginx, 491-493
 - thttpd, 495-496
 - YAWS, 494
- hubs, 352
- Humble Indie Bundle, 116
- hwclock command, 29
- HyperGraphDB, 578
- Hypertext Markup Language (HTML), 51
- Hypertext Transfer Protocol (HTTP), 51

- l
- laaS (Infrastructure as a Service), 609
- idebus=xx command, 406
- idex=autotune command, 406
- idex=dma command, 406
- if statement, executing shell scripts, 275-276
- if/else conditional statements (Perl), 689-690
- ifconfig command, 341, 356-360, 397
- ImageMagick convert utility, 86
- images, OpenStack, 629-632
- Imaging Service (Glance), OpenStack, 619
- IMAP (Internet Message Access Protocol), 511
- implementation, quotas, 233
- Impress (LibreOffice component), 67
- in_array() function (PHP), 744
- incremental backups, 317-318
- Infrastructure as a Service (laaS), 609
- init command, 284
- init scripts, BIOS (basic input/output system), 285-286
- initialization
 - BIOS (basic input/output system), 285-286
 - data directory (PostgreSQL), 558-559
 - network hardware, 353-355
- input (shell control), 246-247
- InputDevice section (xorg.conf file), 36, 38-39
- inserting data in tables, relational database services, 549-550
- installation
 - ADT Eclipse plug-in, 782
 - Android SDK, 781-783
 - Apache web server, 462-466
 - CPAN module (Perl), 697
 - development packages, 667-670

- DHCP, 366-368
- Eucalyptus, 612
- Fetchmail, 517
- FTP software, 499-500
- games, 109-116
- Ground Control, 657
- Java, 781
- juju, 635-637
- LTSP, 593-594
- NFS, 440
- OpenStack, 619-629
- proprietary video drivers, 108-109
- SDK, 781
- Squid, 528
- Ubuntu, 7-32
 - post-installation configuration problems, 31-32
 - preparation, 7-11
 - programs and files, 19
 - shutting down, 18
 - software repositories, 23-26
 - Software Updater, 19-22
 - step-by-step, 11-17
 - sudo command, 22-23
 - system settings, 26-29
 - wireless network configuration, 29-30
- virtual devices, 783
- instance management, OpenStack, 632
- instant messaging, Empathy, 59
- interfaces, 99-105
 - desktop environment, 100
 - GNOME3 and Gnobuntu, 104-105
 - KDE and Kubuntu, 101-102
 - LXDE and Lubuntu, 103-104
 - Xfce and Xubuntu, 102-103
- internal attacks, 392
- Internet applications, 51-64
 - Chromium, 53-54
 - email clients, 55-58
 - Evolution, 56-57
 - Mozilla Thunderbird, 56
 - Empathy, 59
 - Firefox, 52-53
 - Google Chrome, 53-54
 - Internet Relay Chat (IRC), 60-61
 - RSS feeds, 58-59
 - Firefox, 58
 - Liferea, 58-59
 - Ubuntu One cloud storage, 64
 - Usenet newsgroups, 61-64
- Internet connectivity, 374-380
 - commands, 380
 - common configuration information, 374-375
 - Dial-Up access, 378-379
 - DSL access, 376-378
 - troubleshooting connection problems, 379-380
- Internet Message Access Protocol (IMAP), 511
- Internet Relay Chat (IRC), 60-61, 805
- Internet resources, 797-805
- interpreting shell scripts, 250-252
- IP addresses, 533-534
- IP masquerading, 342
- IPv4 addressing, 343
- IPv6, 343-347
- IRC (Internet Relay Chat), 60-61, 805
- IRC server, 61
- isset() function (PHP), 748
- iwconfig tool, 371
- iwlist tool, 371
- iwpriv tool, 371
- iwsy tool, 371

J

Java, 53, 772, 781
 Java Runtime Environment (JRE), 781
 Java Virtual Machine (JVM), 781
 JavaScript, 772-773
 Javascript Object Notation (JSON), 571
 Jetty, 495
 JPG (graphics format), 86
 JRE (Java Runtime Environment), 781
 JSON (Javascript Object Notation), 571
 juju, 634-640

- charms, 638-640
- installation, 635-637
- managing sets of servers, 641-642

 JVM (Java Virtual Machine), 781

K

kate text editor, 201
 KDE, 101-102
 KDE ark archiving tool, 320-322
 KDE process, 305
 KDevelop client, 762-763
 KDevelop Setup Wizard, 762
 kdf tool, 305
 keep-one-running tool, 663
 kernel

- built-in security, 391
- interacting via /proc directory, 144-145
- management, 417-436
 - compiling the kernel, 427-435
 - Linux kernel, 418-422
 - modular kernels, 422-424
 - obtaining sources, 426
 - patching the kernel, 426-428

- recompiling the kernel, 424-425
- troubleshooting during compile, 435-436
- versions, 425
- performance tuning, 410-411
- tuning disk drives, 406-407

Kernel-based Virtual Machine (KVM), 599-603
 kernel source tree, 419-421
 key-based logins, 385-387
 key/value stores, NoSQL databases, 571-574
 Kile, 73
 kill command, 293-294
 Kirkland, Dustin, 677
 Kmail mail client, 58
 Knoppix, 104
 KOffice, 69-71
 ksort() function (PHP), 744
 KSpread, 71
 ksysguard tool, 305
 Kubuntu, 101-102
 KVM (Kernel-based Virtual Machine), 599-603
 KWord, 70

L

LAN, enabling network printing, 454
 Landscape, 306, 640, 643
 LANG environment variable, 197
 lapd-utils package, 582
 large enterprise users, backup programs, 311
 last looping construct (Perl), 692
 LaTeX, 73
 Launcher (Unity desktop), 43
 Launchpad, 648, 651-652

- accounts, 668
- environment, 668-670

- LDAP (Lightweight Directory Access Protocol), 581-588
 - administration, 587-588
 - client configuration, 586-587
 - server configuration, 582-585
- ldapadd utility, 587
- ldapdelete utility, 588
- ldapmodify utility, 587
- ldapsearch utility, 587
- LDM (LTSP Display Manager), 594
- less command, 140, 159, 171, 180-182
- levels of backup, 312
- libraries, mobile development for Android, 780
- LibreOffice, 66-68
- licensing, 2-3
- Liferea, RSS feeds, 58-59
- lighttpd, 493
- Lightweight Directory Access Protocol. See LDAP
- Linux
 - C programming tools, 755-757
 - directories, 142
 - file system hierarchy, 141-146
 - Internet resources, 797-805
 - kernel management, 418-422
 - kernel source tree, 419-421
 - types of kernels, 421-422
 - overview, 787-790
 - Perl, 681-684
- Linux Documentation Project, 379
- Linux file system, 146-149
- Linux Kernel
 - loading, 283
 - mobile development for Android, 780
- Linux Terminal Server Project. See LTSP
- Lisp, 773
- Listen directive, Apache web server configuration, 471
- listening to music, 79-82
- lists, Python, 710-712
- Live Bookmarks, 58
- In command, 171, 182-183
- loading, Linux Kernel, 283
- local GUI clients, database access, 563-564
- localhost interface, 336-337
- locate command, 171, 184
- log files (Apache), 488
- LogFormat statements (Apache), 488-490
- logging, Apache web server, 488-490
- logging in, command line, 137-138
- logging out, command line, 137-138
- logical operators, writing shell scripts, 265-270
- logname command, 234
- logout command, 137
- logs, purging in Perl, 697-698
- Long Term Support (LTS), 25
- loopback interface
 - availability, 336
 - configuration, 336-337
- looping constructs (Perl), 690-693
- loops
 - PHP programming, 737-739
 - Python, 713-715
- lp command, 460
- lpc command, 460
- lpq command, 460
- lprm command, 460
- lpstat command, 460
- ls command, 146-150, 171, 184-186
- LTS (Long Term Support), 25
- LTSP (Linux Terminal Server Project), 589-595
 - features, 594-595
 - installation, 593-594
 - requirements, 590-593

LTSP Display Manager (LDM), 594
 ltsp-server-standalone package, 593
 Lua, 773-774
 Ubuntu, 103-104
 LXDE, 103-104

M

MaaS (Metal as a Service), 610, 639
 MAC (Mandatory Access Control) system,
 AppArmor, 401-403
 machine protection, 394-397
 devices, 397
 passwords and physical security, 395-396
 Tripwire, 396-397
 wireless networks, 395
 macros, 758-759
 Mago, 677
 mail, sending in Perl, 695-697
 mail delivery agents (MDAs), 510-511, 521-522
 mail relaying, 516
 mail server options, 518-519
 mail transfer agents (MTAs), 507-510
 mail user agent (MUA), 511-512
 mailing lists, 804-805
 Major version (kernel), 425
 make command, 171, 757
 make utility (kernel), 420
 make xconfig tool, configuring the kernel,
 431-432
 makefile targets, 759
 makefiles, 757-759
 man command, 140, 172, 186-187
 man pages, 140
 management
 configuration, 130-131
 kernel, 417-436
 compiling the kernel, 427-435
 Linux kernel, 418-422
 modular kernels, 422-424
 obtaining sources, 426
 patching the kernel, 426-428
 recompiling the kernel, 424-425
 troubleshooting during compile, 435-436
 versions, 425
 modular kernels, 422-424
 OpenStack instances, 632
 OpenStack network, 633-634
 OpenStack storage, 633
 passwords, 222-227
 changing in a batch, 227
 password file, 223-224
 policy, 222-223
 security, 225-227
 shadow passwords, 224-225
 photos, Shotwell Photo Manager, 88
 sets of servers
 CFEngine, 643
 Chef, 642
 juju, 641-642
 Landscape, 643
 Puppet, 642
 Software. See software management
 Users. See user management
 Mandatory Access Control (MAC) system,
 AppArmor, 401-403
 manual configuration
 loopback interface, 336-337
 PPPoE, 377-378
 quotas, 233-234
 Samba, 444-447

- manual start, Apache web server, 467-468
- masquerading, Postfix, 514
- Master Boot Record (MBR), 10, 283, 673
- Masters of the Universe (MOTUs), 23
- Math (LibreOffice component), 67
- MBR (Master Boot Record), 10, 283
- mc command, copying files, 328-330
- MDAs (mail delivery agents), 510-511, 521-522
- measuring, MySQL buffer usage, 412-413
- media, data backup, 314-316
- Memcached, 573
- MemcachedDB, 573
- Mercurial, 648-649
- message body (newsgroup articles), 63
- message delivery interval (email), 515-516
- Metacity, 100
- Metal as a Service (MaaS), 610, 639
- methods
 - Python lists, 712
 - Python strings, 709
- Microsoft Exchange Server, 522-524
- Microsoft Windows, productivity applications, 73-74
- Midnight Commander, copying files, 328-330
- MIME (Multipurpose Internet Mail Extensions), 63
- Minor version (kernel), 425
- mirroring data, 314
- mkdir command, 151, 155-156, 172, 187
- mnemonic characters, 150
- mobile development for Android, 779-785
 - Android Runtime, 780
 - Application Framework, 780-781
 - core applications, 781
 - creating application, 784-785
 - installation of Android SDK, 781-783
 - libraries, 780
 - Linux kernel, 780
 - mobile network sniffing, 395
- mod_access (Apache), 481
- mod_alias (Apache), 481
- mod_asis (Apache), 481
- mod_auth (Apache), 482
- mod_auth_anon (Apache), 482
- mod_auth_dbm (Apache), 482
- mod_auth_digest (Apache), 483
- mod_cgi (Apache), 483
- mod_dir (Apache), 483
- mod_env (Apache), 483
- mod_expires (Apache), 483
- mod_headers (Apache), 483
- mod_include (Apache), 484
- mod_info (Apache), 484
- mod_log_config (Apache), 484
- mod_mime (Apache), 484
- mod_mime_magic (Apache), 484
- mod_negotiation (Apache), 484
- mod_proxy (Apache), 484
- mod_rewrite (Apache), 484
- mod_setenvif (Apache), 485
- mod_speling (Apache), 485
- mod_ssl (Apache), 485
- mod_status (Apache), 485
- mod_unique_id (Apache), 485
- mod_userdir (Apache), 485
- mod_usertrack (Apache), 485
- mod_vhost_alias (Apache), 485
- modular kernels, management, 422-424
- Module section (xorg.conf file), 36, 38
- modules
 - Apache web server, 480-486
 - disabling, 481
 - enabling, 481
 - Perl, 695
 - MongoDB, 575-576

Monitor section (xorg.conf file), 36, 39

monitoring tools, 291-306

- console-based monitoring, 291-298
 - disk quotas, 298
 - disk space, 297-298
 - free and used memory, 296-297
 - kill command, 293-294
 - priority scheduling and control, 294-296
- enterprise servers, 305-306
- graphical process, 298-305
 - Conky, 300-305
 - System Monitor, 298
- KDE process, 305
- user activity, 222

Mono, 774

MOTUs (Masters of the Universe), 23, 673

mounting shares, Samba, 449-450

MOV (video format), 95

Mozilla Public License, 3

Mozilla Thunderbird, 56

MP3 (sound format), 78

MPEG (video format), 95

MPMs (multiprocessing modules), Apache web server configuration, 473

MTAs (mail transfer agents), 507-510

mtr tool, checking network connections, 338-340

MUA (mail user agent), 511-512

multicast addressing, 348

multiline loops, Python, 714

multimedia applications, 75-97

- burning CDs/DVDs, 89-93
- digital cameras, 88
- graphics manipulation, 83-87
- sound and music, 75-81
- viewing video, 94-97

multiple terminals, command line, 205-206

multiprocessing modules (MPMs), Apache web server configuration, 473

Multipurpose Internet Mail Extensions (MIME), 63

Multiverse repository, 122

music, 75-81

- adjusting volume, 77
- listening to, 79-82
- sound cards, 76-77
- sound formats, 78-79

Mutt mail client, 57

mv command, 157-158, 172, 187

MySQL

- client/server system, 561-567
- command-line client, 564-566
- configuration, 554-558
- performance tuning, 412-416
 - measuring buffer usage, 412-413
 - query cache, 414-415
 - query optimization, 416
 - versus PostgreSQL, 552-554
- mysql -u root command, 555

N

Nagios, 306

name-based virtual hosts (Apache), 486-488

nano text editor, 200-202

NAT (Network Address Translation), 343

nautilus-image-converter package, 87

navigating Linux file system, 146-149

NcFTPd, 499

negative indexes, Python strings, 709

Neo4j, 578

- Nessus, assessing security vulnerabilities, 394
- NetBeans, 772
- netpbm tools, 87
- Network Address Translation (NAT), 343
- network cable, 351-352
- Network File System. See NFS
- Network Interface Cards. See NICs
- network interface configuration, command line, 356-360
- network management, OpenStack, 633-634
- Network Manager, configuring wireless networks, 29-30
- Network News Transfer Protocol (NNTP), 63
- network storage, 315
- networking
 - checking connections, 338-340
 - configuration tools, 355-364
 - DHCP, 365-371
 - hardware, 349-355
 - hubs and switches, 352
 - initializing new hardware, 353-355
 - network cable, 351-352
 - NIC (Network Interface Cards), 349-351
 - routers and bridges, 353
 - Internet connectivity, 374-380
 - commands, 380
 - common configuration information, 374-375
 - Dial-Up access, 378-379
 - DSL access, 376-378
 - troubleshooting connection problems, 379-380
 - IPv6, 344-347
 - localhost interface, 336-337
 - loopback interface availability, 336
 - loopback interface configuration, 336-337
 - organization, 347-348
 - broadcast addressing, 348
 - multicast addressing, 348
 - subnet masks, 348
 - subnetting, 347-348
 - Unicast addressing, 348
 - printers, 453-460
 - avoiding support problems, 458-460
 - creating network printers, 454-455
 - CUPS, 456-458
 - TCP/IP, 340-344
 - addressing, 341-342
 - IP masquerading, 342
 - ports, 344
 - wireless, 371-374
- newsgroups, Usenet, 61-64, 803-804
- newusers command, 234
- next looping construct (Perl), 692
- Nexuiz, 110
- NFS (Network File System), 440-442
 - client configuration, 442
 - installation, 440
 - server configuration, 440-442
- Nginx, 491-493
- nice command, 294
- NICs (Network Interface Cards), 349-351
 - 10BASE-T, 349
 - 100BASE-T, 349-350
 - 1000BASE-T, 350
 - Token Ring, 349
- Nmap, assessing security vulnerabilities, 394
- NNTP (Network News Transfer Protocol), 63
- node controller, Eucalyptus, 613
- nodes, Eucalyptus, 613
- nonbooting hard drive, 333-334

NoSQL databases, 545, 569-578

- document stores, 574-576

- graph stores, 577-578

- key/value stores, 571-574

- wide column stores, 576-577

Nova (Compute Infrastructure) service,
OpenStack, 618-619

number comparison, writing shell scripts,
262-264, 267

numbers, Python, 705-707

Nvidia, proprietary drivers, 108

O

object orientation (Python), 716-720

object variables (Python object orientation),
717-718

obtaining sources (kernel), 426

octal characters, 150

Ogg-Vorbis (sound format), 78

OGV/OGG (video format), 95

one-liners (Perl), 699-700

OneConf, 131

oops (kernel), compiling kernel, 436

Open Sound System (OSS), 76

open source, proprietary drivers *versus*, 24

Open-Xchange, 524

OpenLDAP, 581

OpenSSH server, 383

openssh-server package, 593

OpenStack, 607, 618-634

- commands, 634

- Compute Infrastructure (Nova), 618-619

- creating an image, 629-632

- Imaging Service (Glance), 619

- installation, 619-629

- instance management, 632

- network management, 633-634

- Storage Infrastructure (Swift), 619

- storage management, 633

operation, Postfix, 512-517

operators

- Perl, 686

- additional operators, 686

- arithmetic operators, 686

- comparison operators, 686-687

- compound operators, 686

- PHP programming, 731-733

opportunistic development, 645-663

- Bikeshed, 661-663

- Ground Control, 657-661

- Launchpad, 651-652

- Quickly, 653-656

- version control systems, 646-650

- Bazaar, 647-648

- Gilt, 649-650

- Mercurial, 648-649

- Subversion, 646-647

optimization, 405-416

- Apache, 411-412

- hard disk, 405-396

- badblocks command, 409

- disabling file access time, 409

- e2fsck command, 409

- file system settings, 408

- hdparm command, 407-408

- tune2fs command, 408-409

- tuning disk drives, 406-407

- kernel, 410-411

- MySQL, 412-416

- measuring buffer usage, 412-413

- query cache, 414-415

- query optimization, 416

- Options directive, Apache web server configuration, 474
- Oracle Beehive, 524
- order statement, Apache web server access control, 476
- organization, networking, 347-348
 - broadcast addressing, 348
 - multicast addressing, 348
 - subnet masks, 348
 - subnetting, 347-348
 - Unicast addressing, 348
- OrientDB, 578
- OSS (Open Sound System), 76
- Outlook, Microsoft Exchange Server, 523
- overview
 - Linux, 787-790
 - Ubuntu, 790-795

P

- PaaS (Platform as a Service), 609
- Package Browsing screen (Software Center), 119
- packaging code, 670-672
- packaging-dev package, installation, 667-668
- packet writing, 93
- PAM (Pluggable Authentication Modules), 226
- panel (Unity desktop), 47-48
- parameters, positional, 253-254
- partition strategies, Ubuntu installation, 10
- passwd command, 217, 234
- password file, 223-224
- passwords, 15
 - management, 222-227
 - changing in a batch, 227
 - password file, 223-224
 - policy, 222-223
 - security, 225-227
 - shadow passwords, 224-225
 - MySQL root users, 555-556
 - security, 395-396
- pastebinit tool, 662
- patch command, 428
- patching the kernel, 426-428
- PATH environment variable, 197
- pattern-matching support (shells), 245-246
- pb get tool (Bikeshed), 662
- pbput tool (Bikeshed), 662
- pbputs tool (Bikeshed), 662
- pbuilder, 668
- pci=biosirq command, 407
- PCRE (Perl-Compatible Regular Expressions), 749
- PCX (graphics format), 86
- PDF, 71-72
- PEAR project, 752
- performance tuning, 405-416
 - Apache, 411-412
 - hard disk, 405-396
 - badblocks command, 409
 - disabling file access time, 409
 - e2fsck command, 409
 - file system settings, 408
 - hdparm command, 407-408
 - tune2fs command, 408-409
 - tuning disk drives, 406-407
 - kernel, 410-411
 - MySQL, 412-416
 - measuring buffer usage, 412-413
 - query cache, 414-415
 - query optimization, 416
- Perl (Practical Extraction and Report Language), 681-700
 - accessing the shell, 694
 - code examples, 695-700

- command-line errors, 683
- command-line processing, 700
- commands, 700
- conditional statements, 689-690
- CPAN (Comprehensive Perl Archive Network), 695
- data structures, 684-686
- installation of CPAN module, 697
- Linux, 681-684
- looping constructs, 690-693
- modules, 695
- one-liners, 699-700
- operators, 686
 - additional operators, 686
 - arithmetic operators, 686
 - comparison operators, 686-687
 - compound operators, 686
- purging logs, 697-698
- regular expressions, 693
- sending mail, 695-697
- simple Perl program example, 682-684
- string constants, 686
- Usenet posts, 698-699
- variables, 684-686
- versions, 682
- Perl-Compatible Regular Expressions (PCRE), 749
- permissions
 - command line, 149-155
 - assigning permissions, 150-151
 - chmod command, 152-153
 - directory permissions, 151-152
 - file permissions, 153
 - set UIDs/GID permissions, 153-155
 - file, 212-213
- personal home page programming. *See* PHP programming
- personal package archive (PPA), Launchpad, 652
- personal video recorders, 97
- photo management, Shotwell Photo Manager, 88
- Photoshop, 84
- PHP mode, entering/exiting, 724
- PHP programming, 723-754
 - arrays, 725-727
 - comments, 729
 - conditional statements, 733-734
 - constants, 728
 - databases, 751-754
 - entering/exiting PHP mode, 724
 - escape sequences, 729-730
 - functions, 740-751
 - arrays, 743-745
 - files, 745-747
 - miscellaneous functions, 747-750
 - strings, 740-743
 - HTML forms, 751
 - including other files, 739-740
 - loops, 737-739
 - operators, 731-733
 - references, 728-729
 - special operators, 734-735
 - switching, 735-737
 - variable substitution, 730-731
 - variables, 724-725
- phpgroupware, 524
- PHPorjekt, 524
- physical security, 395-396
- PID (process ID), 283
- Pidgin, 60
- ping command, checking network connections, 338-340
- pinging data, shell control, 247

- Planet Debian, 59
- Planner (LibreOffice component), 68
- Platform as a Service (PaaS), 609
- plug-ins, Firefox, 53
- Pluggable Authentication Modules (PAM), 226
- PNG (graphics format), 86
- Point-to-Point Protocol over Ethernet. *See* PPPoE
- policy, password management, 222-223
- POP3 (Post Office Protocol version 3), 511
- portable anymap file format, 87
- portable bitmap file format, 87
- portable graymap file format, 87
- portable pixmap file format, 87
- ports, TCP/IP networking, 344
- positional parameters, executing shell scripts, 253-254
- POSIX Extended regular expressions, 749
- post-installation configuration problems, 31-32
- Post Office Protocol version 3 (POP3), 511
- postconf command, 513
- Postfix, 509
 - configuration and operation, 512-517
 - masquerading, 514
- PostgreSQL
 - client/server system, 561-567
 - command-line client, 566
 - configuration, 558-561
 - creating database, 559
 - deleting users, 560-561
 - users, 559-560
 - versus MySQL, 552-554
- posting to Usenet (Perl), 698-699
- power management, configuring system settings, 27
- power shortcuts, Unity desktop, 49
- PPA (personal package archive), Launchpad, 652
- pppd daemon, 378
- PPPoE (Point-to-Point Protocol over Ethernet), 376-378
- Practical Extraction and Report Language. *See* Perl
- preg_match() function (PCRE), 749
- preg_match_all() function (PCRE), 749
- preg_replace() function (PCRE), 750
- printenv command, 198, 229
- printers
 - configuring system settings, 26-27
 - sharing, 453-460
 - avoiding support problems, 458-460
 - creating network printers, 454-455
 - CUPS, 456-458
- printing commands, 460
- priority scheduling, console-based monitoring tools, 294-296
- private cloud, Eucalyptus, 612, 613-616
- privileges
 - MySQL databases, 556
 - PostgreSQL databases, 561
- procedural languages, MySQL versus PostgreSQL, 554
- process ID (PID), 283
- Processes tab (System Monitor), 298
- Procmail, 521
- productivity applications, 65-74
 - GNOME Office, 69-70
 - KOffice, 69, 70-71
 - LaTeX, 73
 - LibreOffice, 66-68
 - Microsoft Windows, 73-74
 - PDF, 71-72
 - XML and DocBook, 71-72
- ProFTPD, 499

programming languages, 767-776

- Ada, 768
- Clojure, 768-769
- COBOL, 769
- Erlang, 770
- Forth, 770
- Fortran, 771
- Go, 771
- Groovy, 771
- Haskell, 772
- Java, 772
- JavaScript, 772-773
- Lisp, 773
- Lua, 773-774
- Mono, 774
- Perl. See Perl
- Ruby, 774-775
- Rust, 775
- Scala, 775
- Scratch, 776
- Vala, 776

programming tools, 755-763

- autoconf utility, 759-760
- C with Linux, 755-757
- debugging tools, 760-761
- GNU C compiler, 761-762
- graphical development, 762-764
- macros, 758-759
- make command, 757
- makefile targets, 759
- makefiles, 757-759

promiscuous mode, 397

proprietary drivers, open source *versus*, 24

proprietary video drivers, 108-109

protection (security). See security

protection of data, MySQL *versus* PostgreSQL, 553-554

proxy servers, 527-535

- access control lists, 529-533
- client configuration, 528-337
- client IP addresses, 533-534
- defined, 527-528
- sample configuration, 534-535
- Squid installation, 528

ps command, 172, 188

public cloud, Eucalyptus, 612-613

PulseAudio, 76-77

Puppet, managing sets of servers, 642

Puppet Forge, 642

purging logs (Perl), 697-698

pwd command, 137, 149

PWD environment variable, 197

PyPI (Python Package Index), 721

Python, 653, 703-721

- conditional statements, 713-715
- dictionaries, 712-713
- executing scripts, 704-705
- functions, 715-716
- lists, 710-712
- loops, 713-715
- numbers, 705-707
- object orientation, 716-720
 - class and object variables, 717-718
 - class inheritance, 719-720
 - constructors and destructors, 718-719
- PyPI (Python Package Index), 721
- Standard Library, 721
- strings, 707-710

Python 2.x, 703

Python 3.x, 703

Python Package Index (PyPI), 721

Q

Q Public License, 3

QA, 675-680

 Bug Squad, 677

 QA Team, 676-677

 Test Drive, 677-680

Qmail, 509

QT (video format), 95

Quassel, 60

query cache, tuning MySQL, 414-415

query optimization, MySQL, 416

Quickly, 653-656

quotacheck command, 233

quotaoff command, 233

quotaon command, 233

quotas, disk, 232-234

R

RAID arrays, 314

RAM disk image, configuring kernel, 432-435

RAW (sound format), 78

rcp command, 384

RDBMSs (relational database management systems). See relational database services

RDP (Remote Display Protocol), 604

recompiling the kernel, 424-425

recorders, personal video, 97

Redis, 573

reduced instruction set computer (RISC) processors, 780

redundant array of independent disks.
 See RAID arrays

references 795, 728-729

regexes. See regular expressions, Perl

regular expressions, 159, 693

regular users, 211

 granting system administrator privileges to, 227-232

 changing UID, 227-229

 root privileges, 229-232

relational database services, 543-567

 client/server system, 561-567

 commands, 567

 creating tables, 548-549

 how they work, 545-547

 inserting data in tables, 549-550

 MySQL

 configuration, 554-558

 versus PostgreSQL, 552-554

 PostgreSQL, 552-554, 558-561

 retrieving data, 550-552

 SQL basics, 547-548

release-build tool (Bikeshed), 662

release tool (Bikeshed), 662

reload command (AppArmor), 403

remote access, 381-389

 SSH (Secure Shell), 383-387

 Telnet, 381-383

 VNC (virtual network computing), 387-389

remote computer, logging in/out of command line, 137-138

Remote Desktop Viewer, 388

Remote Display Protocol (RDP), 604

remote file serving (FTP), 497-506

 configuring Very Secure FTP server, 502-505

 FTP server selection, 497-499

 FTP users, 500-502

 ftphosts file, 505-506

 servers, 498-499

 software installation, 499-500

- remote printing, 453-460
 - removable storage media, 314-315
 - renice command, 294
 - repeat statement, executing shell scripts, 274
 - repositories
 - Apache web server installation, 463-464
 - compiling from source, 129-130
 - software, 23-26
 - repquota command, 233
 - Requests for Comments (RFC), 63
 - requirements
 - hardware specifications, 8
 - LTSP, 590-593
 - rescue disk, 333
 - resources, Internet, 797-805
 - Resources tab (System Monitor), 298
 - responsibilities, DBAs (database administrators), 543-544
 - retrieving data, relational database services, 550-552
 - retrieving email, Fetchmail, 517-521
 - REVOKE statement
 - MySQL, 558
 - PostgreSQL, 561
 - revoking privileges, PostgreSQL databases, 561
 - RFC (Requests for Comments), 63
 - Rhythmbox, 79-80
 - Riak, 574
 - rights
 - MySQL databases, 556
 - PostgreSQL databases, 561
 - RISC (reduced instruction set computer) processors, 780
 - rm command, 157, 172, 188-189
 - rmdir command, 156
 - Root User, 210-212
 - root users
 - command line, 139, 160-166
 - MySQL, 555-556
 - Rootkit Hunter, 404
 - rootsign tool, 663
 - route command, 356-360
 - routers, 353
 - RSS feeds, 58-59
 - Firefox, 58
 - Liferea, 58-59
 - rsync command, copying files, 328-330
 - Ruby, 774-775
 - run-one tool, 663
 - run-this-one tool, 663
 - runlevels, BIOS (basic input/output system), 284
 - changing, 286-287
 - definitions, 284
 - troubleshooting problems, 287-288
 - running jobs repeatedly, cron command, 240-242
 - runtime errors, compiling kernel, 436
 - runtime server configuration settings, Apache web server, 469-474
 - directives, 470
 - editing apache2.conf, 470-473
 - .htaccess configuration file, 473-474
 - multiprocessing modules, 473
 - Rust, 775
- ## S
- SaaS (Software as a Service), 609
 - Samba, 450-453
 - configuration, 450-453
 - global behavior, 446

- manual configuration, 444-447
- mounting shares, 449-450
- smbd daemon, 448
- testing, 447-448
- Samba Web Administration Tool (SWAT), 443
- satisfy directive, Apache web server, 479
- saving files, nonbooting hard drive, 333-334
- Scala, 775
- scalar variables (Perl), 685
- scanners, 85
- schemas, configuring LDAP server, 582-583
- Scorched 3D, 110-111
- scp command, 384
- Scratch, 776
- Screen, 60
- screen images, graphics manipulation, 87
- Screen section (xorg.conf file), 37, 40-41
- script kiddies, 392
- scripts
 - command-line scripting. See Python
 - Perl. See Perl
 - shells. See shells, scripts
 - web scripting. See PHP programming
- SDK (software development kit), 780-781
- stderr stream, redirection, 193
- search engines, 798-803
- Secure Shell. See SSH
- security, 391-404
 - AppArmor, 401-403
 - computer attacks, 392
 - disaster recovery plan, 403-404
 - firewalls, 398
 - machine protection, 394-397
 - devices, 397
 - passwords and physical security, 395-396
 - Tripwire, 396-397
 - wireless networks, 395
 - passwords, 225-227
 - viruses, 397-398
 - vulnerability assessment, 393-394
- select statement, executing shell scripts, 274-275
- sending mail (Perl), 695-697
- Sendmail, 508-509
- sequences (Python)
 - lists, 710-712
 - strings, 707-710
- Server install DVD, 8
- server packages (FTP), 498
- ServerAdmin directive, Apache web server configuration, 472
- ServerAGroup directive, Apache web server configuration, 471
- ServerLayout section (xorg.conf file), 36-37
- ServerName directive, Apache web server configuration, 472
- ServerRoot directive, Apache web server configuration, 471
- servers
 - Apache web server. See Apache web server configuration for LDAP, 582-585
 - DHCP, 368
 - FTP, 497-499
 - HTTP, 491-496
 - Apache Tomcat, 496
 - Cherokee, 494-495
 - Jetty, 495
 - lighttpd, 493
 - Nginx, 491-493
 - thttpd, 495-496
 - YAWS, 494
 - juju, 641-642

- managing sets
 - CFEngine, 643
 - Chef, 642
 - juju, 641-642
 - Landscape, 643
 - Puppet, 642
- NFS configuration, 440-442
- SSH (Secure Shell), 383
- Telnet, 381-382
- service models (Cloud), 609-610
- service set identifier (SSID), 30
- service settings, /etc/services file, 361-362
- services
 - BIOS (basic input/output system), 286-287
 - boot process, 281-282
- Session Message Block (SMB), 443
- session message block printing, 455
- session writing, 92-93
- sets of servers, management
 - CFEngine, 643
 - Chef, 642
 - juju, 641-642
 - Landscape, 643
 - Puppet, 642
- sftp command, 383, 385
- shadow password file (FTP), 501
- shadow passwords, 224-225
- shared data, /usr directory, 145-146
- sharing
 - home directories, 446-447
 - printers, 453-460
 - avoiding support problems, 458-460
 - creating network printers, 454-455
 - CUPS, 456-458
- sharing files, 439-453
 - NFS (Network File System), 440-442
 - Samba, 450-453
- SHELL environment variable, 197
- Shell module, 694
- shells, 242-247
 - basic shell control, 242-247
 - background processing, 247
 - command line, 243-244
 - pattern-matching support, 245-246
 - piping data, 247
 - redirecting input/output, 246-247
 - endless loops, 272
 - scripts, 248-279
 - writing and executing, 248-279
 - functions, 279
- shift statement, executing shell scripts, 275
- Shockwave, 53
- shortcuts, Unity desktop, 49
- Shotwell Photo Manager, 88
- show status command (AppArmor), 403
- shuffle() function (PHP), 744
- shutdown command, 18, 165-166
- shutting down Ubuntu, 18
- Simple Mail Transfer Protocol (SMTP), 507
- slapd package, 582
- Slashdot, 59
- Slashdot Effect, 411
- small enterprise users, backup programs, 311
- small office users, backup programs, 311
- smart hosts, 515
- SMB (Session Message Block), 443
- smbclient command, 449
- smbd daemon, 448
- smbstatus command, 448
- Smokin' Guns, 110
- SMTP (Simple Mail Transfer Protocol), 507
- socks-prox tool (Bikeshed), 662

software

- data backup, 316-325
- DHCP, 367-368
- installing FTP software, 499-500
- licensing, 2-3
- Software as a Service (SaaS), 609
- Software Center, 119-120
- software development kit (SDK), 780
- software management, 119-131
 - APT (Advanced Package Tool), 123-128
 - compiling applications from source, 128-130
 - configuration management, 130-131
 - Software Updater, 122-123
 - Synaptic, 120-122
 - Ubuntu Software Center, 119-120
- software repositories, 23-26
- Software Sources GUI tool, 24
- Software Updater, 19-22, 122-123
- sorting shell scripts, 250
- sound, 75-81
 - adjusting volume, 77
 - formats, 78-79
 - listening to music, 79-82
 - sound cards, 76-77
- sound cards, 76-77
- Sound Juicer, 81
- SourceForge, 647
- sources (kernel), 426
- Spamassassin, 521-522
- special characters, writing shell scripts, 257
- special operators, PHP programming, 734-735
- special variables (Perl), 685-686
- speed, MySQL *versus* PostgreSQL, 552
- Speed Dreams, 114-115
- splint command, 760-761

SQL

- basics, 547-548
- subqueries, 554
- Squid installation, 528
- Squirrelmail, 522
- SSH (Secure Shell), 60, 383-387
 - access to databases, 562
 - setting up server, 383
 - Telnet *versus*, 383
 - tools, 383-387
- ssh command, 172
- ssh-import-id tool, 663
- SSH key, 668
- ssh-keygen command, 385-387
- sshd daemon, 383
- SSID (service set identifier), 30
- standard input, 191
- Standard Library (Python), 721
- standard output, 191
- start command (AppArmor), 403
- starting
 - Apache web server, 467-469
 - NFS, 440
 - X Server sessions, 41
- Startup Applications, 286-287
- stdin stream, redirection, 193
- stdout stream, redirection, 193
- step-by-step installation (Ubuntu), 11-17
 - first update, 16
 - installing, 12-16
 - Wubi, 16-17
- stereotypes (users), 213
- stop command (AppArmor), 403
- stopping
 - Apache web server, 467-469
 - NFS, 440

Storage Infrastructure (Swift) service, OpenStack, 619

storage management, OpenStack, 633

str_replace() function (PHP), 741

stri_replace() function (PHP), 741

string comparison, writing shell scripts, 261-262

string constants (Perl), 686

strings

- PHP functions, 740-743
- Python, 707-710

strlen() function (PHP), 740

strpos() function (PHP), 742

StumbleUpon plug-in, 53

su command, 227-229, 234

Sublevel number version (kernel), 425

subnet masks, 348

subnetting, 347-348

subsections (kernel), 433

substr() function (PHP), 742

Subversion, 646-647

sudo command, 22-23, 160-163, 210, 234

Super User/Root User, 210-212

super users, 138-139, 160-166, 211

SuperTux, 112-113

support

- network printers, 458-460
- wireless networking, 371-372

suspend, 14

SVG (graphics format), 86

SWAT (Samba Web Administration Tool), 443

Swift (Storage Infrastructure) service, OpenStack, 619

switch/case block, 736

switches, 352

switching, PHP programming, 735-737

symbolic debugging, 761

Synaptic, software management, 120-122

SysAdmin *versus* DevOps, 608

system administrators, granting privileges to regular users, 227-232

system function call, 694

System Monitor, 298

system-monitoring tools. See monitoring tools

system rescue, 332-334

system-search tool (Bikeshed), 662

system settings

- configuration, 26-29
- printers, 26-27
- time and date, 27-29
- pconfiguration, power management, 27
- Unity, 48-49

System tab (System Monitor), 298

system users, 212

T

tables, relational database services, 548-550

tail command, 172, 189

tape drives, 315-316

tar backup tool, 317-319

tar command, 205, 326-327

Tarball, 128-129

task automation, 237-242

- scheduling tasks, 237-242
- writing shell scripts, 255-256

TCP/IP (Transport Control Protocol/Internet Protocol), 340-344

- addressing, 341-342
- IP masquerading, 342
- ports, 344

Telnet

- setting up server, 381-382
- SSH *versus*, 383

temporary file storage, 146
 TERM environment variable, 197
 terminal. See command line
 ternary operator, 734-735
 Test Drive tool, 677-680
 testing, 675-680

- Bug Squad, 677
- community teams, 675-677
- Samba, 447-448
- Test Drive, 677-680

 testparm command, 447-448
 text-based console login, 136-137
 text editors, command line, 200-204

- emacs, 203-204
- nano, 201-202
- vi, 202-203

 tftpd, 495-496
 Thunderbird, configuring for LDAP, 587
 TIF (graphics format), 86
 Time, configuring system settings, 27-29
 time command, 294
 Token Ring, 349
 tools

- Eucalyptus, 616-618
- group management, 214-216
- SSH (Secure Shell), 383-387
- system-monitoring. See monitoring tools

 top command, 172, 189-191, 295
 Totem Movie Player, 96
 touch command, 149, 155, 338-340, 386
 traceroute tool, 338-340
 Transport Control Protocol/Internet Protocol.
 See TCP/IP
 trim() function (PHP), 740
 triple quoting, Python strings, 709
 Tripwire, 396-397

troubleshooting

- BIOS runlevel problems, 287-288
- Internet connection problems, 379-380
- kernel compile, 435-436
- post-installation configuration problems, 31-32

 tune2fs command, 408-409
 TuxPaint, 114
 TV hardware, 94-95
 types of kernels, 421-422

U

Ubuntu Cloud. See Cloud
 Ubuntu One cloud storage, 64
 Ubuntu One Music Store, 81
 ubuntu-restricted-extras package, 96
 ubuntu-qa-tools, 677
 Ubuntu Server Edition, 607
 Ubuntu Software Center, 119-120
 Ubuntu Testing Team, 676
 UDP (Universal Datagram Protocol), 340
 UFW (Uncomplicated Firewall), 398-399
 UIDs (user IDs), 153-155, 212
 umask command, 149
 Uncomplicated Firewall (UFW), 398-399
 unexpanded variables, writing shell scripts, 259-260
 Unicast addressing, 348
 Unison backup tool, 324
 Unity desktop, 33-50

- customizing and configuring, 48-49
- Dash, 44-47
- Launcher, 43
- panel, 47-48
- power shortcuts, 49
- workspaces, 43

- X Server, 33-42
 - basic concepts, 34-35
 - changing window managers, 42
 - display manager, 41-42
 - elements of xorg.conf file, 36-41
 - starting X sessions, 41
 - X.org software, 35
- Universal Datagram Protocol (UDP), 340
- Universe repository, 23, 122
- unless conditional statements (Perl), 690
- UnQL (Unstructured Query Language), 571
- unset() function (PHP), 748
- Unshielded Twisted Pair (UTP) network cable, 351
- Unstructured Query Language (UQL), 571
- until looping construct (Perl), 692
- until statement, executing shell scripts, 273-274
- UPG (user private group), 213
- upgrade option, 22
- Upstart, 289-290
- uptime command, 297
- uquick tool (Bikeshed), 662
- Urban Terror, 110
- USB drive, Ubuntu installation, 11
- USB thumb drive, Ubuntu installation, 9
- Usenet newsgroups, 61-64, 803-804
- Usenet posts (Perl), 698-699
- user accounts, 209-213
 - adding, 218-221
 - command line, 138-139
 - commands, 234
 - configuring Fetchmail, 519-521
 - file permissions, 212-213
 - stereotypes, 213
 - Super User/Root User, 210-212
 - user IDs/group IDs, 212
- User directive, Apache web server configuration, 471
- user directories, 143-144
- USER environment variable, 197
- user files, Apache web server, 478-479
- user IDs (UIDs), 153-155, 212
- user management, 209-234
 - disk quotas, 232-234
 - groups, 213-216
 - group listing, 213-214
 - management tools, 214-216
 - passwords, 222-227
 - changing in a batch, 227
 - password file, 223-224
 - policy, 222-223
 - security, 225-227
 - shadow passwords, 224-225
 - system administrator privileges, 227-232
 - root privileges, 229-232
 - temporarily changing user identity, 227-229
- user accounts, 209-213
 - commands, 234
 - file permissions, 212-213
 - stereotypes, 213
 - Super User/Root User, 210-212
 - user IDs/group IDs, 212
- user private group (UPG), 213
- useradd command, 215-216, 234
- UserDir directive, Apache web server configuration, 472
- usermod command, 215, 218, 234
- users
 - FTP, 500-502
 - PostgreSQL, 559-561
- UTP (Unshielded Twisted Pair) network cable, 351

V

Vala, 776

var_dump() function (PHP), 748

variable substitution, PHP programming, 730-731

variables

class and object (Python object orientation), 717-718

executing shell scripts, 257

LogFormat statements (Apache), 488-490

Perl, 684-686

PHP programming, 724-725

shell scripts, 252

variants (Ubuntu), 8

VBA (Visual Basic for Applications), 65

version control systems

data backup, 330-332

opportunistic development, 646-650

Bazaar, 647-648

Gilt, 649-650

Mercurial, 648-649

Subversion, 646-647

versions

kernel, 425

Perl, 682

PHP programming, 723

Ubuntu, 791

Very Secure FTP server, 502-505

vi text editor, 202-203

video

editing, 97-98

formats, 95

hardware, 94-95

personal video recorders, 97

viewing, 94-97

video conferencing, Empathy, 59

viewing video, 94-97

vim command, 172

vim text editor, 200

virtual devices, installation, 783

virtual hosting, Apache web server, 486-488

virtual machines (VMs), 598, 607

virtual network computing (VNC), 387-389

virtual private networks. See VPNs

VirtualBox, 603-604

virtualization, 597-605

KVM (Kernel-based Virtual Machine), 599-603

VirtualBox, 603-604

VMware, 605

Xen, 605

Virus Scanners, 522

viruses, 397-398

Visual Basic for Applications (VBA), 65

VMs (virtual machines), 598, 607

vmstat tool, 297

VMware, 605

VNC (virtual network computing), 387-389

vncviewer, 305

volume adjustment, music and sound, 77

VPNs (virtual private networks), 536-541

vsftpd server configuration files, 504-505

VT-x extension, 599

vulnerability assessment, 393-394

W

war driving, 395

Warsow, 110

WAV (sound format), 78

web access, databases, 563-564

web scripting. See PHP

- web servers, 491-496
 - Apache. See Apache web server
 - Apache Tomcat, 496
 - Cherokee, 494-495
 - Jetty, 495
 - lighttpd, 493
 - Nginx, 491-493
 - thttpd, 495-496
 - YAWS, 494
- WEBM (video format), 95
- websites, 798-803
- WEP encryption, 30
- what-provides tool (Bikeshed), 662
- whereis command, 141
- which command, 172, 191
- while loop
 - PHP, 737
 - Python, 714
- while looping construct (Perl), 691-692
- while statement, writing shell scripts, 271-273
- wide column stores, NoSQL databases, 576-577
- wifi-status tool (Bikeshed), 662
- wildcards, 159
- window managers, 42
- Windows games, 116
- Windows installer (Wubi), 16-17
- Wine application, 74, 116
- wireless network interfaces, 350-351
- wireless networks, 371-374
 - configuration, 29-30
 - security, 395
- Wireshark, 395, 501
- Wolfenstein: Enemy Territory, 110
- workspaces (Unity desktop), 43
- World of Padman, 110
- worms, 392
- WPA encryption, 30
- Writer (LibreOffice component), 67
- writing shell scripts, 248-279
 - accessing variable values, 253
 - assigning value to variables, 252-253
 - automation of tasks, 255-256
 - backslash, 260
 - backtick, 260
 - break statement, 278
 - built-in variables, 257
 - case statement, 276-278
 - comparison of expressions, 261-266
 - comparison of expressions with tcsh, 266-270
 - exit statement, 278
 - for statement, 270-271
 - if statement, 275-276
 - interpreting shell scripts, 250-252
 - positional parameters, 253-254
 - repeat statement, 274
 - running shell program, 249-250
 - select statement, 274-275
 - shift statement, 275
 - sorting scripts for access, 250
 - special characters, 257
 - strings with embedded spaces, 257-259
 - unexpanded variables, 259-260
 - until statement, 273-274
 - variables, 252
 - while statement, 271-273
- Wubi (Windows installer), 16-17

X

- X.org software, 35
- X Server, 33-42
 - basic concepts, 34-35
 - changing window managers, 42
 - display manager, 41-42
 - elements of xorg.conf file, 36-41
 - Device section, 37, 39-40
 - Files section, 36-38
 - InputDevice section, 36, 38-39
 - Module section, 36, 38
 - Monitor section, 36, 39
 - Screen section, 37, 40-41
 - ServerLayout section, 36-37
 - starting X sessions, 41
 - X.org software, 35
- Xamarin, 774
- xargs command, 172
- XChat, 60-61
- XChat Channel List window, 61
- Xen, 605
- Xfce, 102-103

- Xmarks Sync plug-in, 53
- XML and DocBook, 71-72
- XML Copy Editor, 72
- xorg.conf file elements, 36-41
 - Device section, 37, 39-40
 - Files section, 36-38
 - InputDevice section, 36, 38-39
 - Module section, 36, 38
 - Monitor section, 36, 39
 - Screen section, 37, 40-41
 - ServerLayout section, 36-37
- Xubuntu, 102-103

Y

- YAWS (Yet Another Web Server), 494

Z

- Zenoss, 306